Collective Artificial Intelligence:
Simulated Role-Playing from Crowdsourced Data

by

Jeffrey David Orkin

B.S., Tufts University (1995)
M.S., University of Washington (2003)
M.S., Massachusetts Institute of Technology (2007)

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2013

Signature of Author . . . .
Program in Media Arts and Sciences
January 28, 2013

Certified by. . . . .
Deb K. Roy
Associate Professor of Media Arts and Sciences
Thesis Supervisor

Accepted by . . . .
Patricia Maes
Graduate Officer
Program in Media Arts and Sciences

Collective Artificial Intelligence:
Simulated Role-Playing from Crowdsourced Data

by

Jeffrey David Orkin

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on January 28, 2013, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Collective Artificial Intelligence (CAI) simulates human intelligence from data contributed by many humans, mined for inter-related patterns. This thesis applies CAI to social role-playing, introducing an end-to-end process for compositing recorded performances from thousands of humans, and simulating open-ended interaction from this data. The CAI process combines crowdsourcing, pattern discovery, and case-based planning. Content creation is crowdsourced by recording role-players online. Browser-based tools allow non-experts to annotate data, organizing content into a hierarchical narrative structure. Patterns discovered from data power a novel system combining plan recognition with case-based planning. The combination of this process and structure produces a new medium, which exploits a massive corpus to realize characters who interact and converse with humans. This medium enables new experiences in videogames, and new classes of training simulations, therapeutic applications, and social robots.

While advances in graphics support incredible freedom to interact physically in simulations, current approaches to development restrict simulated social interaction to hand-crafted branches that do not scale to the thousands of possible patterns of actions and utterances observed in actual human interaction. There is a tension between freedom and system comprehension due to two bottlenecks, making open-ended social interaction a challenge. First is the authorial effort entailed to cover all possible inputs. Second, like other cognitive processes, imagination is a bounded resource. Any individual author only has so much imagination.

The convergence of advances in connectivity, storage, and processing power is bringing people together in ways never before possible, amplifying the imagination of individuals by harnessing the creativity and productivity of the crowd, revolutionizing how we create media, and what media we can create. By embracing data-driven approaches, and capitalizing on the creativity of the crowd, authoring bottlenecks can be overcome, taking a step toward realizing a medium that robustly supports player choice. Doing so requires rethinking both technology and division of labor in media production.

As a proof of concept, a CAI system has been evaluated by recording over 10,000 performances in *The Restaurant Game*, automating an AI-controlled waitress who interacts in the world, and converses with a human via text or speech. Quantitative results demonstrate CAI supports significantly open-ended interaction with humans, while focus groups reveal factors for improving engagement.

Thesis Supervisor: Deb K. Roy
Title: Associate Professor of Media Arts and Sciences

Collective Artificial Intelligence:
Simulated Role-Playing from Crowdsourced Data

by

Jeffrey David Orkin

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degrees of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2013

Advisor . . . .

Deb K. Roy
Associate Professor of Media Arts and Sciences
MIT Media Lab

Collective Artificial Intelligence:
Simulated Role-Playing from Crowdsourced Data

by

Jeffrey David Orkin

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2013

Thesis Reader . . . .
Nick Montfort
Associate Professor of Digital Media
Massachusetts Institute of Technology

Collective Artificial Intelligence:
Simulated Role-Playing from Crowdsourced Data

by

Jeffrey David Orkin

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2013

Thesis Reader . . . .

Mark O. Riedl
Assistant Professor
Georgia Institute of Technology

Acknowledgements

Contents

List of Figures

List of Tables

1 Introduction

We are witnessing an unprecedented convergence of advances in processing power, interconnectivity, cheap data storage, high fidelity rendering, physics simulations, and natural language technologies. The combination of these technological innovations lays the foundation for new mediums of communication, allowing us to tell stories in ways never before possible. Each new communication medium changes both what stories can be told, and how stories can be told. As recounted by Marie-Laure Ryan (2006), Walter Ong describes how narrative evolved with the introduction of writing:

> The enormous impact of writing technology on thought, and, by extension, on narrative, can be captured in one brief formula: a permanent inscription serves as prosthetic memory. In oral cultures, narrative was used as a mnemonic device for the transmission of knowledge; its memorization was facilitated by prosodic features, fixed formulae, and standardized images; and the limitations of memory were compensated by a relatively free episodic structure which allowed, within reasonable limits, permutation of its units. The development of manuscript writing transformed this open epic structure into tightly knotted dramatic plot.... With its organization of events into an exposition, complication, crisis, and resolution, its symmetrical, carefully controlled rise and fall of tension (known as the Freytag triangle), and its climactic reversal of situation at the apex of the triangle, the dramatic plot exploits the significance of the sequential ordering of events to an extent that would not be possible in oral improvisation.

Technological advances in the 20th century have made possible new interactive mediums for storytelling, where the listener becomes the *player*, playing an active role in the story being told. Unlike static mediums (e.g. books, films), where there is a clear distinction between the storyteller and audience, the player of an interactive story is both an author and audience member of a dynamic experience that varies in response to player input, and the experience can unfold differently with subsequence replays of the same story. If writing serves as prosthetic memory, an interactive medium serves as prosthetic imagination, as the player explores the space of possible story variations, and the medium responds with consequences imagined and encoded by human authors.

Videogames are one application of interactive storytelling. Dan Houser, co-founder of Rockstar Games, expresses the unique opportunities for interactive storytelling mediums when he says, "Books tell you something, movies show you something, games let you do something."[1] With the incredible advances in rendering and physics simulation, the emphasis in commercial games has been on doing things physically. In Rockstar's titles, gameplay is dominated by combat and car crashes. Prescribed storylines with a relatively small number of discrete branches are punctuated with open-ended combat.

The alternative is do things with words. This idea dates back to discussions in the philosophy of language from the 1950s (Austin 1955) -- language is a form of action which can affect the world, just as physical action can. The implication is that players can affect the simulated world and the story through social role-playing interactions with Non-Player Characters (NPCs). Interleaved fluidly with physical interaction, open-ended dialogue with NPCs can help an interactive storytelling medium reach its full potential, dynamically adapting in satisfying ways to everything the player says and does. However, Natural

---

[1] "Q. and A.: Rockstar's Dan Houser on Grand Theft Auto V," *The New York Times*, November 9, 2012.

Language Understanding (NLU) is a hard problem due to the incredible diversity of things people say, and ways they can say them.

There is a tension between player freedom and system comprehension, which can be represented as the two-dimensional feature space illustrated in Figure 1-1. The Holy Grail exists in the top-right corner, where players have complete freedom, and the system can understand everything they say or do. Existing implementations of interactive stories reside closer to one axis or the other. For example, commercial videogames like *Mass Effect 3* (BioWare 2012) limit social interaction to multiple choice dialogue options, which restricts player freedom, while ensuring the system should comprehend choices the player makes. The experimental game *Façade* (discussed in more detail in Section 1.1) provides much more freedom at the cost of some system comprehension. *Façade* allows players to type open-ended dialogue input, and the system will understand words and phrases that were anticipated by designers, in the current context.
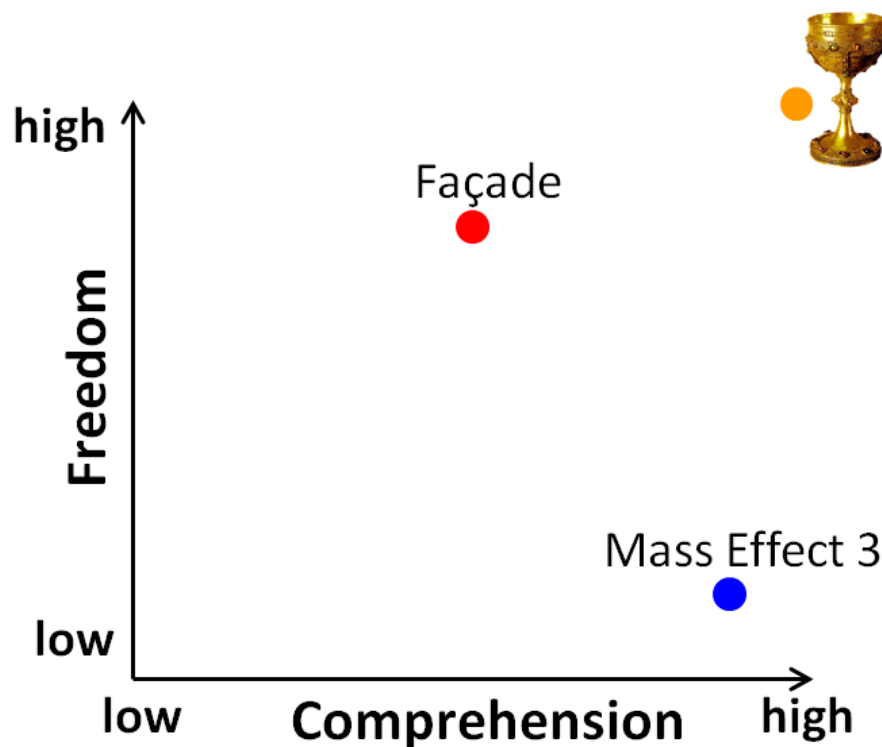
Figure 1-1: Space representing tension between player freedom & system comprehension.

There are two bottlenecks that make it difficult to simultaneously maximize both freedom and comprehension. The first bottleneck is the authorial effort it entails to adequately cover the space of possible natural language inputs. The second is the fact that imagination, like other cognitive processes, is a bounded resource. Any one individual author only has so much imagination. This thesis sets out to prove that due to recent shifts in the operating environment of storytellers, it is possible to stake out a new place in this 2D space closer to the Holy Grail, combining the freedom of *Façade*'s open-ended input with the assurance of comprehension provided by multiple choice interaction. Overcoming the tension between freedom and comprehension requires a medium that can exploit an enormous pool of content to support recognizing and responding to a wider range of interactions than any team of

designers could imagine and encode by hand, as well as new approaches for authoring and structuring this content.

The contribution of this thesis is an end-to-end process which applies Collective Artificial Intelligence (CAI) to the problem of open-ended simulated social role-playing. Interaction with NPCs is simulated from composited performances recorded from thousands of human role-players. The CAI process combines crowdsourcing, pattern discovery, and case-based planning. Content creation is crowdsourced by recording role players online. Browser-based tools allow non-experts anywhere in the world to annotate data, organizing content into a hierarchical narrative structure. Patterns discovered from this meta-data power a novel planning system, which combines plan recognition with case-based planning. The combination of the CAI process and narrative structure produces a new medium for interactively telling stories, which can exploit a massive corpus of content to overcome the tension between freedom and comprehension, supporting NPCs who can interact and converse with humans.

As a proof of concept, this medium is used to automate NPCs who can play roles and interact with human players in a virtual restaurant simulation. However, it is conceivable that the same medium could be useful for other purposes; for example, as an imagination amplification tool for an author writing a novel, to interactively brainstorm potential consequences of character actions. The approach presented here reorganizes the division of labor in content creation, and democratizes the authoring process by including non-programmers in both the data collection and data interpretation processes. At runtime, NPCs compare recent observations to previously recorded games, to select contextually relevant actions and utterances from a database of annotated game logs, containing over 1,000 possible actions, and over 18,000 utterances. This medium has the potential to enable new experiences in videogames, as well as new classes of training simulations, educational software, therapeutic applications, and social robots.

1.1   Challenges in Natural Language Understanding

Some of the earliest videogames were entirely text-based (e.g. Zork, 1980). Games described the world in text, and players typed text to take action in the world. Simple parsers at the time had a minimal vocabulary, and only understood the simplest syntax, but this was a first step toward the ambitious goal of understanding open-ended natural language input. As graphics improved, natural language interaction has taken a back seat, and the state-of-the-art has not advanced beyond the same hand-crafted, multiple choice dialogue trees introduced in the adventure games of the 1980s.

One notable exception to this trend is the experimental interactive drama *Façade* (Mateas & Stern 2005). *Façade* accepts open-ended typed natural language input, and has unique design that allows the game to mask understanding failures in a way that feels natural and contextually appropriate. The player is invited for drinks to the apartment of Grace and Tripp, two self-absorbed NPCs apparently having marital problems. Confronted with this awkward scenario, the player tries to converse with the NPCs, to defuse (or optionally exacerbate) the situation. When the system cannot understand the input, Grace and Tripp proceed to continue arguing, ignoring the player plausibly in the heat of the moment.

Façade's approach to Natural Language Understanding (NLU) is inspiring and extremely well executed, but is not a general solution applicable to all games. The majority of games feature the player as the

main character, who cannot be ignored. The tried and true dialogue trees give players the assurance that everything they say will be understood, at the cost of limited player autonomy, while *Façade*'s approach gives the player freedom, at the risk of understanding failures, leading to reduced agency when the player is unable to affect the story with his or her words. The ideal solution would provide the best of both worlds -- freedom for player expression, with robust understanding of input, to empower the player to use words to take action in the word, fluidly combined with physical interaction.

Unfortunately NLU is difficult problem. In order for NPCs to understand the player, they need to be provided with a vocabulary of words and phrases with adequate coverage of everything players might say. The problem is further complicated if spoken input is desirable, introducing speech recognition inaccuracies, or even complete recognition failure. Finally, even if words are recognized perfectly, and are covered by the vocabulary, there may be ambiguity in interpretation; the NPC needs to understand them in the intended context. NPCs require a comprehensive representation of narrative possibilities that captures anything someone might say, or do physically, in a form that explains when to say or do it. This leads to the authoring problem.

1.2   The Authoring Problem

For any given scenario, there are an enormous number of contextually appropriate things that players could express in natural language, and numerous ways to express the same linguistic act with different words. In addition, a sufficiently open-ended environment might allow thousands of different physical interactions. Current approaches to authoring behavior, based on hand-crafted hierarchical structures (e.g. Behavior Trees, Hierarchical Task Networks), do not scale to representing the combinatorial explosion introduced by seamlessly interleaving sequences of thousands (or more) possible actions and utterances. Yet an NPC who is designed to cooperate and adapt to player choices requires a representation at this scale, in order to recognize what the player is trying to say or do, and take appropriate actions in response. Furthermore, the imagination of a team of designers to anticipate player behavior is finite, and the task of encoding behavior is labor intensive, requiring technical skills.

In some sense, game developers are encountering the same authoring bottleneck that A.I. researchers recognized years ago, as recounted by Roger Schank (1999):

> Not as much has happened in AI as one might have hoped. …. The major reason for this is really one of content. …. We simply were not able to even begin to acquire the content of [human] memory. …. Someone has to get a computer to know what a human knows about using a toaster or playing baseball. …. The real issue was the acquisition of the mundane knowledge that people take for granted.

An ongoing discussion among professional game developers has questioned whether the authoring problem could be addressed with the equivalent of the Photoshop of AI[2] -- an accessible tool, combined with a representationally clean separation between the structure and style of AI content. The intuition

---

[2] Chris Hecker introduced the Photoshop of A.I. in the lecture "Structure vs Style", *Game Developers Conference*, 2009.

being that graphics has benefited handsomely by having the triangle as the core structural building block, which can be stylized by texturing through tools like Photoshop, accessible to non-programmers. Many have concluded that there will never be a Photoshop of AI, because behavior is a complex branching structure, inseparable from the simulation engine on which it runs, requiring the participation of a programmer, or a *procedurally literate* designer.[3] This thesis considers the Photoshop of AI from a different perspective, rethinking where the source material comes from. Rather than hand-crafting behavior and dialogue from a blank page, the authoring process begins by recording thousands of human performances. Though simulation from this content does require a programmer to implement a domain model (see Chapter 5), anyone can record examples of desired behavior, democratizing the creative process, and enabling a content creation pipeline that scales to support efficiently authoring massive amounts of content. From this perspective, a recorded narrative (human performance) is the core structural unit, stylized by blending and sequencing with other recorded narratives. Graphics has dramatically improved by scaling scene rendering from hundreds, to thousands, to millions of triangles. Employing recorded narratives as the core building block, NPC behavior and dialogue, and the interactive stories that emerge from their interactions, can scale in a similar way.

1.3   Interactive Storytelling in the Petabyte Age

The availability of nearly unlimited, affordable data storage and processing has ushered in the Petabyte Age. Big Data is revolutionizing how we approach everything from science (e.g. genome sequencing, drug discovery), to business (e.g. retail analytics, collaborative filtering), to politics (e.g. the Obama campaign's *Narwahl*). Incredible advances in Natural Language Processing (NLP) technologies have come about as a result of Big Data, enabling powerful search engines, automated language translation, and cloud-based speech recognition. These advances in NLP impact our everyday lives at home and at work, but they have yet to play a major role in how we make games, how we tell stories in games, and what stories can be told.

For example, when Microsoft introduced the *Kinect* 3D camera and speech recognition technology in 2009, it was hailed by the press as "absolutely ground breaking and revolutionary technology that ... will change the gaming and AI industry forever."[4] *Kinect* was unveiled with an inspiring demonstration of a virtual boy named *Milo* who players could speak to, and even hand objects by holding them up to the screen. The reality of how *Kinect* has been used in commercial products for NPC interaction has been less ambitious. For example, the blockbuster role-playing game *Mass Effect 3* enables the player to speak to NPCs by reading predefined dialogue options from the screen. Given the challenges detailed in Sections 1.1 and 1.2, it is understandable why progress toward more open-ended interaction has been slow.

This thesis introduces a game called *The Restaurant Game* (TRG) in which NPCs have been authored and automated through a data-driven process, to interact and converse with human players through open-

---

[3] This conclusion is reflected by Michael Mateas in "Revisiting the Photoshop of A.I. Debate", UCSC, February 1, 2012.

[4] theTechnologyBlog.net, June 6, 2009.

ended natural language input. Inspired by the success of the *How May I Help You* project at AT&T (Gorin et al. 1997), where a system was trained to automatically route calls based on data recorded from 10,000 human-human customer service inquiries, TRG was deployed online to record over 10,000 performances of dining in a virtual restaurant, dramatized by anonymous pairs of human customers and waitresses.

The transition from hand-crafted to data-driven interaction impacts how interactive stories are authored, represented, and generated at runtime. Rather than constructing a tree or graph of possibilities, the narrative is represented as a database of recorded gameplay instances, stored with associated meta-data describing recurring patterns of behavior at an abstract level. The human's role in the authoring process involves tagging patterns in the data to specify how fragments of behavior relate to one another, and which fragments are desirable to replay at runtime. At runtime, the AI engine searches for contextually relevant fragments of behavior to play at any given moment, based on observed patterns of behavior. The philosophy behind this data-driven approach is that in order to scale to the density of content required to robustly support open-ended interaction, the future AI engine will look more like a search engine than a finite state machine.

There are two different, yet complementary, revolutions evolving, related to pattern discovery in large data sets, and this thesis explores both in connection with automating NPCs from recorded human performances. First, availability of large data sets increasingly makes statistical recurrence analysis and machine learning algorithms more practical, effective, and applicable for automatically discovering patterns in data. Second, the ease of sharing data with people online has led to new ways to leverage human intelligence in interpreting data through crowdsourcing, outsourcing, and human computation. Both approaches have benefits and drawbacks, evaluated quantitatively and qualitatively when applied to the TRG data set.

1.4   Collective Artificial Intelligence and Narrative Structure

Realizing a new medium for interactive storytelling from composited performances of human role players requires also introducing a new production process, and associated theory of narrative structure. This thesis introduces a production process which applies Collective Artificial Intelligence (CAI). The term *Collective **Artificial** Intelligence* is derived from *Collective Intelligence* -- the idea that a large number of ordinary people can make better decisions than a small number of experts. CAI refers to a similar idea, that a large number of ordinary people can create better models of narrative possibilities than a small number of expert programmers and/or designers. The an end-to-end CAI process for automating role-playing NPCs from recorded human performances relies on a theory of narrative structure for interactive stories illustrated in Figure 1-2.

In the theory of narrative structure associated with CAI, a narrative is composed of a sequence of actions. Each action may be executed by a different actor, indicated by the red and blue outlines (e.g. red is a waitress, and blue is a customer), and may be a physical action or an utterance, indicated by quotation marks. These actions are drawn from a large pool of all possible actions. Because actions are sequenced dynamically, additional structure is required to ensure the delivery of a coherent narrative. Actions are grouped into events (e.g. getting seated, ordering food, paying the bill). Events are grouped into higher level events (e.g. beginning dining, having a meal, concluding dining), forming an event hierarchy. High-level events are grouped into an overarching scenario (e.g. restaurant dining).

Arrows indicate long-term dependencies, which may point forwards or backwards. Forward dependencies indicate causal chains (e.g. the customer ordering pie causes the waitress to bring a pie from the kitchen). Arrows pointing backwards in time indicate references (e.g. a waitress asking about the customer's satisfaction with a steak served earlier).
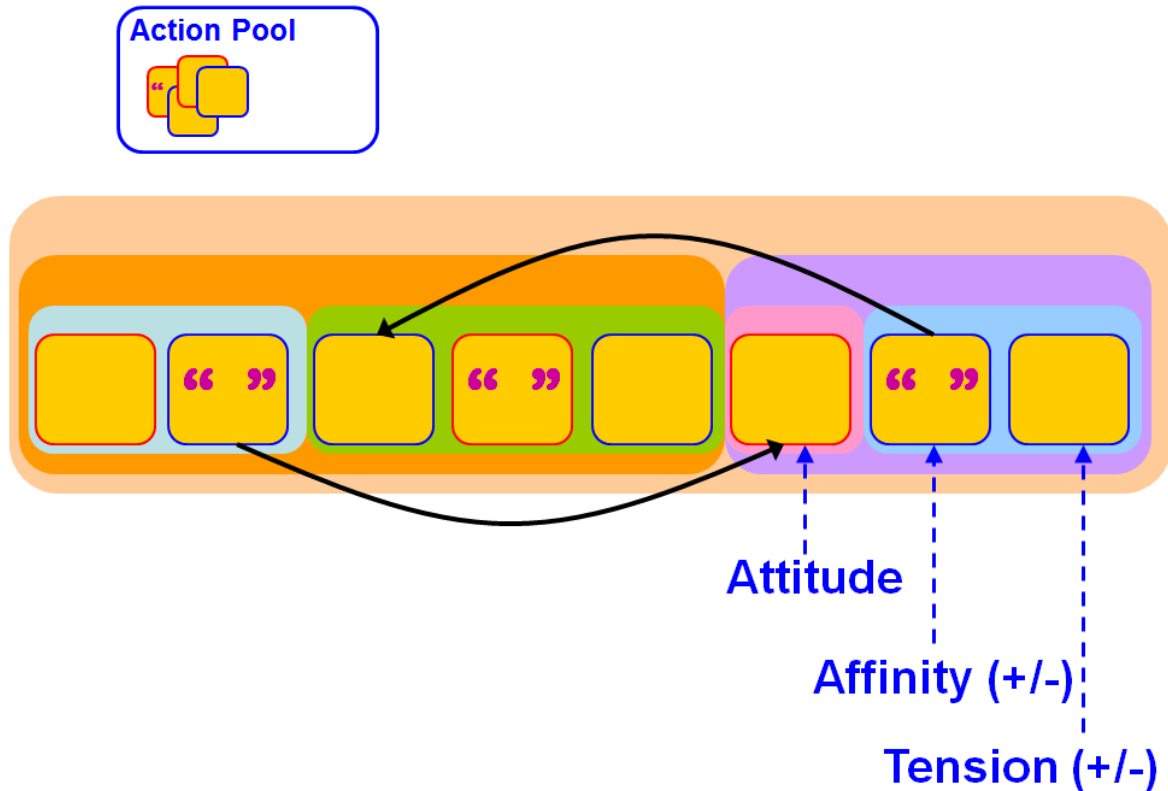
Figure 1-2: The theory of narrative structure for interactive stories, supported by CAI.

Each action may have an associated attitude, affinity modulation, and/or tension modulation. The attitude associated with an action indicates a bias in when to use this action, and how it should be interpreted (e.g. saying something in a polite, rude, or flirtatious way). Affinity modulation affects how the feelings of one actor change with respect to another as the result of some action. Actions can optionally modulate tension, in order to ensure delivery of a compelling narrative, possibly adhering to a desired dramatic arc. The work presented in this thesis implements the core of this theory, including actions, events, dependencies, and attitudes, but does not yet implement affinity or tension.

The CAI process combines crowdsourcing, pattern discovery, and case-based planning, to drive the behavior and dialogue of NPCs who interact with humans in a shared virtual environment, conversing via typed text or speech. The process can be summarized with these three steps:

1) Record many performances of desired behavior.
2) Discover patterns in recorded behavior.
3) Replay fragments of recorded games at runtime.

In this development process, content creation is crowdsourced by recording thousands of human performances, which are mined for patterns representing elements of narrative structure (e.g. events, dependencies). Patterns guide a novel runtime architecture which combines plan recognition with case-based planning to replay fragments of recorded games at runtime. Initial work explored automatically discovering patterns in data. Ultimately, patterns were discovered through a human-machine collaborative process, where human annotated recorded performances with meta-data representing narrative structure, from which patterns were extracted. Browser-based tools allowed hiring people anywhere in the world to annotate transcripts of recorded human performances. The implemented system is called EAT & RUN, where EAT is an acronym for the Event Annotation Tool, and RUN refers to the runtime planning architecture.

Figure 1-3: Interface for human interaction.

Humans interact with this system through an interface that accepts open-ended natural language input (typed or speech), and dynamically generates contextually relevant dialogue options, as semantically similar as possible to the human input (Figure 1-3). The human input may be a complete or partial utterance. Dialogue options are generated by searching for the intersection between utterances in the data that have similar words to the input, and utterances that are contextually appropriate. When speech recognition fails, or words in the input are out of the recognized vocabulary, dialogue options can be generated by context alone. Contextual generation and filtering of dialogue options is made

possible by re-running the same planning system that automates NPCs, from the perspective of the player. An agent associated with the player essentially asks, "If I was an NPC, what would I say?"

1.5   Thesis Overview

This thesis introduces a medium for interactive storytelling, and an underlying narrative structure and CAI process for authoring content in this medium. As a proof of concept, a CAI system has been implemented to generate a simulation from data collected from TRG. TRG has recorded 10,027 performances of a customer interacting with a waitress, which have been tagged by people hired online from the Philippines, India, Pakistan, and the U.S., using browser-based tools. Patterns discovered from these tags power an AI-controlled waitress who can interact and converse with a human customer.

In order to quantitatively evaluate how well the system supports open-ended storytelling, humans interacted with NPC waitresses, using the system to both automate the NPCs, and dynamically generate dialogue options for the players in response to speech input. The results show that the contextually aware data-driven system was able to provide the desired dialogue input 28% more often than a system relying on speech recognition alone. A comparison of player satisfaction with dialogue options generated from the CAI approach versus options generated based on statistical models learned from an unsupervised system show a dramatic improvement. An additional qualitative study conducted with focus groups, comparing TRG to existing games, revealed both strengths of the system, and areas to focus on for future work. Players commented on the unusual amount of freedom given in TRG, and reported the player was able to influence the interaction with open-ended dialogue input. However, without any goals beyond dining at a restaurant, the experience was not as engaging as the other games. In addition, players reported the waitress's personality felt incoherent as her attitude, and even her name, fluctuated as the system replayed fragments of different games. This weakness has been addressed more recently by partitioning the data according to recognizable attitudes (e.g. polite, rude, flirtatious), however this only enforces static attitudes. A system has not yet been implemented to dynamically modulate attitude based on affinity toward other characters, determined by observing interactions, which remains for future work.

CAI is an unfamiliar development process, introducing new challenges for designer control, and debugging. The trade-offs for efficiently integrating massive amounts of content are that tagging errors and inconsistencies may exist throughout the database, and fully testing all possible interaction paths is not possible. Future work will focus on tools for preemptively detecting likely tagging errors, searching the database for similar instances, and seamlessly integrating debugging and authoring tools to facilitate correcting the tags. Additional effort will also be devoted to formalizing the representation of domain knowledge, to be more structured, modular, and reusable, thus decreasing the reliance on programmer intervention. Finally, running the system with additional datasets from other games is required to evaluate generalization beyond restaurant interactions.

The rest of the thesis is organized into the following chapters:

2. Related Work situates this project within the history of related efforts in Natural Language Processing, crowdsourcing, cognitive science, AI, and interactive storytelling.

3. Crowdsourced Content Creation provides details about how data was collected, examples of what the data looks like, and statistics that characterize the TRG data set. Additional data sets are presented, from games re-using the TRG codebase to collect data about human-robot interaction (*Mars Escape*), and dramatic improvisation on the set of a science fiction film set (*Improviso*).

4. Automatic Pattern Discovery reports on efforts to simulate NPC-NPC interactions driven by statistical models learned from an unsupervised system. Approaches explored include learning n-gram models of language and action, classifying dialogue acts with SVMs and HMMs, iteratively clustering events with Affinity Propagation and PLWAP sequence mining, and learning concept labels through correlation.

5. Human-Machine Collaborative Pattern Discovery details the implementation of EAT & RUN, and the online data annotation process. An overview of EAT covers annotation of events, dependencies, attitudes, and domain-specific knowledge, as well as clustering of utterances, and construction of an ontology of concepts. Coverage of RUN includes plan recognition, case-based planning, goals, critics, and domain knowledge. The interface for human interaction is presented, along with transcripts of actual play sessions. This chapter concludes with best practices for applying the CAI process.

6. Evaluation includes a quantitative and qualitative evaluation of an implementation of CAI applied to TRG. The quantitative study explores support for speech interaction, and finds that combining speech recognition with inferred context leads to a system that generates satisfactory dialogue options 28% more often than a system that relies solely on recognized speech. The qualitative study uses focus groups to compare the gameplay experiences of TRG, *Façade*, and *Skyrim*. Focus group discussions reveal that players do find TRG to be more open-ended than other games, and more supportive of using language as action. However, players also commented about the lack of explicit goals in TRG, and issues with coherence of personality, which negatively affected engagement.

7. Conclusion presents contributions, responses to criticism, next steps, and future directions. Contributions include the CAI process, evidence of the value of data-driven interaction, interface for natural language interaction, and data sets from TRG and *Improviso*. Discussion of common criticism of CAI responds to concerns about scaling up, reliance on data collection, crafted versus data-driven experiences, production practicalities, and coherence. Next steps include capturing and generating strong narratives, improving coherence, scaling up, and improving infrastructure. Future work will focus on new data sources, applications to decision support, training, and therapy, and heterogeneous NPC-NPC interactions.

2 Related Work

This work is motivated by a desire for new player-directed experiences in interactive stories, with Non-Player Characters (NPCs) who can communicate with players and adapt to player choices. There is a range of narrative possibilities that are difficult, or impossible, to achieve without the ability to use language as action (e.g. romance, deception, humor). While much progress has been made in the game industry developing cunning opponents with keen spatial awareness, this research works toward socially aware dramatic role players, who can support an experience akin to going online with a dedicated improv troupe.

Two significant developments in games-related research debuted shortly before this thesis research began, and influenced the direction of this work. The interactive drama *Façade* (Mateas & Stern 2005) demonstrated the possibility of delivering a compelling face-to-face social interaction in a mundane everyday environment, combining physical interaction with open-ended natural language. *The ESP Game* (von Ahn & Dabbish 2004) proved that pairing players anonymously online to play a game together was a viable means of collecting large quantities of data capturing general world knowledge, such as labels for images. In some sense, *The Restaurant Game* (TRG) can be considered a combination of these ideas, pairing people anonymously online to engage dramatic role play, generating data about physical and linguistic interaction in an everyday scenario. Recording these human performances is the first step of the Collective AI (CAI) process, leading to many questions about how to leverage this data to understand a human player, and automate an AI-controlled NPC, who adapts to player choices while coherently playing a role, adhering to an overarching narrative. This chapter explores previous work that has influenced the answers to these questions.

2.1 Natural Language Interaction in Interactive Stories

While commercial videogames continue to take a conservative approach to natural language interaction, relying on prescribed multiple choice dialogue trees, a number of research projects have demonstrated the potential for rich, open-ended interaction with language. The motivations for natural language interaction range from increased immersion, to co-operation with NPCs, to building more effective training systems. This section explores how narrative structure can be exploited to help machines understand and generate natural language while interacting with humans.

2.1.1   Natural Language Understanding in Façade

Façade is a fully realized interactive drama which builds on previous work on the *Oz Project* (Bates 1992), combining threads of research related to interactive drama with animated virtual characters (e.g. the *Woggles*), and text-based interactive storytelling (e.g. *Robbery Word*, *Office Politics*, *The Playground*). *Façade* integrates open-ended natural language dialogue interaction into a first-person dramatically charged experience in an immersive virtual environment. The similarities between TRG and *Façade* are obvious, as both place the player in an everyday social scenario, rather than requiring the player to solve a puzzle or complete a mission, and allow interaction through typed text and mouse-driven interaction with the world. The interface for TRG is reminiscent of (and inspired by) *Façade*, however, these projects emphasize different aspects of the experience, leading to different approaches in how language is understood, and used to take action in the narrative.

Façade prioritizes delivering a dramatic experience above other interaction goals, making it particularly well-suited for supporting natural language interaction, despite the associated challenges. TRG is designed to demonstrate a collaborative, player-directed experience, where language can be used effectively as action. The developers of *Façade* ascribe to a philosophy for realizing interactive drama which requires a minimum of three characters, where the player is not the main character. The player is invited to Grace and Tripp's apartment for cocktails, and over the course of about 15 minutes, witnesses the implosion of their marriage, as a heated argument unfolds. The player's typed text input perturbs the rapidly intensifying narrative, encouraging the NPCs toward an amiable, or not-so-amiable, resolution. The dramatic focus and three-character model enables taking a practical approach to the challenge of Natural Language Understanding (NLU), masking understanding failures in a well-executed, plausible, contextually appropriate way -- the NPCs continue arguing, ignoring the player in the heat of the moment, sometimes pausing to look at the player with a confused facial expression. The ability for the player to freely express him/herself at moments that *cry out* for a response, using open-ended natural language input, adds to immersion. However, understanding language is not critical to the primary objective of achieving drama, as Grace and Tripp can often move the narrative forward regardless of player input. In contrast, there are only two characters in TRG, the customer and waitress, and the narrative can only move forward coherently if they understand one another.

Façade's philosophy of interactive drama impacts architectural decisions, and the underlying narrative structure. Narrative is represented as a sequence of dramatic beats (Mateas 2002). Beats are scripted in a reactive planning language called ABL, to choreograph the interactions of Grace and Trip for a short period of time. Each beat has an associated level of tension, and a drama manager is responsible for selecting beats from a bag of beats such that the sequence produces an Aristotelian dramatic arc. One beat is active at a time, and the currently active beat is responsible for interpreting player input. The beat encodes a mapping from expected words or phrases to dialogue acts, and from dialogue acts to NPC responses. In this way, beats provide context for understanding language, however the onus is on the designer to anticipate and encode responses for all possible inputs. This issue is exacerbated by the fact that *Façade* is designed to be replayed at least six or seven times before feeling like the content has been exhausted, requiring authoring several variations of many beats. Michael Mateas (2002) has stated that the authoring burden for implementing *Façade*'s beats was high. Developing the roughly 15 minute *Façade* experience took over five years, and co-developer Stern has estimated that producing a

comparable new drama from the existing technology would take about two person-years[5]. The decision to crowdsource content creation by recording players online in TRG was in part a reaction to this authoring challenge identified by *Façade*. Recording thousands of players playing each role relieves the designer from having to imagine everything someone might say, and when they might say it.

TRG borrows from *Façade*, seamlessly combining an interface that allows typed natural language input at any time with a point-and-click interface for interacting with objects. When interacting with an AI-controlled NPC in TRG, the system responds to human input (typed text or speech) by dynamically populating list of dialogue options, intended to be as semantically similar as possible to the human input. Where *Façade* primarily uses natural language input for purposes of immersion, TRG depends on language as a form of action. The player in TRG is the main character, in a scenario with only two actors, thus dynamically generating dialogue options ensures that the player will be able to effectively use words to move the narrative forward.

2.1.2   Natural Language Understanding in Other Systems

There are a number of previous projects which demonstrate how narrative structure can be leveraged to understand language, and enable using language as action. TRG shares elements of several such projects. The *Leaders* project (Gordon et al. 2004) represents a narrative as a branching storyline, comparable to a choose-your-own-adventure book. Rather than displaying multiple choice options at each branch point, open-ended typed text input is mapped to one of the available branches by a Naive-Bayes classifier (George & Langley 1995). This mapping forms a closed system with open-ended input, where the user never strays from predetermined narrative paths. TRG takes a similar approach of mapping input to known narrative structure, but makes the mapping transparent. Natural language input is used as a query to search for similar utterances observed in the human data, which are then filtered by context, and presented as dialogue options. Transparency ensures the player's input is not misinterpreted, at the cost of immersion (though one could argue that a system that frequently misinterprets input is not very immersive either). The interface can be configured to skip displaying options when the input exactly matches a previously recorded utterance. As the size of the corpus of human performances grows, the likelihood of an exact match increases, thus while the system is closed, there is an aspect of openness that scales with more data.

A rigid, predefined branching structure (e.g. tree, graph) is a simplistic representation of interactive narrative, compared to the fluidity of actual human interaction. Many systems represent narrative more flexibly, as a hierarchical plan, composed of goals can be broken down into sub-goals, eventually reaching leaf nodes representing primitive actions and utterances. While this still leads to representing a narrative as a tree, there may be many possible decompositions of the same task or story, reorganizing modular sub-trees to generate different instantiations. This top-down representation can be reformulated as a grammar of possible decompositions, which can be exploited to infer structure from primitive observations, and predict future observations. For example, if someone enters a restaurant and sits down, one can infer that he is beginning a meal, and might ask for a menu or a drink next.

---

[5] "With this architecture in hand, two trained people could crank out another interactive drama equivalent to Façade in less than a year." http://grandtextauto.org/2008/03/18/ep-86-learning-from-facade/#19

Two previous projects led by members of the Cognitive Machines research group at MIT (where this thesis research was also conducted) have demonstrated how hierarchical plans can be exploited to understand speech input. Peter Gorniak (Gorniak & Roy 2005) recorded pairs of players conversing while solving a puzzle in a role-playing game, and developed a system that could disambiguate the meaning of spoken directives from one player well enough to predict what the other player would do next. Ambiguities could exist at the level of speech recognition, utterance context, or reference resolution. For example, the system could correctly interpret the ambiguous directive "use the blue one," to refer to pulling a specific blue lever in an environment with multiple blue objects, based on the players' current progress solving the puzzle. This was accomplished utilizing a probabilistic parser (Early 1970), supplied with a hand-crafted grammar of all possible ways to solve the puzzle, to perform plan recognition (Kautz & Allen 1986). Plan recognition refers to inferring hierarchical structure (e.g. a hierarchical plan, task, event, or narrative) from an observation sequence. The top-down specification of puzzle solutions informs the bottom up recognition of spoken words. Similarly, Michael Fleischman (Fleischman & Hovy 2006) use a hand-crafted hierarchical task model of an emergency response situation to interpret spoken commands by probabilistically mapping input to valid sub-tasks based on an observation history. The mapping employs a Baysian noisy channel model from the field of Machine Translation (Brown et al. 1993) to estimate the likelihood of an interpreted command given the words returned from an off-the-shelf speech recognizer.

TRG leverages plan recognition based on a hierarchical event dictionary to understand language. This dictionary is comparable to a task model, or plan grammar. The approach taken to construct, and infer hierarchy from, the dictionary is different from that taken in the above projects, but the motivation to exploit context to disambiguate meaning is the same. As part of the CAI process, action sequences representing events are mined from thousands of recorded human performances, providing a more diverse, nuanced grammar or task model than could be crafted by hand. A case-based, rather than probabilistic, approach is taken to inferring a hierarchy from observation sequences. (Case-based techniques are discussed further in Section 2.5). This decision is driven by the social, rather than task-oriented, nature of the TRG data. A restaurant scenario does not work toward a single "win" condition, and may contain extraneous interactions that are not required to complete the over-arching scenario. For example, the customer's decision to stay for dessert rather than immediately getting the bill is arbitrary in TRG, based on the whims of an individual player engaged in dramatic role playing. Thus, as long as the inferred hierarchy can be matched at an abstract level to a recorded human game, it is considered valid. In a social situation, there is not the same notion of the most likely next move, seen in puzzles and regimented military and emergency scenarios.

The idea that narrative context can help machines understand language goes back the idea of *scripts* introduced in the 1970s, proposed as a solution to enable machines to understand written stories. Roger Schank (Schank & Abelson 1977) theorized that humans understand stories and infer missing details by relying on scripts learned from childhood. Scripts consist of roles for people and objects, entry conditions, and a sequence of scenes that capture a chain of events at an abstract level. When we read, "John went to a restaurant. He ordered chicken. He left a large tip", we can infer that he also sat down, looked at a menu, ate his chicken, and was pleased with it. Ultimately, Schank's team found it intractable to encode every possible narrative path through a restaurant scenario. TRG revisits the idea of the restaurant script, trying to learn a rich model of restaurant interaction from thousands of role players, rather than encoding it by hand. At the same time, TRG relies on the existence of a cognitive restaurant script, to enable anonymously pairing people online as customers and waitresses, with minimal training or instructions.

2.1.3   Natural Language Generation in Videogames

TRG leverages narrative structure not only to understand linguistic input, but to generate dialogue options as well. A similar approach has been implemented in the *Disco for Games* (D4G) system (Hanson & Rich 2010), based on collaborative discourse theory (Lochbaum 1998). In this theory, utterances are a form of action, seamlessly interwoven with other types of action, in a hierarchical task model, and a focus stack keeps track of which node is currently being expanded. D4G represents the task model as a hand-crafted Hierarchical Task Network (Erol et al. 1994). D4G does not accept open-ended natural language input, but does allow the player to select an utterance from a list of options at any time, fluidly interleaving physical and linguistic actions. Contextually relevant utterances options are generated dynamically based on possible decompositions of the node of the task hierarchy currently in focus. TRG generates dialogue options by searching for recorded human utterances semantically similar to the human input, then leverages the inferred event hierarchy to filter out contextually inappropriate options. When the human input cannot be understood (due to speech recognition failure, or out of vocabulary words), TRG generates dialogue options purely based on context, similar to D4G. TRG runs the same AI system used to automate NPCs, from the perspective of the player, to generate contextually relevant options.

2.2   Crowdsourced Imagination

The previous section notes that one of the major challenges in developing systems that understand natural language is coverage of possible inputs. This is a symptom of the limited imagination of individual humans, which can be remedied by harnessing the power of the crowd to amplify the imagination of an individual developer, and provide more complete coverage of the input space. The benefits of crowdsourcing are not limited to enhanced understanding of language, as the crowd can also provide common sense knowledge and enhanced creativity, all of which will be explored in this section.

2.2.1   Crowdsourced Natural Language Understanding

Researchers at Bell Labs illustrated the vocabulary coverage problem by bringing human subjects into the lab to apply labels to a collection of images, finding a surprisingly wide variety of words associated with each image (Furnas et al. 1987). Collections ranged from images of common objects (e.g. calculator, nectarine, Lucille Ball), to images of text editing operations (e.g. insert, delete, replace). Across five domains, the study found that in every case two people favored the same term with probability less than 0.2. This issue is evident in the TRG data as well. For example, in a 1,000 game subset, there are five different words (or spellings) used to refer to the bill: bill, check, cheque, tab, and ticket.

The ESP Game (von Ahn & Dabbish 2004) combats the vocabulary coverage problem by enticing the general public to supply labels for images, through the process of playing a game. Players of the web-

based ESP Game are paired anonymously online, and shown a sequence of images. Points are scored each time both players type the same word for an image. Players may not use words that appear on the taboo list for an image, where the taboo list is generated from labels assigned by previous players, thus forcing subsequent players to continue expanding the variety of labels associated with each image. The quality of the labels was evaluated by human subjects, through an image retrieval task, and manual assessment. Human subjects in the manual assessment found that 1.7% of the labels generated by the game were inappropriate for the image, based on inspecting the top six labels for 20 images. The image retrieval task resulted in 100% precision for images retrieved based on 10 labels randomly selected from the corpus. With an observed average rate of 3.89 labels collected per minute by a pair of players, all of the images on Google could be labeled in 31 days by 5,000 players of *The ESP Game* (based on the figure of 425,000,000 images in 2004). While the data produced is valuable for applications such as image retrieval, the players need not be aware of this end-goal. Players are rewarded with free entertainment, a unique social interaction, and the opportunity to advance in online leaderboards.

The ESP Game demonstrates how a game can be employed to crowdsource vocabulary generation. A more immersive project called *Wubble World* (Hewlett et al. 2007) has demonstrated how a 3D virtual world can be employed to learn words for concepts (e.g. colors, shapes) through interaction with human-controlled avatars. However, NPCs in interactive stories need to understand entire utterances, not only individual words, exacerbating the problem of coverage. For example, the same previously mentioned 1,000 game subset from TRG exhibits 167 different ways to ask for the bill, including "I'm ready for my bill," "where's the check?" and even utterances that do not directly reference the bill at all, such as "Can I pay now?" The *How May I Help You* project at AT&T (Gorin et al. 1997) demonstrated that recording many pairs of people conversing in the same situation is a viable way to capture a diverse, yet exploitable sample of different ways to say the same thing. This project recorded 10,000 customer service calls to human operators beginning with the prompt "How may I help you?", and automatically learned to route calls to appropriate departments. Examples of spontaneous speech inputs handled by this system include questions like "Can I reverse the charges on this call?" and "How do I dial direct to Tokyo?" Call routing relied on a system that identified salient phrases in the input, which act as reliable indicators of utterance meaning. Salient phrases are unigrams, bigrams, or trigrams that have high mutual information (Cover & Thomas 1991) with a particular call type. Phrases deemed salient for the same call type could be clustered based on surface similarity, merging phrases like "a wrong number" and "the wrong number."

This notion of salient phrases influenced the data-driven approaches described in Chapter 4 to understand utterances. The success of the *How May I Help You* system, based on 10,000 training examples, was encouraging, as recording 10,000 performances of restaurant interaction was an achievable goal for TRG. However, rather than capturing the response to a single prompt, TRG transcripts encompassed 10 minutes or more of spontaneous role-playing. In order to apply insights from *How May I Help You*, each game was segmented into dialogues, bounded by the physical actions observed before and after uninterrupted sequences of utterances. The posterior action was assumed to give meaning to the dialogue. For instance, the waitress often retrieves a beer from the bar after a dialogue in which the customer orders beer. Phrases observed in some minimum number of game logs were considered salient, and clustered based on surface similarity. The simple dialogue system, described in Section 4.1, generated responses by searching for previously observed utterances with matching sets of salient phrases. In order to train the dialogue act classifier described in Section 4.3, salient phrases were identified based on mutual information between n-grams and dialogue act types. These salient phrases served as linguistic features for the classifier.

2.2.2 Crowdsourced Common Sense

The discussion thus far has focused on leveraging the crowd to understand language, however, the crowd is equally useful for providing common sense knowledge, and possible narratives. TRG records performances of restaurant narratives, from which it is possible to learn common sense about everyday physical and linguistic interactions in a restaurant. There are several notable projects, that have come before and after TRG, with similar goals of acquiring common sense from the crowd.

Long before the term crowdsourcing was coined, *the Open Mind Common Sense* (OMCS) project began collecting common sense assertions on the web (Singh 2002). OMCS allows the general public to contribute assertions such as "The sun is very hot," and "A coat is used for keeping warm." Since 1999, OMCS has collected over a million facts from over 15,000 contributors. A more structured semantic network called *ConceptNet* (Liu & Singh 2004) has been learned from the open-ended English assertions. *ConceptNet* is represented as a directed graph, where nodes are concepts, and edges are assertions that related concepts to one another. Similar to *ConceptNet*, *Cyc* (Lenat 1995) is an ontology that represents common sense knowledge with formal logic, rather than natural language, making authoring knowledge more difficult. These common sense knowledge bases have proven useful for a number of research projects, but without putting the common sense assertions into some context, it is hard for an NPC to leverage this information for purposes of automating interactive storytelling. There have been efforts to learn common sense narratives from the OMCS corpus. One such effort, *LifeNet* (Singh & Williams 2003), produces temporal graphical models drawing inferences through probabilistic belief updating. While this approach is somewhat successful at learning temporal links, due to the noise and diversity of the OMCS data, coverage of any specific scenario is sparse.

More recently, researchers have been able to leverage crowdsourcing platforms like *Amazon's Mechanical Turk* (AMT) to directly request narratives of specific scenarios, paying small amounts of money for each submission. Boyang Li collected step-by-step narratives written in natural language from crowd workers, describing everyday scenarios including fast food dining, and a movie date (Li et al. 2012). Events are not known a priori, and instead generated from the collected narratives through a process combining automatic clustering, based on surface similarity and temporal location, with crowdsourced validation. Once clustered, the system produces a directed graph of possible narratives represented by temporally ordered event sequences, referred to as a script. Compared to TRG, this approach has the advantage of generating scripts without first requiring implementing game for data collection. In addition, the range of possible narratives are not restricted by limitations or assumptions built into a virtual environment. For example, the fast food script includes both dine-in and drive-thru paths. However, this approach generates scripts that are more coarsely grained than the narrative structure associated with CAI. Li's fast food script includes events like "place order" and "pay for food," while the TRG data captures the thousands of variations of action and utterance sequences which comprise these events, providing the fine grained representation required to automate NPCs. There are also limitations related to the coherence of narratives produced from Li's scripts, as the system is not yet capable of detecting mutually exclusive branches (e.g. do not sit at a table after picking up food from the drive-thru). The n-gram driven system described in Section 4.1 suffers from similar issues with global coherence, ultimately addressed by providing higher level structure through approaches described in Chapter 5.

With a focus on improvisational characters rather than narrative, Brian Magerko collects a different kind of common sense data from the crowd (Magerko et al. 2011). Magerko's work shares the interest of this thesis in creating NPCs capable of improvising a scene, along with other human or AI-controlled

characters. However, like Li's work, the scene is not dictated a priori by a preconceived virtual environment. Instead, NPCs interact and reason about observed interactions, in an endeavor to reach cognitive consensus about which scene they are dramatizing. NPCs rely on a synthetic cognitive system designed to model the thought process of a human improv actor, taking actions expected to contribute towards a shared mental model of the scene. In an implementation of a game called *Party Quirks*, NPCs execute pantomime animations (similar to charades) in an effort to help a human guess which character prototype the NPC is portraying (e.g. ninja, cowboy, pirate). The NPC selects actions based on their Degree of Membership (DOM) with attributes associated with the hidden character prototype. Examples of actions include BITES_THINGS, and HIDES_BEHIND_THINGS. Attributes are properties like SPEED, STEALTH, USES_MAGIC, and FEARLESS. Each action may be a member of multiple attributes, with different DOMs, represented as fuzzy ranges of floating point values between 0 and 1. The association of attributes to character prototypes is also represented by a floating point [0…1] value. NPCs can exploit these values to select actions such that they gradually disambiguate the character prototype, prolonging the game for the human partner. Harvesting the values from the crowd for attribute associations and DOM ranges not only decreases the authoring burden, but also captures sociocultural common sense about how different types of characters should behave.

This thesis takes a different approach to improvisation, but shares motivations of Magerko's work to realize NPCs who can dynamically collaborate with other actors without top-down control over the scene. Though the setting for the scene is predetermined in CAI (e.g. a restaurant), the open-endedness of the interaction leads to NPCs faced with ambiguity ("excuse me miss" might begin a new order, or initiate a complaint), requiring the ability to form a shared mental model with other actors, in order to maintain coherent interaction. CAI takes a case-based, rather than probabilistic, approach to disambiguation, as described in Chapter 5, and relies on soft assignments of recognized events, which can be revised as more information becomes available with subsequent observations.

2.2.3   Crowdsourced Creativity

Data collected from the crowd can support generation of narrative representations with ample coverage of language variation and common sense background knowledge, but can also introduce surprises and insights that may not have occurred to an individual or small group of authors. While crowdsourcing content creation introduces the risk of generating unwanted or inappropriate material, there is the potential to amplify the creative ability of an individual or small group of authors. An increasing number of notable experiments with crowdsourced creativity both have emerged, both in and outside of academia. For example, Aaron Koblin's *theSheepMarket.com* hosts 10,000 drawings of sheep created by AMT workers, each paid $.02 to draw a sheep facing left. *LocalMotors.com* crowdsources automobile designs, accepting submissions from anyone. Users vote on designs through monthly contests, and the company manufactures designs that get enough buzz to suggest the viability of selling 500 cars. Jesse Schell's *PuzzleClubhouse.com* hosts an episodic crowd-designed game, where community members submit and vote on game concepts, story ideas, and artwork. A professional development team assembles the crowdsourced content into a playable game. The first installment features puzzles consisting of frogs with back-mounted lasers.

More relevant to this thesis research, Reid Swanson's *Say Anything* project (Swanson & Gordon 2009; 2012) generates text-based narratives interactively, taking turns with a human author, automatically generating subsequent sentences mined from personal blogs. This system takes a simplistic approach to

selecting the next sentence, measuring similarity between the user's last input and candidate sentences based on a vector space model with term frequency-inverse document frequency (TF-IDF) (Jones 1972) feature weights. Later iterations of the system added measures to better ensure coherence, by offering the user multiple options for the next sentence, and ranking candidates to maximize coherence based on Entity-Grid derived features (Barzilay & Lapata 2005). While the original blog authors are unaware that they are participating in this experiment, the data can still be considered crowdsourced, as it is collected online from a large pool of individuals.

Li, Magerko, and Swanson's projects all leverage crowdsourced data that is unconstrained by a predetermined virtual environment, but the data captured is not at the granularity required to automate the moment-to-moment decisions of an NPC. There are likely potential synergies between these projects and CAI, where a first pass of unconstrained crowdsourced content creation could inform the design of a virtual environment used to capture fine-grained recorded performances. This process could be iterative, expanding the repertoire of possible narratives based on observed interactions during data collection. Chapter 5 includes a list of auxiliary events, added to the list of annotated events based on unanticipated, yet recurring observations in the data. This list includes events such as asking to speak to the manager, stealing appliances, and the customer asking the waitress to join him at the table.

2.3   Coherence, Tension, and Affinity

Crowdsourcing content creation offers a potential solution to the authorial burden highlighted by *Façade*, yet combining fragments of narratives from thousands of individual authors introduces new challenges for maintaining coherence, and producing interesting stories. There are several kinds of coherence, all of which need to be addressed. Most important is structural coherence, where the sequence of events in the narrative is plausible, and makes sense to a human audience. As the qualitative evaluation in Chapter 6 demonstrates, assembling narrative fragments into a coherent structure, however, can still result in NPCs that appear schizophrenic, if drawn from performances inconsistent in attitude or factual information (e.g. names, occupations). Note that this is a different kind of schizophrenia than that recognized by Phoebe Sengers (1999), which relates to NPCs oscillating between competing goals. Some of these issues can be addressed by partitioning the corpus into fragments associated with specific attitudes (accomplished in TRG with the Attitude tag described in Chapter 5), and tracking concepts like proper names, for purposes of adapting utterances to conform by variable substitution (not yet implemented in TRG). Several existing narrative systems yield insights into ensuring structural coherence, and modeling affinity to maintain coherent relationships between characters. Finally, a story generation implementation from the past serves as a cautionary tale for systems that ensure coherence, while disregarding tension or interest.

2.3.1   Structural Coherence

In general, interactive narrative systems ensure structural coherence through an architecture that imposes top-down control over possible decompositions of the story. For example, *Façade* represents narrative as a sequence of beats, where each beat consists of a reactive plan that choreographs Grace and Tripp, and monitors player input in anticipation of actions and utterances the NPCs can respond to. Reactive plans are hand-scripted in ABL, enabling expertly crafted dramatic dialogue, at the cost of reduced player agency, when the player's input was not anticipated (or desired) by the script author.

Another top-down approach is to represent the entire narrative as a hierarchical plan, decomposable into modular chunks that can be reused or reordered to support narrative variation. The *Memesis* system (Young et al. 2004) employs a narrative mediation process to balance narrative coherence with player agency. This mediation process can either engage in intervention or accommodation. Intervention prevents an action disruptive to the plan from executing (e.g. a gun files to fire, when aimed at an indispensable character). Accommodation forces the system to re-plan, to allow a goal to be achieved in an alternate way when player actions have interfered with a required plot point (e.g. starting a fight with a character intended to be a key ally later on). Re-planning can be computationally expensive, and requires alternative content, again leading to an authoring bottleneck.

Marc Cavazza describes a system that combines top-down character control with bottom-up emergent storytelling (Cavazza et al. 2002). Autonomous NPCs are driven by hierarchical plans, and interact emergently with other NPCs producing stories. Hand-crafted situated reasoning modules resolve conflicts between actual interactions and an NPC's current plan to maintain coherence (e.g. an NPC may choose to hide from, or converse with, an NPC encountered earlier than expected, impacting future plans). In this system, the player is not an embodied character, but rather a spectator that can manipulate objects in the environment to influence the story.

Like Cavazza's system, CAI combines top-down representation of narrative with bottom up interaction between characters. However, CAI is designed to support open-ended, moment-to-moment, face-to-face interaction between NPCs and human players. CAI does not utilize a centralized narrative planner. Instead, each NPC selects actions and utterances through a bottom-up case-based process, drawing candidates from the corpus of recorded performances found to have similar observation histories to the current interaction (case-based planning is further explored in Section 2.5). A collection of *Critic* processes scrutinize how candidate actions will modify the narrative structure, preventing taking actions that will interfere with global coherence. (Section 2.4 further discusses the narrative structure, illustrated in Figure 1-2). For example, *Critics* might ensure that the waitress seats the customer before taking his order, brings the correct dishes based on his order, and waits to bring the bill until he has finished his last course. One of the *Critics* evaluates the event structure of the narrative, while another critiques long-term dependencies. NPCs employ a *Plan Recognizer* to infer an event hierarchy from observations of new actions and utterances. This event hierarchy conforms to a top-down structure dictated by an event dictionary generated from the corpus of recorded performances. An additional *Critic* biases an NPC to adhere to a specified attitude, delivering a more coherent performance by reducing oscillation between narrative fragments that convey specific attitudes (e.g. polite, rude, drunk).

2.3.2   Coherence of Affinity

Evaluation results in Chapter 6 demonstrate that the CAI approach does deliver a structurally coherent, player-directed interactive experience. However, work remains to address coherence with respect to affinity between characters over time. Partitioning the corpus by attitude is a step toward modeling affinity, but the current implementation lacks a mechanism to dynamically modify the affinity NPCs have toward other characters. Commercial role-playing games (RPGs) typically model affinity simplistically, adjusting a variable per NPC in response to player actions. *Façade* adds an interesting twist, tracking the affinity that the system believes the *player* has toward each NPC. When the typed input is understood, *Façade*'s drama management system determines whether the input is aligning the player more closely with Grace or Tripp, and factors this affinity into the selection of future behaviors for the NPCs.

A recent experimental interactive narrative, *Prom Week* (McCoy et al. 2011), takes a more sophisticated approach to modeling affinity between characters with a system called *Comme il Faut* (CiF). Characters (high school students, preparing for the prom) are connected through three different fully-connected social networks: a relationship network, romance network, and an authenticity network. Nodes in the network are characters, and edges indicate how each character feels toward another, with respect to friendship, romance, and respect. Actions taken by any character result in social network status updates, possibly modifying how others feel toward the acting character. *Prom Week* is played as a third-person simulation (or god game, ala *The Sims*), allowing a disembodied player to select commands by clicking on NPCs. NPCs respond to commands by uttering associated lines of dialogue. Available commands for an NPC at a given time are determined by meeting preconditions of potentially thousands of social status rules, encoded in formal logic. For example, an NPC can only start a fight with someone if they are enemies, and there has been a previous act of provocation. Augmenting TRG with a sophisticated model of affinity like CiF could address issues with social incoherence observed in the qualitative evaluation, such as the waitress agreeing to date the customer after he stole the cash register. An interesting future direction could be an exploration of learning social status rules from a corpus of recorded performances, either automatically or based on human annotation of cause-effect relationships between actions and affinity.

2.3.3   Managing Tension

A story generation system called *Tale-Spin* (Meehan 1976) employed classical planning techniques to dynamically generate narratives based on the goals of characters. While the stories generated were coherent, they were not interesting. Marie-Laure Ryan (1991) uses *Little Red Riding Hood* to illustrate why an interactive narrative system should not strive for coherence alone. In a system like *Tale-Spin*, a goal-directed wolf would simply eat the little girl in the woods, next eat the grandmother, and finally get killed by the hunters.

> From a practical point of view the plan of the wolf is far superior in the flat version than in the actual tale: why should the wolf delay his meal by going first to the grandmother, when he could find immediate satisfaction in the woods? This points to an important difference between the point of view of the wolf, who is a member of the narrative universe and confronts it from the perspective of real life, and the point of view of the author who shapes and contemplates the narrative universe from the perspective of art. The goal of the wolf is to solve a practical problem, the goal of the author is to create a successful story. The tellability of the story is invested in the dramatic encounter of Little Red Riding Hood with the wolf disguised as the grandmother, and in the sudden turn leading from the success of the wolf to his undoing by the guns of the hunters. The narrative climax is not generated by the preceding events; it is rather the preceding events that are generated by the climax. In the authorial perspective, logic operates backwards and there are no accidents. Events are created with a foresight which is radically foreign to the blind progression of pure simulation. While the purpose of simulation is to discover what will happen in a world under specific circumstances, story-generating programs should rather pursue the goal of finding out how a world must be set up, so that certain events can be made to happen.

Façade is designed from the outset to deliver an experience that conforms to a dramatic arc. Beats have associated levels of tension, and are sequenced such that the interaction reaches a climax. Another storytelling system called *Mexica* includes an associated interestingness value with story content (*Mexica* is discussed further in Section 2.5.4). The TRG corpus consists of performances from thousands of people, with varying levels of skill at dramatic improv, who may or may not have delivered compelling performances. No mechanism has been implemented yet in CAI to modulate tension. A promising future direction would enlist human annotators to tag the tension level of notable actions, utterances, or entire events. This meta-data could guide a new *Critic* to bias NPC action selection toward moves that will eventually lead to interactions of the desired level of tension.

2.4 Narrative Structure

The narrative structure associated with CAI (illustrated in Figure 1-2) ensures a coherent interaction, representing narrative as a nested hierarchy of events, composed of physical and linguistic actions, augmented with long-term dependencies. The elements of this narrative structure are motivated by practical considerations, while taking inspiration from theories of narrative, cognitive science, and the philosophy of language.

Definitions of narrative vary widely. Porter Abbott (2002) has written that "as soon as we follow a subject with a verb, there is a good chance we are engaged in narrative discourse," giving a minimalist example of narrative, "I fell down." Gerald Prince (1987) defines narrative as the representation of real or fictional events and situations in a time sequence. In order to operationalize a theory of narrative, and implement an interactive narrative system, a more detailed definition is required that can answer questions about the granularity and composition of events, and the role of causality.

2.4.1 Temporal Action Sequences

At the lowest level, CAI represents narrative as a sequence of actions, with no notion of causality. Elizabeth Bates (1979) observed that humans have a unique ability to learn action sequences, even when there is no understood causal connection. "We accept many cultural activities without questioning how or why they work." She gives an example of a daughter removing a bone from a roast before putting it in the oven, as she had learned growing up watching her mother, only to learn from her mother years later, "in my oven that was the only way I could fit the damn thing in." Bates notes that the ability to pick up large numbers of arbitrary relationships without analyzing or understanding reason for the relationship can be viewed as more rather than less human. She cites evidence that even creatures as simple minded as rats are driven to seek out causality. Rats associate nausea with food stimuli, even when separated up to 24 hours, with lots of stimuli in between.

2.4.2 Interleaving Actions and Words

Action sequences in CAI consist of an arbitrary mix of utterances and actions, fluidly intermixed as a common currency. The idea that language is a form of action, which can affect the world, often interchangeable with physical action, comes from John Austin's ideas (1955), which lead to Speech Act theory, later expanded by John Searle (1969). While other philosophers focused on determining truth conditions of statements, Austin recognized that some language serves a purpose other than to assert truth or falsehood. These speech acts, referred to by Austin as *illocutionary acts*, include asking questions, making promises, and giving directives for others to carry out. The dialogue act classifier described in Section 4.3 includes a Speech Act label as component of the triple that represents an utterance, along with components for content and referent. For example, the utterance "I'll have a pint" can be represented as the triple {DIRECTIVE, BRING, BEER}. Ultimately, the evaluated implementation of CAI relies on semantically clustering utterances by hand, into thousands of folders, to capture fine-grained nuance between meanings hard to capture in a finite set of labels, themselves described in

natural language. For instance, subtle nuance is important when automating NPCs who need to understand the difference between when to say "would you like a drink," versus "would you like to start with a drink." Saying the latter after serving food items would be perceived as a bug.

Treating actions and utterances as a common currency enables integrating social interaction as part of planning, to complete tasks (referred to as events in CAI). Lev Vygotsky (1978) observed that humans learn at an early age to use language as a tool to manipulate their environment, providing a new form of building blocks around which behavior in reorganized in the mind.

> The most significant moment in the course of intellectual development, which gives birth to the purely human forms of practical and abstract intelligence, occurs when speech and practical activity, two previously completely independent lines of development, converge. As soon as speech and the use of signs are incorporated into any action, the action becomes transformed and organized along entirely new lines. The specifically human use of tools is thus realized, going beyond the more limited use of tools possible among the higher animals.

Language makes it possible to use other humans as tools, in tasks that are easier with, or impossible without, collaboration. In the case of TRG, there are tasks that may be physically possible to accomplish alone, but require collaboration due to established social conventions. For example, while a customer is capable of retrieving food from the kitchen himself, he will typically place an order, to be fulfilled by the waitress, adhering to the rules of socially acceptable behavior in a restaurant (assuming a sit-down restaurant, rather than fast food).

2.4.3   Language Games and Events

The plan recognition process in CAI infers an event hierarchy based on the observed sequence of actions and utterances (as described in Chapter 5). Utterances are given meaning by the environment of actions and utterances that surround them, and give them context. This is a form of scaffolding, like that theorized by Jerome Bruner (1977). Bruner describes the process of early language acquisition through participating in social interaction games like peek-a-boo. Through repeated interaction, an infant learns the social situation, which acts as scaffolding for language. Over time, the child learns to predict actions, and is able to swap roles with the caregiver, and eventually substitute redundant language for physical action.

Each type of event in CAI can be treated as a social interaction game, like peek-a-boo, with rules that emerge by inspecting hundreds (or thousands) of occurrences of the event. The event dictionary, generated from human annotations of events (described in Chapter 5), captures the rules for games like ordering food, paying the bill, as well as games like flirting, and exchanging pleasantries (e.g. "How is your day going?" "Not bad, yours?"). Ludwig Wittgenstein (1965) discusses isolating patterns of linguistic interaction into language games for specific purposes, using context to grounding the meaning of utterances. Wittgenstein finds that it is infeasible to try to discretely define the meaning of a word or utterance, using a formal representation like logic, or by using language itself to construct a definition.

> We find that what connects all the cases of comparing is a vast number of overlapping similarities, and as soon as we see this, we feel no longer compelled to say that there must be some one feature in common to them all. What ties the ship to the wharf is a rope, and the rope consists of fibers, but it does not get its strength from any fiber which runs through it from one end to the other, but from the fact that there is a vast number of fibers overlapping.

44

Illustrating Wittgenstein's point (as well as Furnas' observation from Section 2.2), events in TRG can be represented by thousands of different, unique sequences of actions and utterances. Given any pair of sequences representing the same event, there may be no overlap. In addition, words and phrases observed as building blocks for one event may be used for completely different purposes in other events. Marvin Minsky (1974) agrees with Wittgenstein, using the example of trying to represent a concrete definition for what a chair is. There are always exceptions -- unique designs that have only one leg, or tiny toy chairs. Minsky proposes representing concepts as similarity networks, similar to Wittgenstein's metaphor of a fibrous rope, with a vast number of overlaps but no single defining thread.

> We do not want to be committed to an inflexible, inclusion-oriented classification of knowledge.... The "crisscross resemblances" of Wittgenstein are then consequences of the local connections in our similarity network, which are surely adequate to explain how we can feel as though we know what is a chair or a game–yet cannot always define it in a "logical" way as an element in some class-hierarchy or by any other kind of compact, formal, declarative rule.

The event dictionary in CAI looks more like a grammar than a similarity network, but serves the same purpose of defining events with a loose collection of exemplar instances, rather than a single formal definition. This instance-based approach can be considered case-based, as discussed in the next section, and allows an NPC to recognize the same event expressed a wide variety of ways, scalable as the corpus grows. Inducing an event dictionary automatically from the TRG corpus is difficult for a number of reasons -- many actions and utterances have ambiguous event membership, some actions and utterances do not contribute to any event, and multiple players are interacting at once, often engaging in multiple events at the same time. For these reasons, the evaluated implementation of CAI relies on human annotation of events.

The success of the human-machine collaborative pattern discovery process for the TRG corpus (described in Chapter 5), demonstrates that humans have a shared common sense about what goes on in a restaurant, making the task of event discovery easy for humans, despite the aforementioned complications which make the task difficult for machines. Annotators may be benefitting from humans' natural inclination to parse intentional behavior. Dare Baldwin (2001) has shown that even 10-month-old infants recognize the initiating and concluding boundaries of intentional events. Infants shown videos of everyday activities, such as a woman picking up a towel from a kitchen floor, were found to look significantly longer when motion was suspended (by pausing the video) interrupting an event, rather than at points where an intention was fulfilled. In both cases, motion is jarringly suspended, and the longer looking time indicates greater renewed interest in the more surprisingly placed pause.

2.4.4   Causality and Structure

All of this focus on events, and context, is not to say that causality can be ignored. There are certainly instances where numerous structurally sound choices exist for the next utterance or action in a story, with respect to the event hierarchy, but few are valid with respect to fulfilling long-term dependencies. Causal chains are an integral part of the representation of Schank's scripts. Schank's work was intended to help machines understand stories, from incomplete accounts of events that a story teller deemed worthy of telling. If someone tells a story about getting served a rare burger, and leaving a small tip, the system can rely on known causal chains between events in the restaurant script to deduce that dissatisfaction with the burger caused the small tip. Schank's system leverages a mechanism called a script applier to fill in the finer-

grained details surrounding the courser grained events. In this case, the script applier might fill in details related to placing an order for a medium burger, prior to receiving a rare burger.

An interactive storytelling system works in reverse from Schank's story understanding system. NPCs execute a sequence of fine-grained actions, and base future decisions on the past interaction history. Placing an order begins a causal chain, setting up an expectation to receive the ordered item. When the order is not fulfilled as anticipated, this might setup another causal chain leading to a small tip. Chapter 5 describes two *Critic* processes implemented in CAI to validate forward and backward dependencies. Validating a backward dependency prevents an NPC about asking about a steak that was never served. Validating a forward dependency ensures the NPC serves the item that was actually ordered. Dependencies are established through a case-based process of matching the observed event history to game logs with similar histories in the TRG corpus, where dependencies have been annotated by humans.

2.5   Case-Based Planning

Schank's team found the task of encoding scripts by hand, accounting for everything that could possibly happen in a restaurant, to be intractable. This experience led Schank to begin thinking about how knowledge is acquired, stored, and reorganized in the mind, whether digital or human. Schank theorized that most real-world problem solving is based on recalling and adapting past experiences, rather than rationalizing from first principles (Schank 1983). Thus, much of the AI research at the time was misguided.

> What we should absolutely not do is assume that when people solve problems, they do so using rational, explicit knowledge known consciously by them. Real-life problem-solving behavior is almost never original. Almost all solutions to novel problems are adaptations of previous solutions known to the problem solver. These solutions, the knowledge of those cases, and the procedure by which those cases are found, is entirely non-conscious. Nearly all human problem solving uses knowledge we are not aware we have. We internalize these cases and suppress the conscious reminding. Artificial Intelligence has been severely set back by the illusion of the primacy of rational knowledge. This manifests itself in the focus on what I call "artificial problem solving". There has been a focus in AI on cryptarithmetic, the Tower of Hanoi problem, the missionaries and cannibals problem, chess, go, and theorem proving. These problems are rather artificial. Human intelligence evolved over millions of years without ever being tested on them. The kinds of problems that human intelligence evolved to handle necessarily relied upon the use of prior experience.

Enabling machines to reason from past experience required a radical detour, changing direction from the AI field's focus on semantic representations of knowledge, to episodic representations. Systems needed to be designed to solve problems by being *reminded* of similar situations from the past. As a result, Schank's theory of scripts evolved to explain the relationship between generalizable knowledge structures representing stereotypical interactions, and specific memories capturing the nuance and variety of actual human interaction.

I am arguing that a script is a collection of specific memories organized around common points. The restaurant script must actually contain particular memories, such as the experience in Legal Seafood -- where you pay before food arrives. I do not believe in the script as a kind of semantic memory data structure, apart from living, breathing, episodic memories. Elements of episodes that are identical are treated as a unit, a script. Subsequent episodes that differ partially from the script are attached to the part of a script they relate to. Differing parts of the episode are stored in terms of their differences from the script. Such episodes can be found when similar differences are encountered during processing.

Many of Schank's ideas from the 1980s are even more relevant today, in a world where it is increasingly easy to record human behavior and dialogue on the web, on mobile devices, and in video games. Large corpora of recorded human interaction serve as a collective, synthetic episodic memory, which can be leveraged by machines to understand behavior, and plan actions to execute in the world. It is almost surprising how well Schank's thoughts about human language and cognition foreshadow the data-driven approach taken by CAI.

Conversations are really a series of remindings of already processed stories. The mind can be considered a collection of stories or experiences one has already had. It is almost as if we never say anything new. We just find what we have already said and say it again. The story-based conception of talking presupposes that everything you might ever want to say has already been thought up. This is not as strange as it seems. An adult has views of the world that are expressed by ideas he has thought up already and has probably expressed many times.

2.5.1 Case-Based Reasoning and Planning

Schank's students formalized this theory of reminding into an implementable approach to AI called *Case-Based Reasoning* (CBR) (Kolodner 1983), later applied to plan formulation as *Case-Based Planning* (CBP) (Hammond 1990). CBR / CBP consists of four steps:

(1) Case retrieval:
Retrieve cases from a corpus which are similar to the problem one is trying to solve.

(2) Ballpark solution proposal:
Propose one or more cases that could be used to solve the problem.

(3) Adaptation:
Adapt variables of the proposed case(s) to align the problem.

(4) Critique:
Scrutinize the adapted case(s) to validate whether a solution has been found.

This process is made concrete with a simple example of planning waitress behavior in TRG, following the planning process described as part of CAI in Chapter 5. Suppose an NPC waitress needs to respond to a customer, who has just said, "Can I get a glass of water to start?" The waitress clusters this utterance with others believed to be semantically similar, and infers that the customer is beginning an ORDER event. The waitress retrieves cases (in the form of game logs) from the TRG corpus which include an ORDER event that begins with an utterance believed to be functionally similar to the customer's input. She then proposes candidate game logs that extend or respond to this ordering event. In some of these

proposals, the customer orders water, and in others he orders something else to drink (e.g. beer, wine, coffee). All proposals are adapted such that the customer orders water, and are passed to a set of *Critic* processes to scrutinize. The process completes once any proposal is validated by all *Critics*. Imagine that the first proposal specifies that the waitress should respond by asking, "Can I see some I.D.?" One of the *Critics* will reject this proposal based on meta-data in the game log that indicates a dependency exists between this utterance, and a previous utterance that refers to beer. The next proposal specifies that the waitress should say "Sure, that will be right out", which is validated by all of the *Critics*, and vetted as the next action for the waitress to execute. There have been 11 *Critics* implemented so far, which scrutinize proposals from different perspectives, covered in more detail in Chapter 5.

CBR has proven useful for problems like medical diagnosis, where doctors desire explanations for symptoms, but the set of possible explanations cannot be easily enumerated for classification. Kristian Hammond demonstrated extending CBR to planning with a CBP system called Chef, which operated in the domain of Szechwan Chinese cooking. Based on a user's goals for different tastes, textures, ingredients, and type of dish, Chef would formulate a plan specifying the steps for a new recipe. Prior to the introduction of CBP, planning systems pursuing multiple goals would formulate plans to satisfy each goal independently, then attempt to merge the plans into one. This naive approach can lead to problems when steps from one plan conflict with, or undo the progress of, another plan. A CBP system can retrieve plans from the past that satisfy as many of the existing goals as possible, rather than planning from scratch, and then adapt the proposed plan(s) to try to satisfy remaining goals. In the TRG corpus of human-human interactions, it is not uncommon to find fragments of game logs where players are engaged in multiple events at once -- for instance, simultaneously clearing a table and taking a dessert order, while the customer drinks. In addition to the ability to formulate plans to satisfy multiple goals at once, Janet Kolodner has commented on the practical benefits of a more intuitive content authoring process: "While it is hard to get experts to tell you all the knowledge they use to solve problems, it is easy to get them to recount their war stories."

The rest of this chapter highlights a number of previous CBR and CBP systems, implemented for purposes of simulating social interaction and common sense reasoning, playing videogames, and generating stories. Aspects of these systems overlap with CAI – some systems interact with humans in real-time, some generate natural language, some extract cases from recorded human player behavior. Synthesizing these ideas leads to a new medium for telling stories in everyday social settings, face-to-face with embodied simulated role-playing characters, based on compositing recorded human performances. One factor which sets CAI apart from earlier systems is the scale of the case base -- comprised of 1,000 human performances, rather than tens or hundreds -- which impacts architectural and implementation decisions.

2.5.2   Case-Based Common Sense Reasoning

Push Singh's *Em-One* (2005) uses a corpus of narratives to guide the physical and linguistic interaction of two NPCs in a virtual environment, who are collaborating in an everyday scenario of assembling a piece of furniture together. At a high level, the approach to reasoning in CAI is quite similar to *Em-One*. Both perceive observations from a virtual environment, retrieve cases to respond to observations, and critique proposed cases with *Critic* processes to select a coherent next action. The desire to robustly support human-NPC interaction in CAI, as well as NPC-NPC interaction, demands orders of magnitude

more content (more cases) than existed in *Em-One*, leading to significant differences in authoring and implementation.

Cases in *Em-One* are narratives represented as hand-crafted frames (Minsky 1974), where a frame is a data structure representing a stereotypical situation, similar to Schank's scripts. Frames have slots that can be filled by other frames, forming a flexible, hierarchical representation. Recording human performances in CAI facilitates generating an enormous case base. Though these recorded game logs require additional processing to capture the semantics of each interaction, Chapter 5 demonstrates how non-experts can efficiently generate this meta-data, while Chapter 4 hints at possibilities to automate such processes in the future. In addition, human performances capture the nuances of temporal interleaving of behavior that cannot be easily represented in a more rigid hierarchical frame.

Critics in *Em-One* are organized into three layers: reactive, deliberative, and reflective. Reactive *Critics* search for narratives which propose actions to respond to observations. Deliberative *Critics* critique proposed actions, validating causal and goal-oriented constraints. Reflective *Critics* identify and circumvent problems observed in traces of decisions from the reactive and deliberative layers, such as an NPC caught in a cycle of repeatedly making the same mistake. While this three-layered architecture is conceptually clean, CAI combines the deliberative and reflective layers for practical reasons, instead sorting *Critics* to minimize processing of proposals. When iterating over thousands of proposals, it is beneficial to quickly discard candidates without first computing structural and causal relationships. For example, if the game engine has already reported that an NPC has failed to pick up an object (perhaps because someone else picked it up), a reflective *Critic* should reject proposals to try to pick the object up again, before executing any more *Critics*. Reactive *Critics* are replaced by interaction goals (described in Chapter 5), allowing an NPC to not only propose actions which respond to the most recent observation, but to also consider actions which may complete an earlier established causal chain, or move the narrative forward toward subsequent events.

2.5.3   Case-Based Planning in Games

Successful applications of CBP to simulation and strategy games in the past have provided valuable insights for implementing CAI. However, automating NPCs who can collaboratively play roles in stories using linguistic and physical actions is a different problem, requiring a different approach to indexing cases. In addition, exploiting a corpus of thousands of cases enables simpler alternatives to online learning and adaptation proposed in the past.

The approach to authoring cases in CAI is most similar to that taken for the *Darmok* system (Ontanon et al. 2007), which employs CBP to automate an opponent in the Real-Time Strategy game (RTS) *Wargus*. Authoring begins by recording traces of human players (though players are competing, rather than cooperating). Gameplay traces are then annotated by humans to indicate the goals that each action contributes to. A goal may span any number of actions, and each action may contribute to multiple goals, at different levels of abstraction, forming a hierarchy. For example, attacking an enemy tower may contribute to a DESTROY_TOWER goal, which is a subgoal of WIN_GAME. The way these goal annotations are used to extract cases differs from CAI, due to differences between RTS gameplay and simulated social role-play.

Cases in *Darmok* are extracted from the annotated goals, where each case represents a goal, including the action sequence or subgoals taken to accomplish the goal, and a retrieval index indicating where the

case is relevant. The index is represented as the state of the world, expressed as a vector of unit counts on the map when the goal was initiated in the trace. In contrast, CAI stores each entire recorded performance as a case. Rather than segmenting performances to extract events as cases, each event instance acts as an index for retrieving a case, leading to many indexes for each performance. In addition, performances can be retrieved by additional indexes, generated from supplemental layers of meta-data indicating dependencies, attitudes, and domain-specific information. Schank (1995) has commented on the importance of indexing on thinking:

> Thinking involves indexing. The more information we are provided with about a situation, the more places we can attach it to in memory. A story is useful because it comes with many indices. Indices may be locations, attitudes, quandaries, decisions, conclusions, or whatever. The more indices we have for a story, the more places it can reside in memory.

Preserving entire recorded performances allows *Critic* processes to exploit information at any point in the interaction to ensure coherence. This is particularly important in a system that understands and generates natural language, because an utterance may refer to something that happened long in the past, or establish a causal chain, to be completed long in the future. Planning actions from an uncut recording also allows an NPC to continue replaying from the same game log indefinitely, moving fluidly from one event to another, or meandering between temporally overlapping events in a plausible, yet spontaneous way.

Darmok has the luxury of indexing cases by a complete yet compact representation of the state of the world, due to the relatively small variety of unit types. The state of the world in TRG is more difficult to explicitly define, because of the open-ended nature of the game. State depends on things that were said by either player, actions taken, how objects were interacted with, as well as the current position and orientation of objects. Indexing cases by events is an incomplete, yet practical means of retrieving proposals expected to be in the ballpark of what an NPC should do next. With regard to indexing with incomplete information, CAI is more similar to a CBP system called *Mayor* (Fasciano 1996), than *Darmok*.

Mayor is a system developed to play the simulation game *SimCity*. The state of the world in *SimCity* is complex, comprised of many variables which capture the economic, ecological, and social well-being of a sprawling city. Each case is indexed by a subset of variables relevant to effects of executing a particular case, allowing the CBP system to retrieve proposals within the ballpark of reasonable things to do next, given some limited view of the world. Due to reasoning with incomplete information, some plans may fail. As a hypothetical example, perhaps adding roads is expected to increase trade and boost the economy, but leads to increased pollution that decreases property values. *Mayor* is equipped with mechanisms to learn by diagnosing the cause of failure, employing a dependency network indicating how modulating one aspect of the simulation will impact other aspects. For example, increasing traffic will increase pollution, while increasing law enforcement will decrease crime. Once the cause of failure is identified, a case can be modified to avoid repeating the same mistake in the future. The idea of a system that can learn from its own mistakes and adapt is theoretically elegant. In practice, identifying the cause of failure, and deducing the correct adaptation, can be difficult, and may require a lot of domain knowledge (like *Mayor*'s hand-crafted dependency network). For these reasons, CAI does not attempt to learn from failure, and instead takes advantage of its large corpus to replan by simply retrieving an alternate case when a plan fails. Reflective *Critics* mentioned earlier avoid repeating the same mistake cyclically. This approach suggests a trade-off between corpus size and system complexity, where if there is enough data available, sophisticated learning and adaptation may not be necessary.

2.5.4   Case-Based Planning in Storytelling

Mateas (2003) proposed CBP as a potential solution to the content authoring burden experienced developing *Façade*. CBP has been applied to (non-interactive) story generation with varying degrees of success, and more recently applied to interactive storytelling. While these implementations rely on a corpus of hand-crafted narratives, the approaches to CBP bear resemblance to that implemented for CAI, while also including inspiring features worth consideration as extensions to CAI in the future.

Minstrel (Turner 1994) was an effort to model human creativity, which generated stories significantly different from any in the system's memory, while trying to fit the schema for some specified a theme. Generation employs a CBP process, adapting episodes from a knowledge base of existing stories in the *King Arthur* domain. For example, *Minstrel* can adapt an episode about a princess drinking a potion to injure herself for a new story that requires a knight to kill himself. The adaptation process relies on hand-crafted transformation methods, and can produce stories that humans may find odd (e.g. the knight eats a princess), or nonsensical. Noah Wardrip-Fruin (2009) concludes that the problem lies in simulating one part of human intelligence, creativity, but not the rest, providing no means to reject inappropriate stories. This problem becomes more severe as the size of the corpus grows – the system becomes more creative, but has more opportunities for strange adaptations. One of the goals for this thesis is to demonstrate how an interactive narrative system can scale up to robustly composite thousands of narratives. *Minstrel* could benefit from including a final critique phase after adapting narratives, however these *Critics* would require domain knowledge about what humans consider normal in the *King Arthur* domain. Reiterating the point made in regards to adaptation and domain knowledge in *Mayor*, there is a tradeoff between corpus size and the complexity of adaptation. While TRG does perform some minor adaptations (e.g. swapping associated food items), adaptation is minimized, instead relying on a large corpus to find alternative plans.

Like *Minstrel*, *Mexica* (Pérez y Pérez & Sharples 2001) models the human authoring process, implemented as an *engagement-reflection* cycle, which could be described as CBP. Engagement begins with an initial action, repeatedly appended with additional actions drawn from a corpus of human-authored stories, represented as action sequences. Reflection breaks impasses, verifies coherence, and ensures interesting stories based on meta-data about the dramatic tension at each moment in the supplied stories. *Mexica* also has a mechanism called a story-world context to track variables like affinity between characters to further ensure coherence. An example given of how story-world context could influence the direction of the story describes how a princess curing a knight's injuries might lead to the knight rewarding the princess. One of differences between a text-based story, and storytelling with embodied characters in a virtual environment, is that the text-based story can pursue multiple narrative paths simultaneously, which might intersect in the future. *Mexica*'s story-world context keeps track of which events characters are aware of, influencing actions selected in the future. For instance, if the princess was not present when a farmer attempted to kill the knight at the market, her affinity toward the farmer will not change. More sophisticated modeling of affinity has been previously mentioned, in association with *Prom Week*. Simulating multiple simultaneous narrative paths adds another dimension to modeling coherent affinity, and suggests another potential way to scale up CAI in the future, opening new possibilities for interaction.

The *Riu* system (Zhu & Ontanon 2010) is another text-based storytelling system that employs CBP, differing from *Minstrel* and *Mexica* in generating *interactive* stories. Like the previously mentioned systems, *Riu* relies on a case-based system to draw analogies between the story being generated, and those in a corpus of narratives. These analogies lead to continuations of the current story, as well as

interaction options for the player. The system can leverage the corpus to imagine the consequences of player actions, and interestingly these imaginations may lead the system to prevent taking an action, giving an explanation to the player. For example, a player's choice to play with a stray cat may lead the system to extend the story with the player's inner thoughts, communicating that the player decided not to play with cat because it brought back sad memories of a pet lost in the past. In a sense, *Riu* combines the generation of options from the *Leaders* project with the narrative intervention of *Memesis*, exploiting CBR to explain the intervention. Integrating an inner-voice like this into CAI could add a unique twist to embodied face-to-face interaction in a virtual world in the future, filling the player in on a backstory through in-game choices, or acting as commentary for a training simulation to facilitate understanding within a lesson.

3 Crowdsourced Content Creation

Porter Abbot (2002) has described how Leo Tolstoy was allegedly surprised by his own characters:

> Tolstoy records that shortly after Vronsky made love to Anna Karenina, to the author's great surprise his character began preparing to kill himself. Tolstoy recalled writing on feverishly to find out how the scene would end.

One might say that Tolstoy had an unusually active imagination. This chapter describes a new approach to content creation, which augments the imagination of an individual author, such that everyone can share Tolstoy's experience of being surprised by their own characters. The Collective Artificial Intelligence (CAI) process begins by recording thousands of performances of online dramatic role-play between humans online. Recording spontaneous, anonymous, improvisational performances online produces a data set that covers a space of narrative possibilities that it is unlikely any single human author, or even team of human designers, could craft by hand, due to the limits of imagination. Compositing this data into an interactive narrative medium, which can be leveraged to generate possible next actions and utterances given an observation history, could enable a variety of new ways to tell stories -- for example, amplifying the imagination of an individual author, or interactively generating a story that adapts to an embodied human role player in a virtual world. This research focuses on the later, generating dialogue and behavior for NPCs who play roles in stories, while adapting to the behavior of a human player.

A common code base provided the foundation for developing and deploying three different online games, yielding three datasets, each exploring different types of interaction. *The Restaurant Game* collects data about interaction in an everyday scenario, Mars Escape records human-robot interactions in a puzzle scenario, and players of *Improviso* dramatize an alien encounter on the set of a low-budget science fiction film. Generation from *The Restaurant Game* data is the primary focus of the thesis, while re-using the same system to generate from the other data sets remains for future work. This chapter describes these games, and shares lessons learned from data collection process.

Figure 3-1: Screenshot from *The Restaurant Game* (from the waitress's perspective).

3.1 The Restaurant Game

The Restaurant Game (TRG) (Orkin 2007) anonymously pairs people online to role-play as a customer and a waitress in a virtual restaurant. Players can type chat text to each other, navigate the 3D environment from a first-person perspective, and interact with 47 types of objects through a point-and-click interface. Minimal instructions are given to the players – the customer is told to have dinner, and the waitress is told to earn money, with the expectation that players will bring into the game world their socio-cultural knowledge of what goes on real restaurants.

The Restaurant Game

OBJECTIVE:

YOU ARE A WAITRESS AT A RESTAURANT. YOUR OBJECTIVE IS TO WAIT
FOR THE NEXT CUSTOMER TO ARRIVE, AND THEN EARN AS MUCH MONEY
AS YOU CAN. WHEN YOU DEPOSIT A CHECK IN THE REGISTER, YOU WILL
EARN 50% OF THE PRE-TAX TOTAL PLUS THE TIP.

OBJECTIVE COMPLETE WHEN YOU HAVE COMPLETED YOUR OBJECTIVE,
EXIT THE RESTAURANT, AND WALK TO THE
'OBJECTIVE COMPLETE' SQUARE ON THE SIDEWALK.

AFTER FILLING OUT A BRIEF SURVEY AT THE END OF THE SESSION, YOU
WILL BE PRESENTED WITH A PERSONALITY PROFILE BASED ON HOW YOU
CHOSE TO PLAY THE GAME.

BE YOURSELF, AND HAVE FUN !!

CLICK HERE TO START THE GAME

Figure 3-2: Objectives given to Waitress.

The Restaurant Game

OBJECTIVE:

YOU ARE A CUSTOMER AT A RESTAURANT.
YOU HAD LUNCH AT NOON, AND IT IS NOW 7PM.
YOUR OBJECTIVE IS TO HAVE DINNER AT THE RESTAURANT.

OBJECTIVE COMPLETE WHEN YOU HAVE COMPLETED YOUR OBJECTIVE,
EXIT THE RESTAURANT, AND WALK TO THE
'OBJECTIVE COMPLETE' SQUARE ON THE SIDEWALK.

AFTER FILLING OUT A BRIEF SURVEY AT THE END OF THE SESSION, YOU
WILL BE PRESENTED WITH A PERSONALITY PROFILE BASED ON HOW YOU
CHOSE TO PLAY THE GAME.

BE YOURSELF, AND HAVE FUN !!

CLICK HERE TO START THE GAME

Figure 3-3: Objectives given to Customer.

While the goal of the research is to automate NPCs who can interact with humans in a plausible restaurant interaction, the game is intentionally open-ended, allowing a wide range of improvisation and misbehavior. All objects offer the same set of interaction options: Goto, PickUp, PutDown, Give, LookAt, SitOn, Eat, Use. Thus, players may choose to sit on chairs and eat steak, or alternatively, sit on steak and eat chairs. Objects react to actions in different ways; food diminishes from the plate bite by bite, while eating chairs results in only a chomping sound effect.

Figure 3-4: User interface for object interaction.

Surprisingly, about 75% of players demonstrated mostly ordinary restaurant behavior, while the other 25% engaged in bizarre behavior, such as stacking cherry cheesecakes into a staircase, and climbing onto the roof of the restaurant. (Percentages are based on human flagging of "junk" data, described in Chapter 5). Capturing atypical behavior is desirable, as it provides examples of how an NPC should respond to unexpected events, and ideally leads to an experience that is more robust and open-ended than current commercial games provide, supporting players who try to find the boundaries of the possibility space. Famed game designer Will Wright (*SimCity*, *The Sims;* Maxis 1989; 2000) has commented that supporting player misbehavior and exploration is part of good game design. This thesis demonstrates how crowdsourcing the content creation process can capture a wide variety of atypical behavior.

Players obviously enjoy being subversive to some degree. And so we want to provide that and let them know that we're on their side in that way […]. It's just another form of a player taking control. I think for most people, their kind of general aesthetic with games is that the more I control this experience, the better the game is […] in terms of [finding that] I can choose to go off into an interesting path, and the game will support that path. This animal we're calling subversion is really just empowering the players to not hit walls as often […]. At a fundamental level it's kind of convergent with what I would call "good game aesthetics."

- Will Wright

Figure 3-5: Screenshot of stacking cherry cheesecakes and climbing on the roof, emailed by a player.

3.1.1 Implementation and Data Logging

TRG was built with the Torque Game Engine v3.1 from GarageGames. Art assets (3D models, textures, sound effects) were imported from *The Sims 2* (Maxis 2004) using *SimPE* (Ambertation 2004). Characters were animated with *MilkShape3D* (Chumbalum Soft 1996), and the restaurant's architectural structure was modeled in *QuARK* (the Quake Army Knife) (2001). Players downloaded the game from a web page, with versions available for Windows and Mac OSX.

In addition to scripting the user interface and object interactions in TorqueScript, the C++ game engine code was modified to support mouse-directed gaze, persistent object manipulation, data logging, socket-based communication with external applications, NPC commands, and pathfinding. The first-person perspective was intended to encourage players to use their bodies and language as they would in the real world (as opposed to chat dialogue detached from face-to-face interaction). To that end, the engine was modified to animate each player's head to reflect where they were pointing the camera with the mouse. Rather than instantiating and deleting objects as they come in and out of the player's inventory (as games typically do), the engine was modified to make all objects persist, and physically attach to the player's hand when picked up, making it possible to track object positions throughout the interaction.

Code was added to log every action, position change, and chat text typed by the human players. Time-coded observations are logged to a text file, and optionally transmitted to external listening applications via sockets. External applications can also send commands to the game engine via sockets, for execution by NPCs who navigate and interact in the game world. The systems described in the following Chapters utilize this socket communication layer to observe changes to the game world, and send instructions indicating what NPCs should do next.

```
[COGLOG] 6061977 EVENT CUSTOMER(Player) ENTERED doorTrigger(RegionTrigger)
[COGLOG] 6062055 STATECHANGE CUSTOMER(Player) ATPOS="(321.146 358.792 240.541)"
[COGLOG] 6063446 EVENT WAITRESS(Player) FACING CUSTOMER(Player)
[COGLOG] 6063446 STATECHANGE WAITRESS(Player) FORWARDDIR="(0.989292 -0.145949 0)"
[COGLOG] 6063446 STATECHANGE WAITRESS(Player) ATPOS="(314.443 359.182 240.54)"
[COGLOG] 6063446 SPEECHACT WAITRESS(Player) "welcome"
[COGLOG] 6066368 EVENT CUSTOMER(Player) FACING WAITRESS(Player)
[COGLOG] 6066368 STATECHANGE CUSTOMER(Player) FORWARDDIR="(-0.990476 0.137684 0)"
[COGLOG] 6066368 STATECHANGE CUSTOMER(Player) ATPOS="(320.765 358.845 240.541)"
[COGLOG] 6066368 SPEECHACT CUSTOMER(Player) "hi"
[COGLOG] 6069039 EVENT WAITRESS(Player) FACING CUSTOMER(Player)
[COGLOG] 6069039 STATECHANGE WAITRESS(Player) FORWARDDIR="(0.989292 -0.145949 0)"
[COGLOG] 6069039 STATECHANGE WAITRESS(Player) ATPOS="(314.443 359.182 240.54)"
[COGLOG] 6069039 SPEECHACT WAITRESS(Player) "please have a seat"
[COGLOG] 6074211 EVENT CUSTOMER(Player) FACING WAITRESS(Player)
[COGLOG] 6074211 STATECHANGE CUSTOMER(Player) FORWARDDIR="(-0.894119 0.440718 -0.0794913)"
[COGLOG] 6074211 STATECHANGE CUSTOMER(Player) ATPOS="(320.765 358.845 240.541)"
[COGLOG] 6074211 SPEECHACT CUSTOMER(Player) "where?"
[COGLOG] 6077489 STATECHANGE CUSTOMER(Player) ATPOS="(319.976 359.222 240.542)"
[COGLOG] 6077493 EVENT CUSTOMER(Player) EXITED doorTrigger(RegionTrigger)
[COGLOG] 6083586 EVENT WAITRESS(Player) FACING CUSTOMER(Player)
[COGLOG] 6083586 STATECHANGE WAITRESS(Player) FORWARDDIR="(0.989292 -0.145949 0)"
[COGLOG] 6083586 STATECHANGE WAITRESS(Player) ATPOS="(314.443 359.182 240.54)"
[COGLOG] 6083586 SPEECHACT WAITRESS(Player) "customers choice"
[COGLOG] 6102211 EVENT WAITRESS(Player) FACING CUSTOMER(Player)
[COGLOG] 6102211 STATECHANGE WAITRESS(Player) FORWARDDIR="(0.989292 -0.145949 0)"
[COGLOG] 6102211 STATECHANGE WAITRESS(Player) ATPOS="(314.443 359.182 240.54)"
[COGLOG] 6102211 SPEECHACT WAITRESS(Player) "you choose where you want to sit"
[COGLOG] 6108927 STATECHANGE CUSTOMER(Player) ATPOS="(318.712 359.725 240.542)"
[COGLOG] 6108946 EVENT CUSTOMER(Player) ENTERED podiumTrigger(RegionTrigger)
. . .
[COGLOG] 6132664 STATECHANGE CUSTOMER(Player) FORWARDDIR="(0.345101 0.930914 0.119601)"
[COGLOG] 6132664 STATECHANGE CUSTOMER(Player) ATPOS="(314.176 362.945 240.543)"
[COGLOG] 6132664 SPEECHACT CUSTOMER(Player) "ummmm let's see"
[COGLOG] 6135196 STATECHANGE CUSTOMER(Player) ATPOS="(314.176 362.945 240.543)"
[COGLOG] 6135352 EVENT CUSTOMER(Player) ENTERED table3Trigger(RegionTrigger)
. . .
[COGLOG] 6139930 STATECHANGE CUSTOMER(Player) ATPOS="(307.938 362.384 240.546)"
[COGLOG] 6139930 ACTION CUSTOMER(Player) SITON chair6(DBChair)
[COGLOG] 6139930 STATECHANGE CUSTOMER(Player) SITTINGON="chair6(DBChair)"
[COGLOG] 6142821 STATECHANGE WAITRESS(Player) ATPOS="(314.443 359.182 240.54)"
[COGLOG] 6142821 STATECHANGE WAITRESS(Player) DEST="(309.925 360.189 240.632)"
[COGLOG] 6142821 ACTION WAITRESS(Player) GOTO table3(DBTable)
[COGLOG] 6142993 EVENT WAITRESS(Player) ENTERED table3Trigger(RegionTrigger)
[COGLOG] 6143039 STATECHANGE WAITRESS(Player) ATPOS="(313.318 359.394 240.541)"
[COGLOG] 6143039 ACTION WAITRESS(Player) PUTDOWN dynamic2753(DBMenu)
[COGLOG] 6143039 STATECHANGE dynamic2753(DBMenu) ATTACHEDTO="NULL"
[COGLOG] 6143039 STATECHANGE dynamic2753(DBMenu) ON="table3(DBTable)"
[COGLOG] 6143039 STATECHANGE dynamic2753(DBMenu) ATPOS="(311.177 360.455 242.635)"
[COGLOG] 6145123 STATECHANGE CUSTOMER(Player) ATPOS="(309.952 361.922 240.773)"
[COGLOG] 6145123 ACTION CUSTOMER(Player) PICKUP dynamic2753(DBMenu)
[COGLOG] 6145123 STATECHANGE dynamic2753(DBMenu) ON="NULL"
[COGLOG] 6145123 STATECHANGE dynamic2753(DBMenu) ATTACHEDTO="CUSTOMER(Player)"
[COGLOG] 6147118 ACTION CUSTOMER(Player) LOOKAT dynamic2753(DBMenu)
[COGLOG] 6147571 EVENT WAITRESS(Player) FACING CUSTOMER(Player)
[COGLOG] 6147571 STATECHANGE WAITRESS(Player) FORWARDDIR="(-0.89335 0.291024 -0.342391)"
[COGLOG] 6147571 STATECHANGE WAITRESS(Player) ATPOS="(312.704 359.534 240.541)"
[COGLOG] 6147571 SPEECHACT WAITRESS(Player) "here's a menu.  Take your time"
```

Figure 3-6: Sample log file.

```
6044508        WAITRESS WALKS TO podium
6044993        WAITRESS PICKSUP Menu
6058118        CUSTOMER WALKS TO door

6063446  [CONVERSATION BETWEEN WAITRESS AND CUSTOMER]
6063446  WAITRESS: "welcome"
6066368  CUSTOMER: "hi"
6069039  WAITRESS: "please have a seat"
6074211  CUSTOMER: "where?"
6083586  WAITRESS: "customers choice"
6102211  WAITRESS: "you choose where you want to sit"

6108727        CUSTOMER WALKS TO podium

6132664  [CONVERSATION BETWEEN CUSTOMER AND WAITRESS
6132664  CUSTOMER: "ummmm let's see"

6135196        CUSTOMER WALKS TO table3
6139930        CUSTOMER SITSON chair6
6142821        WAITRESS WALKS TO table3
6143039        WAITRESS PUTSDOWN Menu ON table3
6144661        CUSTOMER PICKSUP Menu
6147118        CUSTOMER LOOKSAT Menu

6147571  [CONVERSATION BETWEEN WAITRESS AND CUSTOMER]
6147571  WAITRESS: "here's a menu. Take your time"
6150633  CUSTOMER: "I'll have vegetable soup and the nectarine tart"

6200711        WAITRESS WALKS TO counter

6216133  [CONVERSATION BETWEEN WAITRESS AND CHEF]
6216133  WAITRESS: "one soup please"

6217133        Soup APPEARS ON Counter
6221461        WAITRESS PICKSUP Soup
6222149        WAITRESS WALKS TO table3
6225368        WAITRESS PUTSDOWN Soup ON table3
6225430        WAITRESS PICKSUP Menu
6226196        WAITRESS WALKS BETWEEN counter AND table2
6228086        CUSTOMER EATS Soup
6242774        WAITRESS PUTSDOWN Menu ON MenuBox
6242774        Menu DISAPPEARS
6245743        WAITRESS WALKS TO table3

6301211  [CONVERSATION BETWEEN WAITRESS AND CUSTOMER]
6301211  WAITRESS: "anything for drink"
6309711  CUSTOMER: "water"
6315243  WAITRESS: "coming right up"

6316743        WAITRESS WALKS TO bar

6320508  [CONVERSATION BETWEEN WAITRESS AND BARTENDER]
6320508  WAITRESS: "one water"

6321508        Water APPEARS ON Bar
6324633        WAITRESS PICKSUP Water
6325649        WAITRESS WALKS TO table3
6328743        WAITRESS PUTSDOWN Water ON table3
```

Figure 3-7: Log file transformed into a human-readable transcript.

3.1.2 Player Statistics

Servers launched on February 21, 2007, and continued running until September 9, 2010. 10,027 two-player games were recorded, played by 6,649 unique players (15,965 players logged in, but not all could find a partner online at the same time to play with). The table and figures below report statistics about the players and games played. Figures 3-9 and 3-10 compare the amount of traffic to the project web page to the number of people who actually participated in a two-player game online. The project was introduced with a blog post, and word spread to other blogs and news media outlets. Players could optionally report where they heard of the project during the registration process. Figure 3-8 illustrates the impact of different sources of communication about the project, and Figure 3-11 illustrates geographic demographics of traffic to the project web page.

| Total Players Logged In | 15,965 |
| Players Who Completed Games | 6,649 |

	Mean	Median	Max
Game Duration	10.84 min	9.5 min	153.37 min
Games Played Per Person	3	2	119

Table 3-1: Gameplay statistics from 10,027 games.
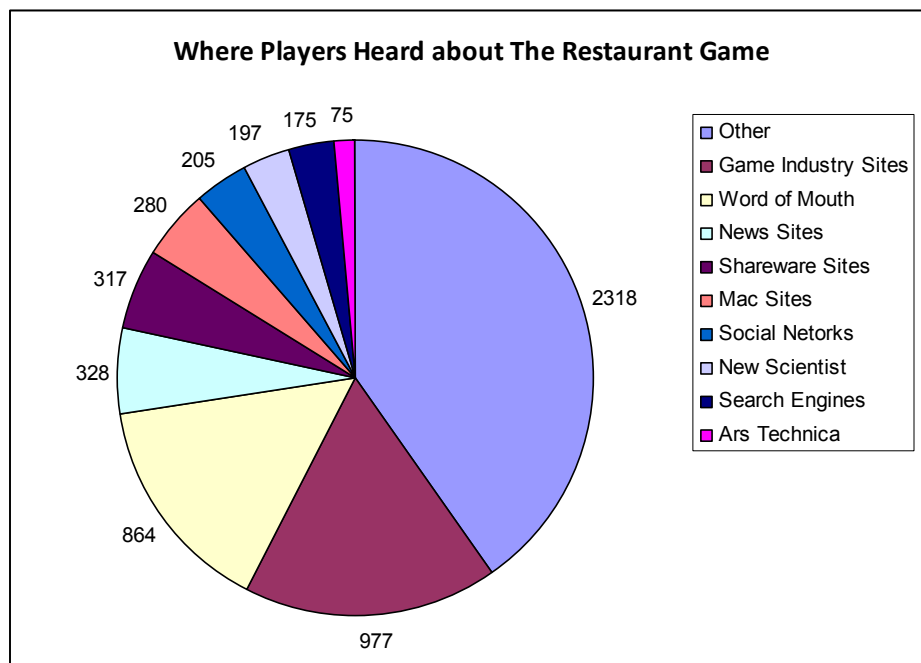
Figure 3-8: Where players heard about *The Restaurant Game*.
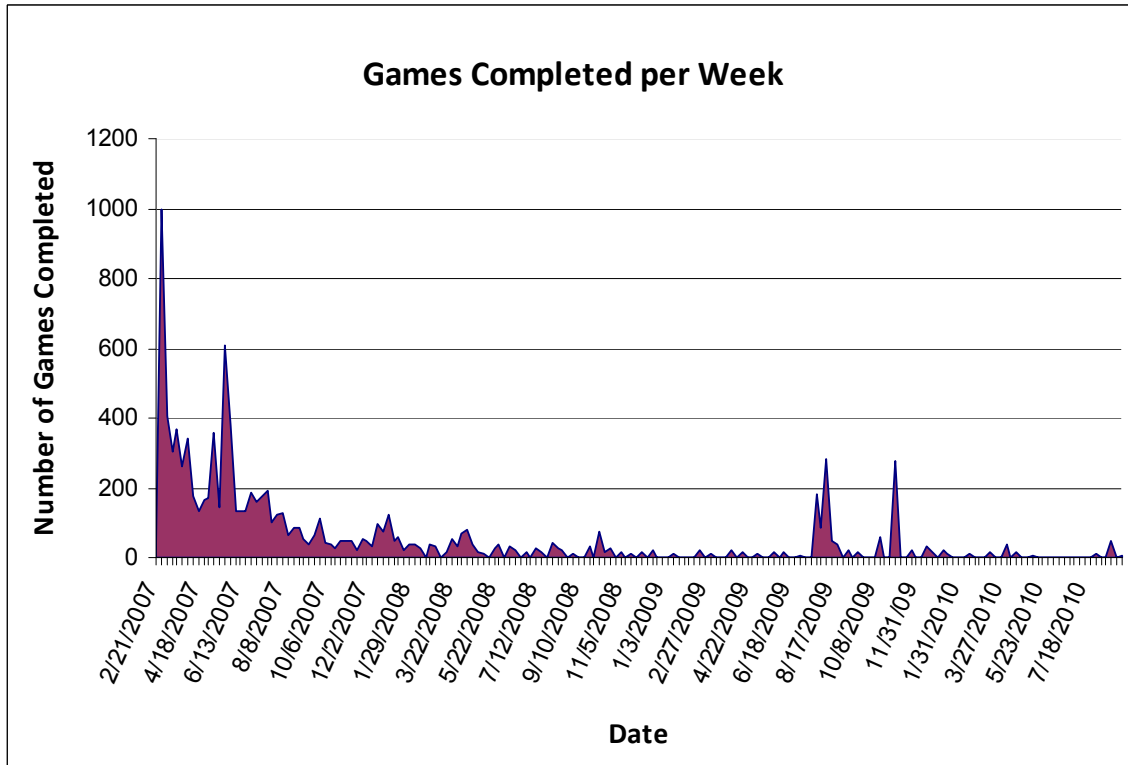
Figure 3-9: Number of two-player games completed per week.
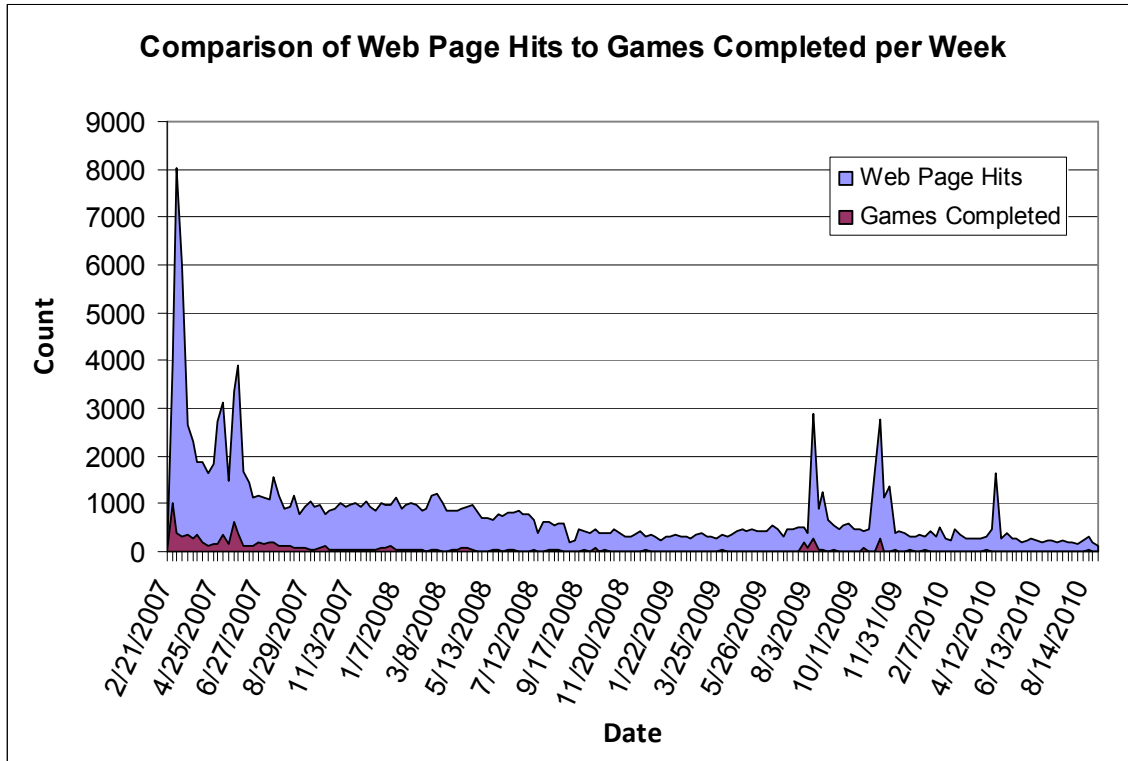
Figure 3-10: Comparison of traffic to project web page to two-player games actually completed.

Location

% of visits: 100.00%

Map Overlay

Site Usage

1 ▭ 67,059
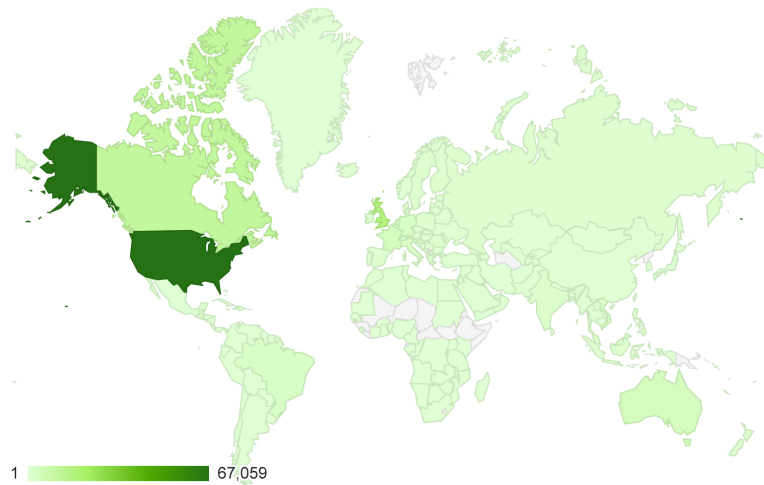
Visits	Pages / Visit	Avg. Visit Duration	% New Visits	Bounce Rate
152,676	**1.19**	**00:00:30**	**86.98%**	**85.90%**
% of Total: 100.00% (152,676)	Site Avg: 1.19 (0.00%)	Site Avg: 00:00:30 (0.00%)	Site Avg: 86.98% (0.00%)	Site Avg: 85.90% (0.00%)

	Country / Territory	Visits	Pages / Visit	Avg. Visit Duration	% New Visits	Bounce Rate
1.	United States	67,059	1.20	00:00:35	84.87%	84.73%
2.	United Kingdom	16,783	1.19	00:00:25	88.86%	85.81%
3.	Canada	10,935	1.19	00:00:26	89.78%	85.69%
4.	Australia	4,321	1.15	00:00:25	89.17%	89.08%
5.	France	4,048	1.27	00:00:26	92.79%	81.00%
6.	Italy	2,903	1.14	00:00:20	90.39%	89.46%
7.	Vietnam	2,720	1.24	00:00:29	94.41%	84.34%
8.	Germany	2,583	1.14	00:00:29	84.98%	89.01%
9.	Netherlands	2,559	1.15	00:00:34	88.55%	88.67%
10.	Brazil	2,475	1.20	00:00:24	87.47%	85.82%

Figure 3-11: Google analytics geographic demographics of project web page traffic.

3.1.3 Corpus Statistics

The tables and figures in this section detail statistics about how many unique actions and words were observed in 10,027 log files, and how the vocabulary and action lexicon grow with each additional log file. Actions are role-specific and context-sensitive, leading to thousands of potential combinations of parameters. For example, the waitress picking up pie from the counter is one action in the lexicon. Actions that interact with similar types of objects are clustered, where similarity is determined automatically based on observed affordances (e.g. customers interact with food items like steak, salmon, and salad in the same way, so food items are clustered, in turn allowing actions associated with food items to be clustered). Chapter 4 provides specifics about how log files are processed to learn the vocabulary of words and lexicon of physical actions, and how actions are clustered. The figures in this section illustrate the enormous diversity in physical interaction and language observed in spontaneous human-human interaction. The histograms highlight the sparsity of this data, where the majority of actions and words are observed only once in 10,027 games.

	Unclustered	Clustered
Size of Action Lexicon	16,622	10,165

	Mean	Median	Max
Actions per Game	106.74	95	1,187

Table 3-1: Action lexicon statistics from 10,027 games.

Size of Vocabulary	36,265 words		
Number of Unique Utterances	198,8845		

	Mean	Median	Max
Utterances per Game (All)	36.13	29	1,000
Utterances per Game (Waitress)	18.53	15	547
Utterances per Game (Customer)	18.36	14	453
Utterance Length (Waitress)	3.78	3	21
Utterance Length (Customer)	3.58	3	31

Table 3-2: Vocabulary statistics from 10,027 games.
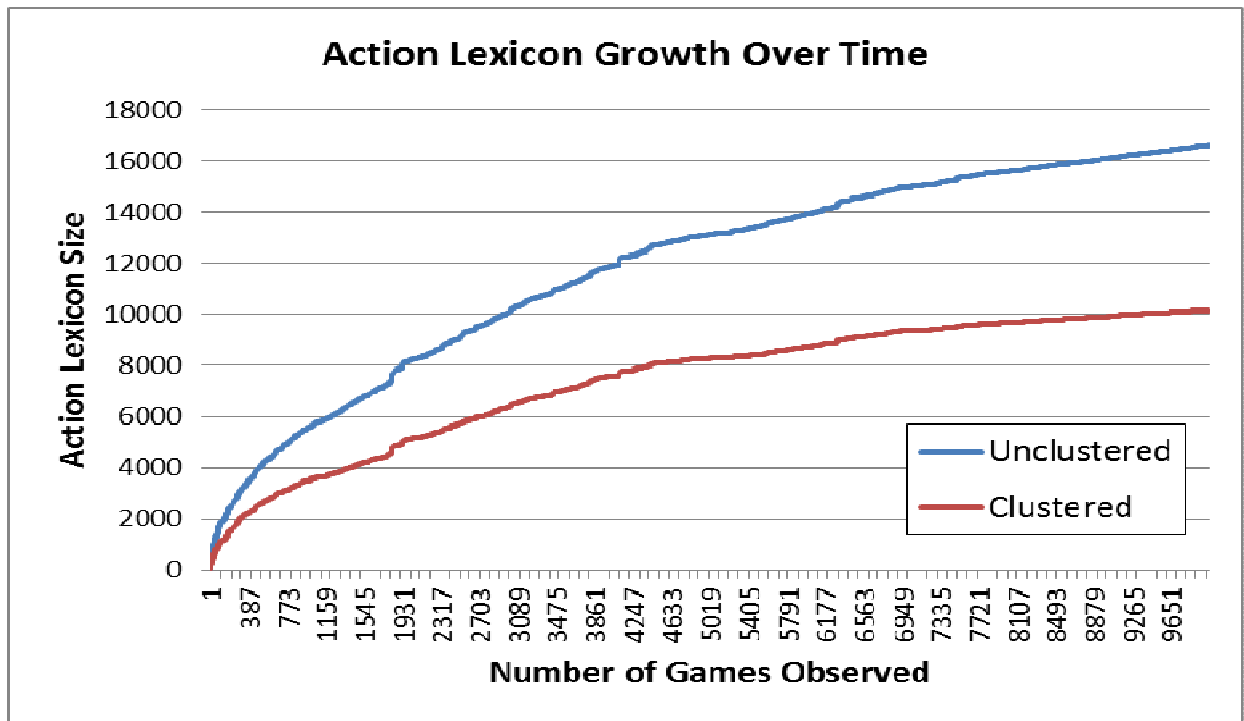
Figure 3-12: Growth of Action Lexicon as games are observed over time.
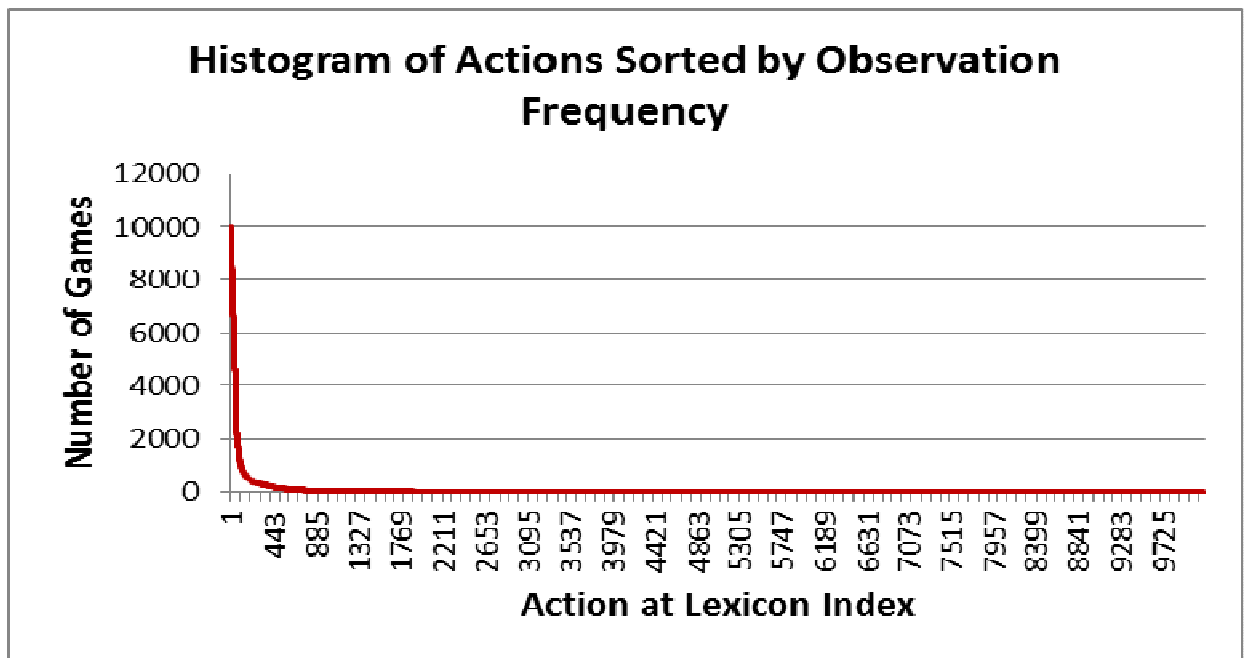
Figure 3-13: Histogram of number of games each clustered action was observed in.
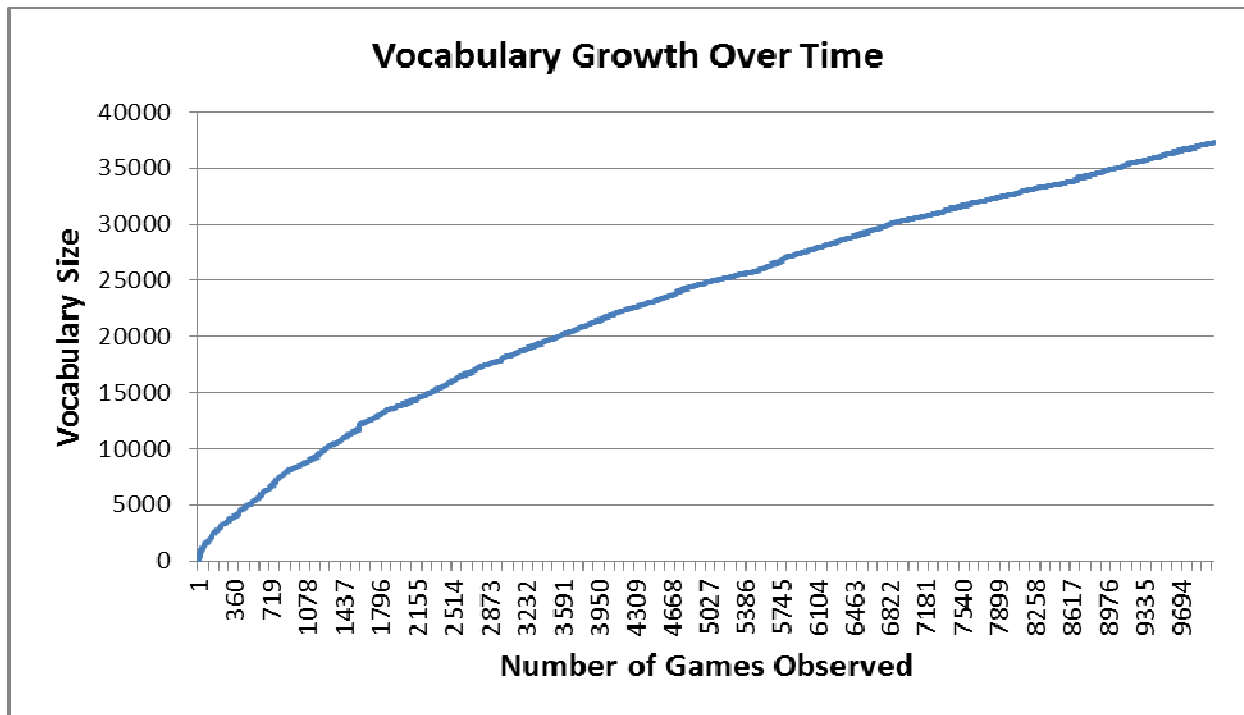
Figure 3-14: Growth of vocabulary as games are observed over time.
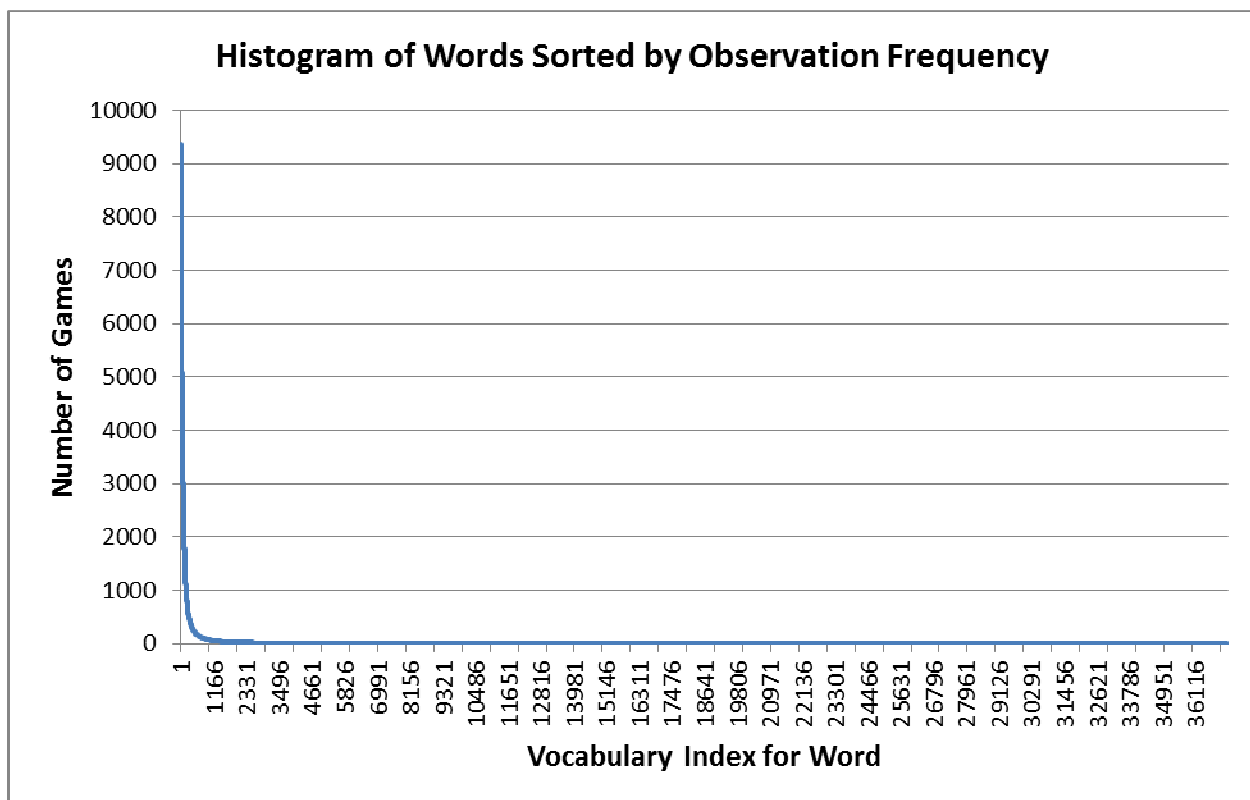
Figure 3-15: Histogram of number of games each word was observed in.

3.2   Other Data Sets

TRG is the primary focus of this thesis. Two other data sets are briefly described, as evidence that the implemented platform is generalizable, and as motivation for future directions using the same system for data-driven interaction.

3.2.1   Mars Escape

In collaboration with the Personal Robots Group (PRG) at the MIT Media Lab, TRG was adapted to create a new game for studying the interaction between a human and a robot on a space station. The goal of this project was to transfer behavior learned from the virtual world to a physical robot in the real world. Mars Escape (Chernova et al. 2010) is a puzzle-based game, which is task-focused, as compared to TRG which captures role-playing demonstrations of social norms, or the dramatic improvisation encouraged by *Improviso*. The robot avatar resembles PRG's *Nexi*, a Mobile Dexterous Social (MDS) robot, and the puzzle is a virtual recreation of a puzzle physically installed at the Boston Museum of Science. Due to the challenges of automating the real robot, the puzzle is fairly simple. Players work together to find and move a set of objects before time (measured in oxygen supply) runs out.
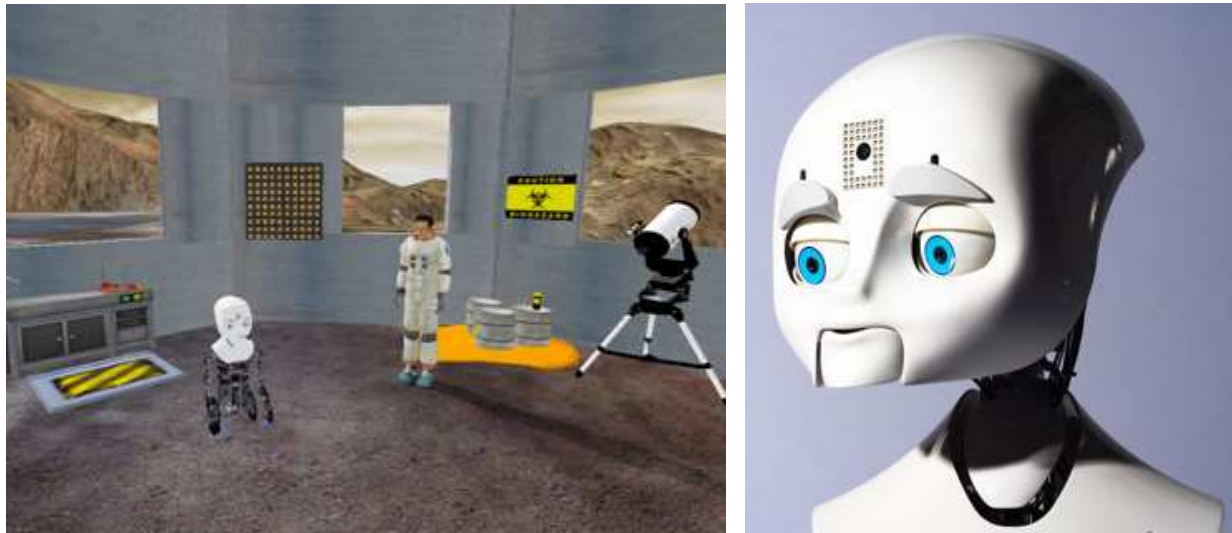
Figure 3-16: Screenshot from *Mars Escape* (left) and a photo of the actual *Nexi* robot (right).

Mars Escape recorded 558 two-player games, and the data was used to automate *Nexi*. Action sequences were extracted from the data using processes described in Chapter 4, and these sequences served as input to an existing case-based reasoning system. A study comparing reports from interactions in the online game to reports from interacting with the data-driven physical robot at the Museum of Science found comparable results when subjects were asked whether the robot performed well at the task, behaved rationally, and contributed to the success of the team (Chernova et al. 2011). This study was carried out before the implementation of the case-based planning system described in Chapter 5, however the Mars Escape puzzle is more constrained, and not as dependent on language as interaction in TRG or *Improviso*; in fact, many players completed the puzzle together without saying anything at all.

3.2.2 Improviso

Improviso (GAMBIT 2011) pairs players online as the Director and Lead Actor on the set of a low-budget science fiction film. This data collection game explores dramatic improvisation and storytelling, played through three short scenes, with up to five characters. While TRG capitalizes on social-cultural common sense to capture examples of mundane everyday social interaction, *Improviso* leverages pop-cultural common sense -- shared familiarity with alien encounters drawn from movies, comic books, and video games. The project was a collaboration with the Singapore-MIT GAMBIT Game Lab, and started from the TRG codebase. GAMBIT provided a team of summer interns (producer, game designer, sound designer, tester, three artists, and two programmers) for nine weeks. Beyond the summer session, two programmers and a part-time artist continued development for about three months to complete features, implement a tutorial, debug, and launch the game online. *Improviso* debuted at PAX East 2011, and servers ran from March, 2011 to April, 2012, recording 877 game logs.

In the first scene, the Lead Actor plays as Ted, a reporter who awakes tied up on a crashed alien spaceship, and the Director plays as Agent Smith, a government agent who may or may not be an alien in disguise. In subsequence scenes, the Director can toggle between Smith and three other characters – Dr. Sarah Stein, K-17 the alien, and a little girl named Katie. Scenes take place inside the spaceship, in the Area 51 laboratory, or in the desert exterior. The team chose a whimsical art style resembling a school play, with all sets, costumes, and props made of cardboard, as a solution to the production problem of modeling a large number of 3D assets. In addition to manipulating objects and typing chat text, the Director can spawn special effects (e.g. thunder and lightning, explosions, and alien invasion), change the sound track to set the mood, swap masks on characters (e.g. Agent Smith reveals himself to actually be an alien), and kill and revive characters.

Though automating characters from the *Improviso* corpus remains for future work, the development and launch of *Improviso* has produced valuable insights about the design and deployment of data collection games. In some respects, *Improviso* can be viewed as a study in counterintuitive consequences – a game that garnered positive press and awards, yet few people played, and embraced well-trodden science fiction themes (aliens and government conspiracies), yet confused game players – issues that will be explored more deeply in the section 3.3.

In-game overlay text:

You are Agent Smith.
You transported Ted to Area 51 for his protection.
The government is doing experiments on a captured alien.
Ted sets the alien free.
You can switch to Dr. Stein or the Alien.
You are holding a gun.
Press F10 when you want to end the s

anatomy is fascinating! look at these xrays
AGENT SMITH said to TED: I don't care about all that doctor. just tell me if it's dangerous

Character panel labels:
Dr. Stein
F1 X - 17
F2 Dr. Stein
F3 Agent Smith

Figure 3-17: Screenshot of *Improviso*. Agent Smith, Dr. Stein, and an alien inside the Area 51 lab.

Scene	Location	Description
1	Spaceship	Agent Smith rescues Ted the reporter from the alien spaceship.
1	Spaceship	Agent Smith interrogates Ted about trespassing at the classified alien crash site.
1	Spaceship	An Alien probes Ted to learn about humans.
1	Spaceship	Ted is convinced Agent Smith is an alien.
2	Spaceship	Ted and Agent Smith search for evidence of alien life.
2	Area 51	The government examines Ted in Area 51 for evidence of alien contamination.
2	Spaceship	Agent Smith reveals his true alien identity to Ted.
2	Spaceship	Another Agent Smith arrives. Ted tries to kill the imposter.
2	Area 51	After Ted is transported to Area 51, he reveals himself to actually be an alien!
2	Spaceship	Ted helps Agent Smith rescue other hostages before escaping the ship.
2	Area 51	Ted persuades Dr. Stein to free a captured alien from Area 51.
2	Area 51	Ted steals evidence about the government's alien cover up.
3	Desert	Agent Smith calls in the military to exterminate the aliens.
3	Desert	Ted seduces Dr. Stein and convinces her to set him free from Area 51.
3	Spaceship	Agent Smith saves Ted from an alien imposter of Agent Smith.
3	Spaceship	An alien imposter of Agent Smith kills Ted.
3	Desert	The Alien calls in reinforcements to destroy Earth.
3	Spaceship	Ted helps the Alien return home.
3	Area 51	An Alien breaks out of one of the tanks in Area 51 and wreaks havoc.
3	Area 51	Smith lets everyone go after Ted convinces him that it was all a misunderstanding.

Table 3-3: Scene descriptions given to players of *Improviso*.

Scene 1/3: Agent Smith interrogates Ted about trespassing at the classified alien crash site.

TED said to AGENT SMITH: HELLO!
TED said to AGENT SMITH: help!
AGENT SMITH said to TED: Shut up and listen to me!
TED said to AGENT SMITH: but I'm stuck!
AGENT SMITH said to TED: You were caught in this crash zone!
AGENT SMITH said to TED: What are you doing in here!
TED said to AGENT SMITH: I got lost, then I was attacked!
AGENT SMITH said to TED: Yes. That was me!
AGENT SMITH said to TED: And i tied you up!
AGENT SMITH said to TED: NOW SPEAK UP! Or else…
TED said to AGENT SMITH: Did you prob me too?
AGENT SMITH said to TED: I tied you down for a reason!

The soundtrack has been changed to Scary.

TED said to AGENT SMITH: I was looking for something interesting to report on
AGENT SMITH said to TED: Now... give me a good reason why i should let you out...

AGENT SMITH picked up the PROBE.

AGENT SMITH said to TED: Or i will use this probe on you!
TED said to AGENT SMITH: I'm an ex marine and can help fight the aliens
AGENT SMITH said to TED: Okay; i trust you for now. Lets go.

AGENT SMITH set TED free from the rope
AGENT SMITH picked up the HANDGUN.

AGENT SMITH said to TED: Pick up that probe and follow up!

TED picked up the PROBE.

Scene 2/3: After Ted is transported to Area 51, he reveals himself to actually be an alien!

AGENT SMITH said to TED: Now that we got out...
AGENT SMITH said to TED: You can explain yourself. Better now.
TED said to AGENT SMITH: Yes, yes I can, I feel much safer here
DR. STEIN said to TED: Let me just... what is that bump on your forehead??

DR. STEIN picked up the X-RAY.

TED said to DR. STEIN: oh nothing, you don't need to check it
DR. STEIN said to TED: You are wrong... let me xray this thing...

The soundtrack has been changed to Exciting.
DR. STEIN used the X-RAY on TED.

DR. STEIN said to TED: This isnt a normal bump... OH MY GOD! SOMETHING IS MOVING IN THERE
TED said to AGENT SMITH: OH NO

DR. STEIN picked up the TRANQUILIZER GUN.
TED has put on the K-17 mask.

TED said to DR. STEIN: YOU FOUND OUT!
DR. STEIN said to AGENT SMITH: OH NO! Its an alien!
AGENT SMITH said to TED: Drop dead; you skunk!
TED said to DR. STEIN: MY BROTHEREN WILL BE HERE SOON

AGENT SMITH attacked TED with the HANDGUN.
TED died
The soundtrack has been changed to Sad.

AGENT SMITH said to TED: I think we did it... Did we kill it?

DR. STEIN picked up the SURGICAL TOOLS.
DR. STEIN used the SURGICAL TOOLS on TED.

DR. STEIN said to TED: Its dead. The only proof of life outside there.
AGENT SMITH said to DR. STEIN: Im sorry.

Scene 3/3: An alien imposter of Agent Smith kills the real Ted.

AGENT SMITH said to TED: Ted. why are you here?
TED said to AGENT SMITH: I came to report a crash
AGENT SMITH said to TED: I know about this crash; I knew about it all the time...
TED said to AGENT SMITH: so you can help me with my report?
AGENT SMITH said to TED: Sure. what do you need help with... i can provide some... insights.

AGENT SMITH picked up the GUN.

TED said: This... this egg? what is it?
AGENT SMITH said to TED: Its my son...

The soundtrack has been changed to Tense.
AGENT SMITH has put on the K-17 mask.

TED said to AGENT SMITH: what's happening?!
AGENT SMITH said to TED: The only thing that is left to do now is to seal your hopeless fate...
TED said to AGENT SMITH: NOOOOOOOO

The soundtrack has been changed to Scary.
The DIRECTOR started the Lightning special effect.

AGENT SMITH said to TED: Prepare... to die....
TED said to AGENT SMITH: you'll regret this!

The DIRECTOR started the Blast-off special effect.
AGENT SMITH attacked TED with the GUN.
TED died

Figure 3-18: Transcript from an actual online two-player session of *Improviso*.

3.3 Lessons Learned About Game-based Data Collection

Online games provide an opportunity to collect enormous amounts of data about human behavior and language in context. With each game created, valuable lessons have been learned that could benefit designers of future data collection games.

3.3.1 Platform Selection and Timing

Games come into existence at a particular time and place within the larger landscape of technologies, trends, and communities for games. Options for target platforms and game engines change over time, and this choice has impact on which players, and how many players, a game can reach. One can only speculate why TRG attracted over 16,000 players, while *Improviso* attracted under 1,000. (TRG was online longer, but most of the data came in the first year). It is likely that choice of game engine, and associated target platforms and method of deployment had some effect on the number players reached.

It is possible that there is simply a wider audience for restaurants than aliens, as section 3.3.2 explores, but the differences in the gaming landscape between launches in 2007 and 2011 cannot be ignored. This period of time has witnessed remarkable disruptions in how, where, and who plays, as games on the web and mobile have become viable alternatives to playing games on a console or PC, with much larger potential audiences. There was much less competition for recruiting players in 2006, than in 2011 where there are many attractive (and free) options vying for players' time. In addition, many of the free-to-play games in 2011 are browser based, written in Adobe Flash and immediately accessible from a web-based portal like Kongregate, or have moved off the PC entirely to iOS and Android devices where players are increasingly spending their time.

TRG and subsequent games were built with the Torque 3D game engine (Garage Games 2006) from GarageGames, which was the most popular and well-supported engine for independent game development available in 2006. TGE requires downloading and installing games locally. The decision to re-use the TRG codebase for Mars Escape and *Improviso* was driven by the desire to leverage the infrastructure already in place (described in section 3.1.1), which would have been prohibitively time consuming to port to a new engine. A web-based or iOS game may have attracted more players for these more recent games. On the other hand, the market has become saturated with free, high-quality games, thus porting to a new platform would not have necessarily guaranteed more players. As an anecdotal data point, UCSC's experimental game Prom Week has attracted about 14,000 players on Kongregate since its launch eight months ago, a similar rate of adoption to TRG, somewhat supporting the theory that potential audience has migrated to the web.

3.3.2 Design Decisions

The alien encounter premise for *Improviso* was intended to be more appealing to typical gamers than a restaurant interaction, but the game attracted far fewer players. It is hard to argue that the choice of restaurant or science fiction theme alone affected the appeal, as there are examples of enormously successful commercial games with either theme – *The Sims* is one of the best selling franchises of all time, and numerous hit series feature aliens (e.g. *Halo, Resistance, Crysis*) (Bungie 2001; Insomniac Games 2006; CryTek 2007). The last section suggests that the discrepancy may have been due to the choice of platform, but it is equally likely that aspects of the *Improviso* design were not attractive to potential players.

Early play tests of *Improviso* revealed that the science fiction setting confused players who were expecting a different style of gameplay, with one play tester literally commenting, "Cool, it's paper Halo!" While it is undeniable that environments populated with guns and aliens are familiar to gamers, they set up expectations for the interaction that did not hold true. Guns in *Improviso* make noise when fired, but leave it up to the players to dramatize the consequence (e.g. "Ow! You shot me!"). Players expecting shooter-style gameplay were confused or disappointed, and in general players were often unsure what to say and do when playing as a government agent or scientist. For many players, a restaurant scenario may be considered safe to dramatize, while science fiction may seem like something they could fail at. We addressed confusion, and tried to decrease the intimidation of the setup in three ways: 1) We created a detailed tutorial that emphatically encouraged the player to ***act***; 2) the game begins with a video of people playing as intended; and 3) each scene begins with details, specific directions about what each actor is supposed to do in the scene. Table 3-4 describes the available scenes, and figure 3-19 illustrates how players were given specific scene directions. This approach was successful in encouraging the desired behavior (see Figure 3-18 for an example transcript of actual online gameplay), and positive press with headlines like "GAMBIT Game is Drama School for AI"[6] indicated our message was being communicated as intended. However, it is likely that we also lost players without the patience to endure the lengthy tutorial and video. In contrast, a minimal tutorial was required for TRG, because people naturally know what to say and do as a customer or waitress with minimal instruction and the theme may have led to self-selection of those players interested in engaging in social interaction (rather than combat).

---

[6] GAMBIT Game is Drama School for AI, *Rock Paper Shotgun*, March 18, 2011.
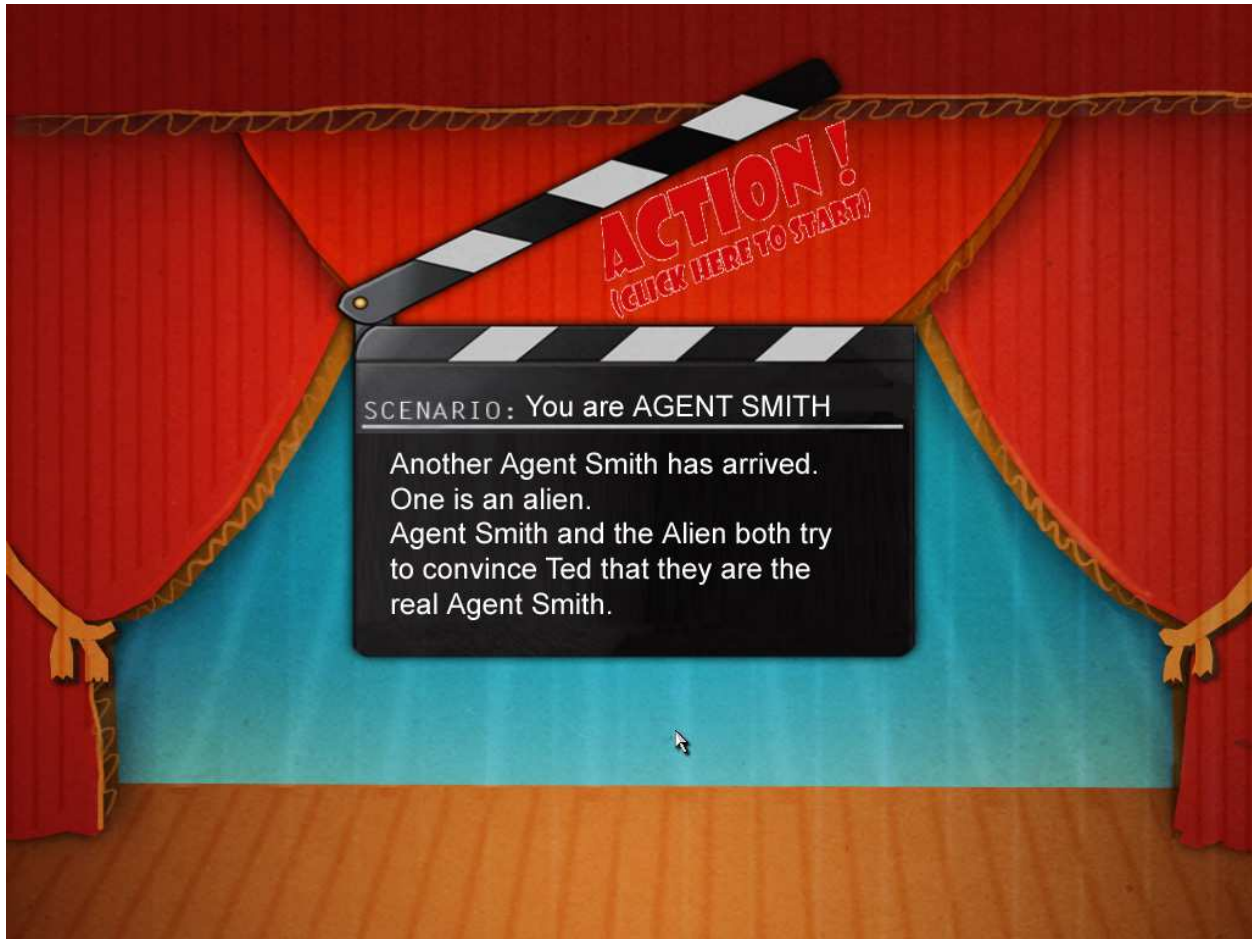
Figure 3-19: Specific scene directions given to each player in *Improviso*.

3.3.3   Recruiting and Incentivizing Players

With the number of alternatives vying for players' attention, it is valid to ask why someone would play one of these data collection games. First and foremost, these games need to be enjoyable in their own right, offering an experience that people will voluntarily complete at least once, hopefully more, and ideally will tell others about. To that end, TRG is intentionally open-ended, leaving players room to improvise, encouraging unpredictable memorable interactions, and *Improviso* goes further in the direction of entertainment with special effects, explosions, whimsical artwork, and campy objectives for each scene. This design philosophy resulted in experiences recognized as unique by the mainstream and gaming press. TRG was announced with a blog post, which attracted the interest of other blogs, and eventually media outlets including NPR, *New Scientist, The Boston Globe, The Guardian, H+,* and *SEED* magazine's design issue. In addition to similar press coverage, *Improviso* was accepted by the gaming community as a legitimate independent game, accepted as an *IndieCade 2011* finalist, winning award from Bytten.com, and featured as a game of the week on *IndieGames.com*.

A number of additional tactics were employed for TRG to entice players, but it is impossible to know the impact of each. The web page for TRG encouraged players to contribute to science, and promised to

include them as game designers in the credits of a future game to be created from the recorded data. The idea of contributing to a new game would probably be more motivating if the time between contributing data and seeing tangible results was minimized, which can be difficult in the process of research. To date the new game created from the data has not yet been released. Each play through of TRG concluded with a survey, asking players to describe who they thought the other player was in real life -- age, gender, occupation, and what they had for breakfast (see Figure 3-20). The personality profile persisted over multiple sessions, averaging the results from each partner, with the intent of encouraging replay in order to see the profile evolve.

Louis von Ahn (2006) introduced the term *Games with a Purpose* (GWAP) to refer to games designed to entice humans to complete tasks which generate data useful for solving real world problems that are easy for humans but challenging for machines (e.g. image labeling, image segmentation). Players of these games do not need to have any awareness that they are contributing data to solve a problem, and are primarily motivated by the free entertainment experience, as well as potentially entertaining social interaction. Many GWAPs pair players anonymously online, sometimes reinforcing the social aspect with leader boards tracking number of games players, and points accrued. The games described here might be considered GWAPs, as players are paired anonymously to partake in online social interaction, and players' awareness of the ultimate goal of generating an interactive system from this data is not required for participation. However, unlike von Ahn's GWAPs, the data produced by TRG requires additional interpretation and structuring through the CAI process before it is useful for other tasks.

Figure 3-20: Post-game survey in _The Restaurant Game_ asks players to describe each other.

Improviso attempted to facilitate sharing gameplay experiences virally by letting players save the script generated from their performance on the film set. Ideally a game about making movies would allow players to record a video of interaction. Unfortunately the TGE did not have support for recording video, adding such a feature was infeasible with available time and resources. The text-based script, however, did prove successful, as players were observed posting scripts to gaming forums to share experiences with others.

Achieving critical mass is a challenge for any game that requires two players to be online at once. Until that point (if that point is reached at all), it is likely that players logging in at different times, from different geographic locations will miss each other, fail to find a partner, and might not ever try to play again. 9,316 players of TRG completed the tutorial, logged in, and never found a partner. TRG and

Improviso tried to alleviate the problem by allowing the first player online to freely explore the game environment until the other player arrived, extending the period of potential overlap with another player. More development would be required to entertain players for longer periods of waiting time. An AI-controlled NPC might be a solution to entertaining a player until another human arrives, but this is of course cyclical logic, as the human data is required to automate the NPC! Perhaps in the future, if the system for automating NPCs is already in place, preliminary versions of the NPC could entertain players until humans arrive, and data from these human-NPC interactions may even be useful in evaluating the quality of the model so far.

3.3.4   Hackers and Malicious Attacks

Any online system is at risk for hacking, or other malicious attacks. In the years that TRG was online, there were several incidents. In 2009, members of a forum called FacePunch.com took interest in TRG, and managed to launch their own servers. The TGE client and server are actually the same executable, configured to run in different ways. Forum members familiar with the TGE figured out how to modify the scripts to enable running their own servers, thus valuable player data was getting captured in Sweden instead of Cambridge, MA. Fortunately one of the hackers actually sent an email asking for permission to continue running servers. Ultimately, the hackers agreed to change the game-type specified by their servers, to prevent them from appearing in the list of servers presented to ordinary TRG players. Forum members continued playing enthusiastically for about two weeks, as the hackers modified the game to allow more players, naked customers, an expanded two-story restaurant, and (of course) guns. This was a satisfactory resolution to the situation, but hints at the opportunity to leverage the incredible talents of hackers and hobbyists in the future to scale up and expand data collection efforts by allowing others to evolve or construct new scenarios, collecting a wider variety of data from scenarios that have enthusiastic support from the community.

Figure 3-21: Swedish hackers mod *The Restaurant Game*.

Aside from hacking, at various times TRG servers were besieged with players who discovered the project through online forums such as 4chan.com and somethingawful.com, who played in large numbers and exhibited a greater tendency toward malicious behavior. Not only does this produce nonsensical data, it also repels players who are trying to play the game more seriously, yet are anonymously paired with troublemakers. A bad first experience with a malicious player may leave a permanent negative impression of the game, leading a player to never try again. For periods of time where analytics of traffic to the project download web page showed the overwhelming majority coming from a site like 4chan, data was discarded.

Anti-hacking and anti-modding is not in the spirit of this work, which is trying to bring communities together online to create new interactive experiences. However, a system left completely vulnerable runs the risk of failing to generate data useful for the purpose for which these games were designed. The ideal solution strikes a balance between openness and protecting the integrity of the collected data. Two measures were taken in the implementation and launch of *Improviso* to prevent hacking and undesirable experiences online, based on experiences running the TRG data collection effort. First, *Improviso* shipped with compiled Torque Scripts, rather than source code for the scripts. While unnecessary, because TGE employs a just-in-time compiler for scripts, shipping compiled scripts makes it much more difficult for end-users to modify the final product. (Perhaps shipping separate versions with compiled and source Torque Scripts, connecting to different servers respectively, would be a more open approach). Second, at the end of each game, *Improviso* asked players whether they would be willing to play with the same partner again in the future. This data was stored on the client, and was checked on

log-in when searching for servers to join, to ensure that players were not paired online with people who they have not enjoyed playing with in the past. The intent was to inconspicuously cluster malicious players to play with each other, without them realizing this was happening. This is an alternative to banning them outright, which may provoke them to launch an attack on the game servers. In addition, it might produce interesting data about atypical play without repelling players who want to play the game as directed. While this solution was tested to validate that it works on a technical level, *Improviso* did not attract the critical mass of players necessary to evaluate whether it had the desired effect on the community of players.

4 Automatic Pattern Discovery

As computer scientists, when presented with a large collection of data, there is a temptation to apply machine learning algorithms, in hopes of automatically discovering patterns. This is the thinking that motivated the approaches described in this chapter. While a large body of literature exists covering robotic *Learning from Demonstration* (LfD) (Argall et al. 2009), LfD focuses on policy learning for low-level control systems, and has not yet reached the semantic levels required for automating role-playing characters from recorded human performances. The patterns of interest in this work represent elements of the narrative structure introduced in Section 1.4, such as dependencies, and events composed of fluid combinations of actions and words.

The approaches described in this chapter did yield promising results, however I ultimately went a different direction (human-machine collaborative pattern discovery described in Chapter 5) for reasons discussed in Section 4.4. This chapter synthesizes previously published work (Orkin 2007; Orkin & Roy 2009; 2010), as well as related work from collaborators. Aspects of this early work contributed to subsequent developments, and may be useful for future work.

4.1   Learning Recurring Sequences of Actions and Words

The log files produced by *The Restaurant Game* (TRG) capture a fluid stream of spontaneous actions, state changes, and unrestricted natural language utterances. This stream is not readily understood by a machine, thus diluting the stream into a sequence of discrete units (actions and utterances) is a required first step toward learning patterns useful for generation. Where possible, actions and utterances are clustered to facilitate recognition of recurring patterns in the data. As illustrated in Section 3.1.3, the corpus contains almost 200,000 unique utterances, and over 16,000 unique actions (due to a combinatorial explosion of variables). The Action Lexicon, Dialogue Library, and n-gram models described in this section lay the foundation for data-driven interaction described in Section 4.2.

4.1.1   Learning an Action Lexicon

The *Action Lexicon* (A-LEX) contains a catalogue of every action observed in any of the 10,027 game logs. Actions are context-sensitive and role-dependent, stored in a STRIPS-like format (Fikes & Nilssen 1971). For example, the following is an action representing a customer picking up a salad (with one bite taken) from a table while sitting in a chair:

```
ACTION:  PICKUP
  PARAMETERS:
    ROLE = CUSTOMER
    OBJECT = Salad
  PRECONDITIONS:
    ACTOR::SITTINGON = Chair
    OBJECT::ATTACHEDTO = NULL
    OBJECT::DELETED = FALSE
    OBJECT::ON = Table
    OBJECT::SHAPE = Bite1
  EFFECTS:
    OBJECT::ATTACHEDTO = CUSTOMER
    OBJECT::ON = NULL
```

The lexicon is compiled by parsing each log file, line-by-line, tracking the current state of each object, based on STATECHANGE reports. Each time an ACTION is reported in a log, the action is added to the A-LEX (or the count of an existing action is increased), with pre-conditions determined by the state of the associated target object at that time. Action post-conditions are determined by STATECHANGE reports that follow the ACTION, which share the same time code.

Many actions are functionally similar, with respect to the role they fill in patterns of interaction with other actions and utterances. For example, eating steak at the table and eating salmon at the table serve essentially the same purpose. By clustering functionally similar objects, associated actions can also be clustered, introducing a shallow hierarchy into the lexicon, which facilitates pattern discovery by reducing the size of the lexicon (at the top level of the hierarchy). Objects are clustered through an ad hoc approach, based on the probability of observing each interface action (e.g. pick-up, eat, sit-on, etc.) applied to the objects. Interface actions with probabilities above some empirically determined threshold are considered to be the affordances of the objects. Objects with the same affordances are clustered, and actions associated with these objects with matching parameters, pre-, and post-conditions are clustered. Figure 3-11 illustrates the effect clustering has on the growth of the lexicon. This approach to clustering works well for objects that are commonly manipulated, and less well for objects such as the pot and pan that players seldom touch, and thus do not get clustered. When the variety of objects is relatively small (under 100 object types), manually clustering might be preferable. An additional alternative would be a human-machine collaborative approach, where automatically clustering objects is the first step, which simplifies the task for the human to complete.

4.1.2   Learning a Dialogue Library

The *Dialogue Library* (D-LIB) contains contextually-clustered sequences of utterances observed in human-human games. This library allows an NPC to efficiently find an utterance to respond to a previously observed action or utterance in some context. The library is constructed by segmenting games into dialogues, where a dialogue is an uninterrupted sequence of utterances that follows a physical action, and is followed by another physical action. All dialogues that follow the same physical action are grouped, assumed to be contextually similar.

Utterances within dialogues are encoded with signatures which identify the salient phrases found within. The encoding process begins by identifying salient words and phrases, based on recurrence statistics about the number of games in which every sequence (of length one to ten words) is observed. Sequences observed above some threshold number of games are considered salient. Sequences are clustered if they are longer than two words, and differ by only one word. The signature assigned to an utterance is the unordered set of indices corresponding to all salient sequences found within the utterance. The same encoding process can be applied to natural language input at runtime, allowing an NPC to efficiently compare input with previously observed utterances.

4.1.3   Learning N-gram Models of Actions and Words

N-gram models refer to a simple statistical modeling technique from the field of Natural Language Processing (Jurafsky & Martin 2000). Borrowing this technique, and applying it to both actions and words gives a means of estimating the likelihood of a sequence of actions and words, which can be useful for an NPC when selecting what action to take next. Section 4.2.1 describes a system that employs an n-gram model to guide agent behavior, while Section 4.2.3 discusses the strengths and weaknesses of such an approach.

An n-gram model estimates the likelihood of a sequence of words of length N by counting the frequency of this sequence in a training corpus, and dividing by the total number of unique n-grams. Unigrams estimate the likelihood of atomic words. Bigrams estimate the likelihood of pairs of words. Trigram estimate triplets and so on. The likelihood of an entire sentence can be estimated by iterating over every sequence of N words, and multiplying the n-gram probabilities together. For example, estimating the likelihood of the sentence "The dog laughs." with trigrams looks like this:

$$P(\text{the, dog, laughs}) = P(\text{the} \mid \text{START})$$
$$\times\, P(\text{dog} \mid \text{START, the})$$
$$\times\, P(\text{laughs} \mid \text{the, dog})$$
$$\times\, P(\text{STOP} \mid \text{dog, laughs})$$

The trigram $P(\text{laughs} \mid \text{the, dog})$ is the probability of seeing "laughs" given that we have already seen "the dog." Enclosing each sentence in START and STOP markers captures the probability of starting or ending a sentence with a particular n-gram. In practice, computing the sum of the log probabilities keeps the likelihood from going to zero.

As illustrated by the figures in Chapter 3, even after 10,000 games, the action lexicon and vocabulary are still growing. Incorporating discounting and smoothing techniques counter the effects of data sparsity. In order to estimate the likelihood of previously unseen action or word sequences or atoms, true counts are discounted using Katz Back-Off smoothing (Jurafsky & Martin 2000), and the missing probability mass is distributed among previously unseen n-grams.

Figure 4-1 graphs all of the action sequences observed in 5,000 recorded games, from the start of the game at the top, to the end of the game at the bottom. The graph only includes physical actions, because generating a graph including all observed unique utterances would be intractable (though utterances can be semantically clustered into courser grained dialogue acts, as is explored in Section 4.3.1). The left side of the graph is unfiltered, while the right side is filtered with a bigram model to emphasize the strongly recurring patterns of behavior. All bigrams with a likelihood below some threshold fade into the background, and a simpler structure emerges, representing an automatically learned Schank-like "script" of typical restaurant interactions.
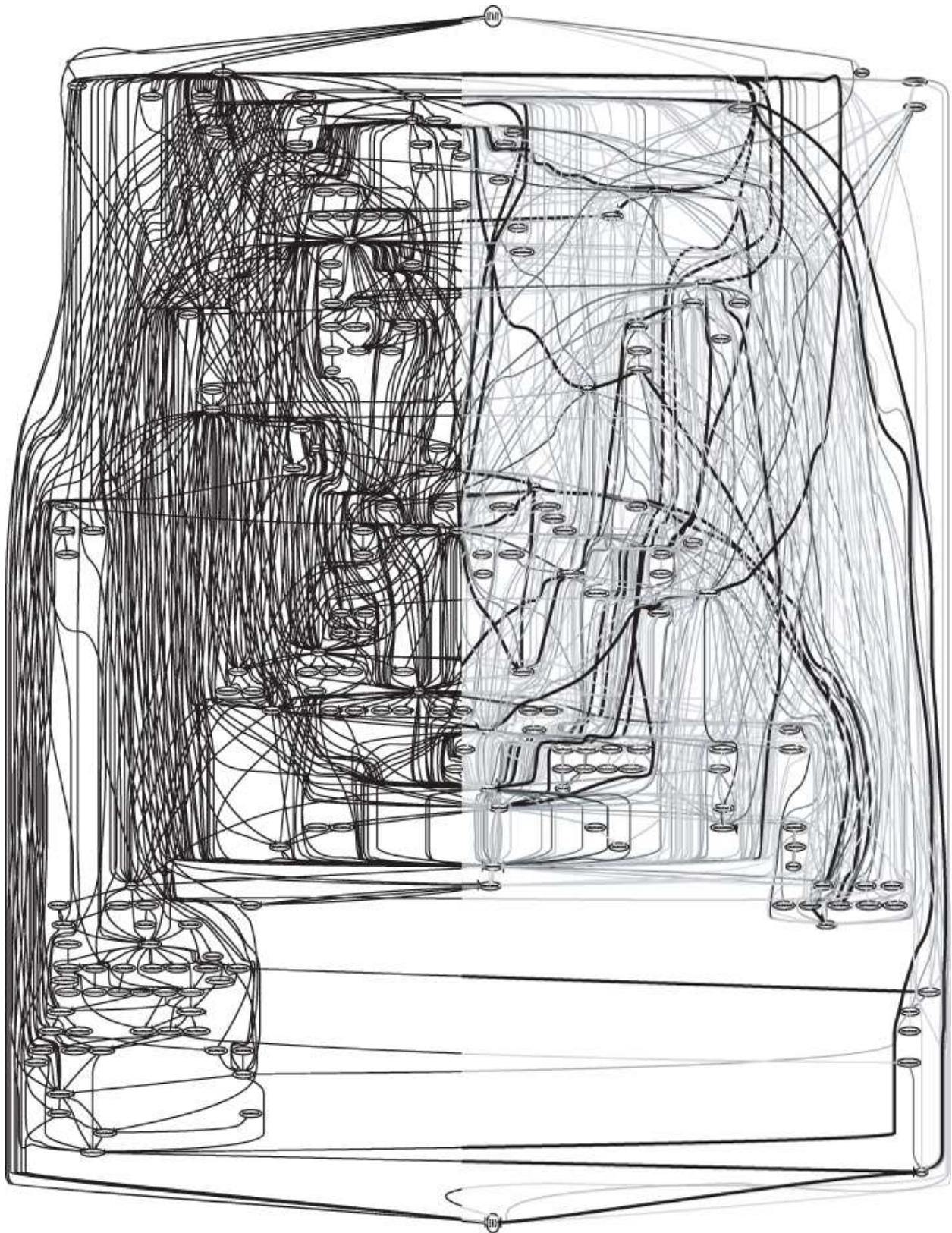
Figure 4-1: Action sequences observed in 5,000 games, unfiltered (left), filtered with bigrams (right).

A study conducted in 2007 (Orkin 2007) demonstrated that estimates of likelihood computed with n-gram models learned from TRG game logs correlate strongly with human judgment of typical behavior in a restaurant. Human subjects rated 300 log files on a 1-7 Likert scale indicating how typical they felt the interaction was for expected behavior in a real restaurant. Likelihood for these files were then computed with n-gram models trained on 5,000 logs with no overlap with the test set. Separate models were learned for actions and words, due to the fact that actions and words exist at different levels of abstraction. (Section 4.3.1 takes a step toward a representation of utterances at the correct level of abstraction to combine seamlessly with actions). The estimate for each log was computed as an interpolation between the estimates from the separate action and linguistic models, with the weights determined empirically based on a held-out validation set of 100 log files (0.75 * 4-gram action model + 0.25 * 2-gram linguistic model). Comparing the likelihood estimates to human judgment resulted in a Pearson's R = 0.576 correlation coefficient, which is strongly significant for 300 games at the $p < .01$ level. This result validates that the data collected was representative of what people actually do in restaurants, and that when placed in a familiar environment, players have a tendency to bring behaviors from the real world into the virtual.
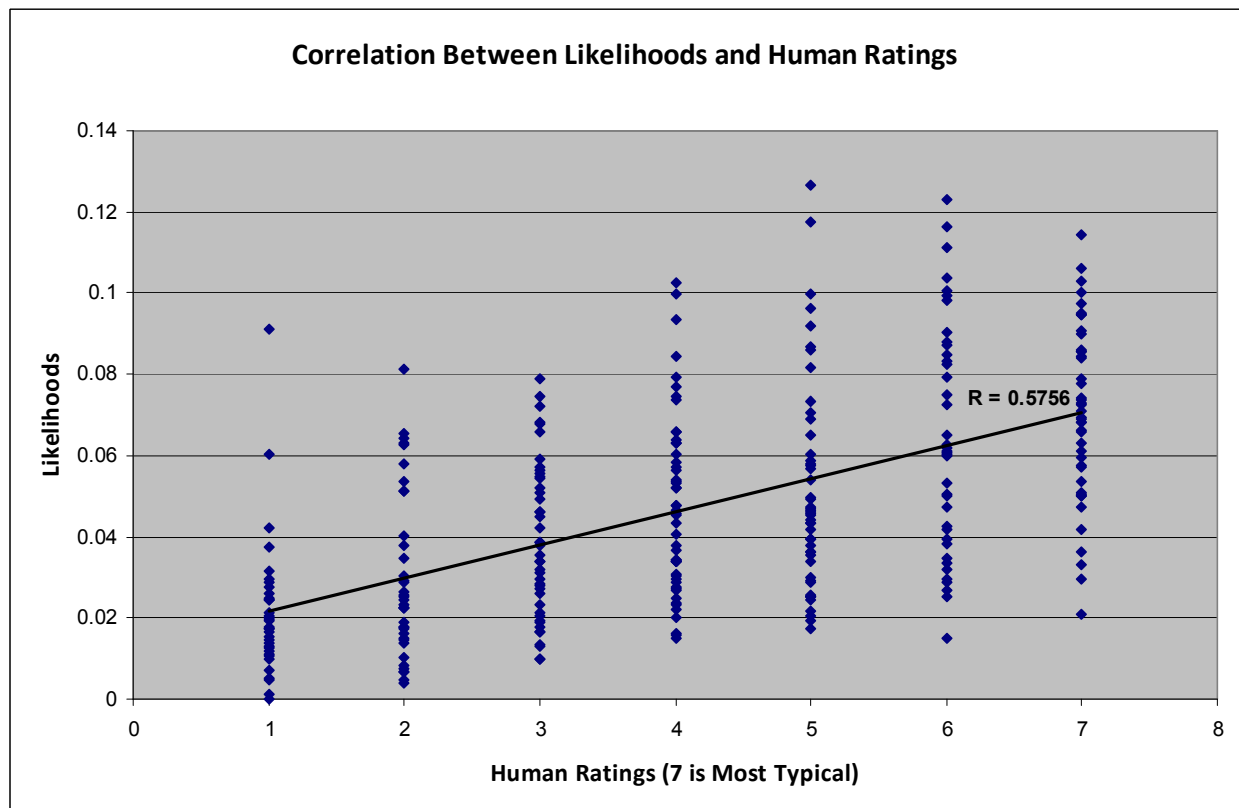
Figure 4-2: Scatter plot of correlation between n-gram likelihood estimates and human ratings.

4.2 Generating Social Behavior with N-Gram Models

The first attempt to automate data-driven NPCs combined the data structures described in Section 4.3 with a replay system, and a proposal-critique system. This section provides an overview of how this system works, how it was evaluated, and discusses the successes and failures of this approach.

4.2.1 Planning, Replays, Critics, and N-Grams

Each NPC in the game world is controlled by an agent residing on an external AI server (written in Java), networked with the game engine via sockets. This agent receives observations from the game world, replans if necessary, selects the next action for the NPC, and sends commands to the game engine for execution by the NPC. In this system, a plan refers a game log, which has been compiled into a sequence of actions and utterances -- indices into the A-LEX and D-LIB. More accurately, a plan is a pointer into a specific point in a compiled game log, which the agent replays by incrementing the pointer and sending a command for the NPC to execute the next action or utterance. An agent continues replaying the same game log until a *Sensor* detects a problem. Figure 4-3 depicts the agent architecture.
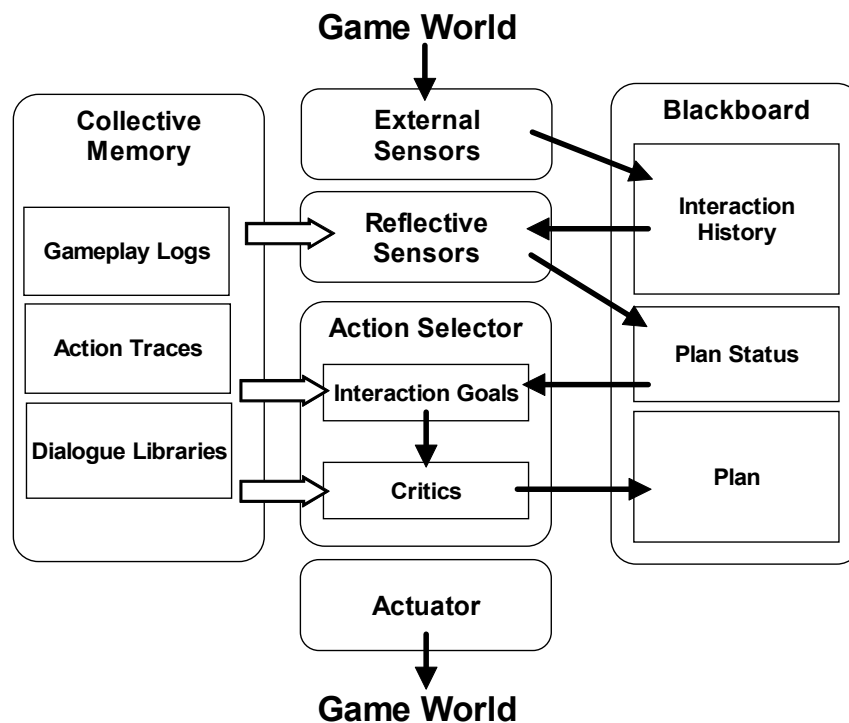
Figure 4-3: Architecture for a data-driven agent.

There are five types of *Sensors*, which invalidate plans in the event of different failure conditions. Once invalidated, the agent searches for the highest priority, relevant *Goal* to activate. The *Goal* proposes new plans to replay, while *Critic* processes scrutinize their validity. When a proposal passes all *Critics*, it becomes the currently active plan for execution through the replay system. *Sensors* and *Critics* that

judge the likelihood of the next action do so by employing a tri-gram model of actions, trained on 5,000 game logs. Tables 4-1, 4-2, and 4-3 detail the *Sensors*, *Goals*, and *Critics*.

When an agent observes natural language input (an utterance from another agent or a human player), *GoalRespondToUtterance* becomes the highest priority. This *Goal* employs the D-LIB to search for the most similar, previously observed utterance, in the current context, where context is determined by the most recently observed physical action. When multiple matching utterances exist, the utterance with the best matching history of utterances within the same dialogue is selected. In the event that the histories match equally well, ties are broken arbitrarily. Once the agent has established the best matching utterance, the proposed response is the utterance or physical action that follows the matching utterance in the associated game log.

Sensor	Description
SensorBrokenExpectation	The observed physical action does not match the next action in the current plan.
SensorExpiredExpectation	Too much time has passed while the agent was waiting for some action to occur.
SensorFailedActionExecution	The agent's action resulted in unexpected state changes, or no change at all.
SensorInterruption	A physical action was observed when an utterance was expected, or vice-versa.
SensorUnlikelyNextAction	The action that the agent plans to execute next completes is considered unlikely by the n-gram model.

Table 4-1: Descriptions of Sensors.

Goal	Description
GoalRespondToUtterance	Respond to an utterance directed at the agent by speaking or taking a physical action.
GoalWaitForInteraction	Wait to see what transpires next after the human player speaks to someone other than the agent (e.g. the chef or bartender).
GoalForceDialogueConclusion	Execute a physical action to force a dialogue to conclusion, when no response can be found.
GoalInitiatePhysicalAction	Initiate physical interaction after the agent has concluded dialogue with an utterance.
GoalInitiateDialogue	Initiate dialogue with another player.
GoalRespondToPhysicalAction	Execute a physical action in response to the last observed physical action.
GoalBeginInteraction	When all else fails, start over by initiating dialogue contextually appropriate, given the last observed physical action.

Table 4-2: Descriptions of Goals.

Critic	Description
CriticUnlikelyNextAction	The next physical action in a plan is unlikely, according to the n-gram model, given the previous physical actions in the interaction history for the same role.
CriticUnmetPrecondition	The next physical action in a plan is impossible given the agent's belief about the current state of the world. For example, the customer cannot drink a beverage if no beverage exists.
CriticHistoryMisMatch	The physical actions observed since the last dialogue do not match the physical actions that precede the next action in the proposed plan.
CriticRequiredRole	The player who will act or speak next does not match a requirement set by the *Goal* (and communicated via the blackboard). For example, after action execution fails, or an expectation expires, the agent should immediately try to say or do something else to move the interaction forward. In these cases, plans should not be approved that set up expectations for the other NPC to act or speak.

Table 4-3: Descriptions of Critics.

4.2.2   Evaluation with the BLEU Score

It is often unclear how to best evaluate systems that simulate human behavior. The data-driven planning system described in this section was quantitatively evaluated by comparing generated agent-agent interactions to recorded human-human interactions (Orkin & Roy 2009). The evaluation was based on data generated from 100 gameplay sessions, where both roles (customer and waitress) were played by autonomous agents, and compared the output to gameplay sessions between two human players.

The BLEU score (Papineni et al. 2002) from the field of machine translation was employed as the evaluation metric. In the task of machine translation, the machine is asked to translate a sentence from one natural language to another (e.g. from French to English). The BLEU score is a modified measure of precision between a candidate translation and a corpus of reference sentences, where precision is modified by limiting the total count of a word by the maximum number of times that word appears in any single reference. Ordinarily, the BLEU score is applied to sentences composed of sequences of words. In order to evaluate agent behavior, a sequence of observed physical actions can be considered to be a sentence, and can be compared to sentences formed by humans interacting. From this perspective, agents in this study can be thought to be translating their interpretation of the restaurant scenario, from human demonstrations to agent dramatizations. This metric evaluated behavior purely based on physical interactions, because at this point, representing utterances at a compatible level of abstraction had not yet been addressed. However, the system driving the NPCs (as described in the previous section) oscillates between physical interaction and dialogue surface text matching as the replay system encounters actions and utterances. Thus, linguistic interaction indirectly affects the evaluation score, as dialogue acts as conductive material providing transitions between consecutive physical actions.

The BLEU score was used to compute the modified trigram precision of an agent-agent game, as compared to a corpus of human-human reference games. From a pool of 8,430 completed games, 5,000

training games and 3,000 test games were randomly selected. The training games formed the basis of the agents' collective memories, while the test games provided references for computing the BLEU score. Figure 4-4 illustrates a comparison between the histograms of sorted BLEU scores computed for 100 agent, human, and randomly generated games, all compared with the same corpus of 3,000 reference human games. The mean BLEU scores of human and agent behavior are very close, 0.55 and 0.6 respectively, with the agent's mean slightly higher. Variance of human scores is greater, as humans are given greater freedom of expression than the *Critics* allow the agents. Standard deviation of human BLEU scores is 0.15, compared to 0.1 for agent scores. Human games score both higher and lower than any agent games. Both human and agents consistently score significantly better than the random baseline. Random baseline games are constructed by stitching together fragments of randomly selected games, until each game contains at least 20 physical actions, and an unlimited number of utterances.
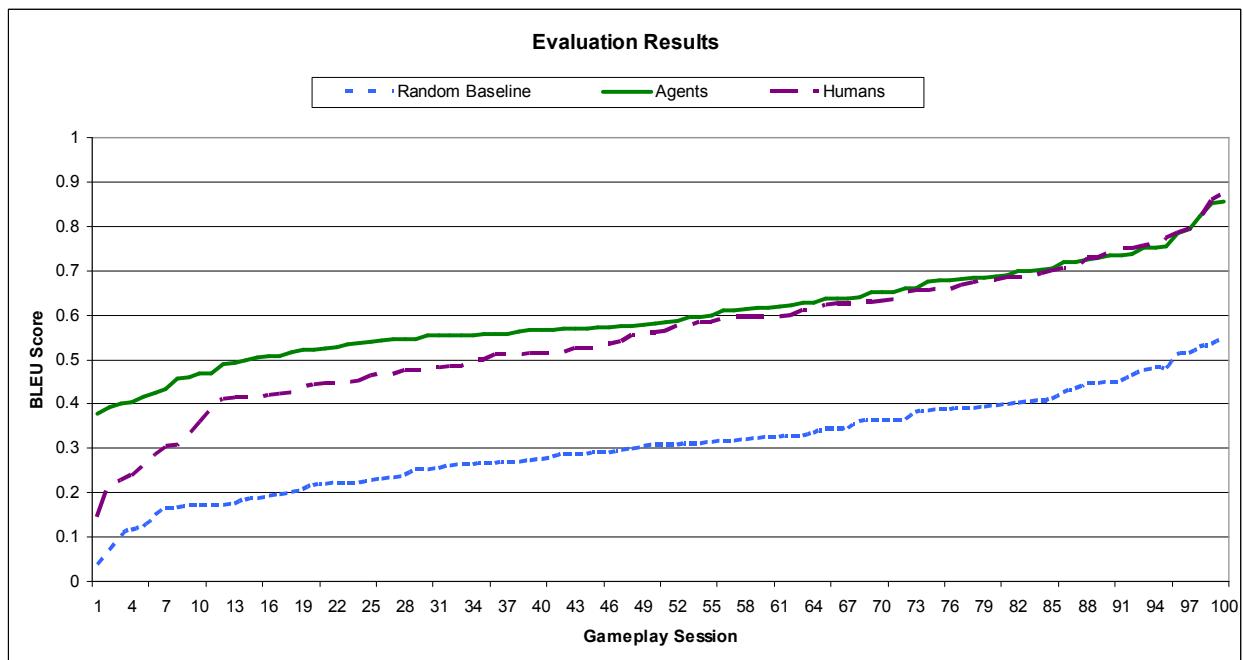
Figure 4-4: Comparing BLEU score histograms for 100 games.

4.2.3   Discussion of the Strengths and Weaknesses of Generation with N-Grams

Based on Figure 4-4, it appears as if agents are behaving as coherently as human players, or even better, but the BLEU score does not give the complete picture. Figure 4-5 reports interaction in the agent game with the highest BLEU score, 0.85. On the surface, the agents are saying and doing what one would expect to observe in a restaurant -- significant given the enormous variety of language and interaction possible in TRG, as illustrated by the poor performance of the random baseline. Fragments of interaction exhibit apparently coherent dialogue and physical behavior. Looking at the big picture, however, a human can recognize a number of issues -- customers normally order an entree only once, the waitress brings a menu after the customer agreed to the tasting menu and later brings lobster after the customer orders salad, the waitress brings the bill after the customer request to order food (again), and the waitress enters a loop of ordering many drinks after giving the customer the bill. The BLEU score

validates that the system has preserved the *texture* of human social interaction, in terms of meaningful low-level local coherence, while the metric's limited three symbol perspective is incapable of detecting issues in the higher-level intentional structure of the session. Relying only on n-grams and utterance surface-similarity produces a system that maintains low-level coherence, but over the long term reveals non-sequiturs, cycles, and a general lack of understanding of the intent and content of what is being said. One could argue that the BLEU score alone is an inadequate metric for evaluating human interaction, and must be complimented by other metrics that capture global coherence. Given the simplistic approach to data-driven interaction described thus far, which lacks any notion of higher-level structure or long-term dependencies, BLEU is an adequate metric for evaluating the low-level coherence produced by the system as implemented.

```
CUSTOMER: "hi"
WAITRESS: "hello sir"                                          // Coherent initial dialogue.
CUSTOMER: "can i get a table ?"
WAITRESS: "sure, take a seat."
CUSTOMER: "thanks"

        CUSTOMER SITSON chair6(Chair)

WAITRESS: "i recommend the a chef's tasting menu"
CUSTOMER: "yes please"
WAITRESS: "Great I'll be right with you."

        WAITRESS PICKSUP dyn029(Menu) FROM podium(Podium)     // Non-sequitur bringing
        CUSTOMER STANDSUP FROM chair6(Chair)                  // menu after an order
        WAITRESS GIVES dyn029(Menu) TO CUSTOMER               // was placed.
        CUSTOMER SITSON chair6(Chair)

CUSTOMER: "can i get a cobb salad and a glass of red wine?"   // Cycle of ordering again.
WAITRESS: "lobster please"                                    // Dependency not tracked.

        dyn078(Lobster) APPEARS ON counter3(Counter)

WAITRESS: "water please"

        dyn086(Water) APPEARS ON bar(Bar)                     // Coherent
        WAITRESS PICKSUP dyn078(Lobster) FROM counter3(Counter)  // sequence of
        WAITRESS PUTSDOWN dyn078(Lobster) ON table3(Table)    // serving and eating.
        WAITRESS PICKSUP dyn086(Water) FROM bar(Bar)
        CUSTOMER EATS dyn078(Lobster)
        WAITRESS PUTSDOWN dyn086(Water) ON table3(Table)
        CUSTOMER EATS dyn086(Water)

WAITRESS: "will your guest be joining you soon?"              // Out of context dialogue.

        WAITRESS PICKSUP dyn078(Plate)
        WAITRESS PUTSDOWN dyn078(Plate) ON counter4(Counter)

CUSTOMER: "i'm not sure.i'll go ahead and order...           // Cycle starting new order.
WAITRESS: "Ill get your bill"
CUSTOMER: "would u like to drink with me"

        WAITRESS TOUCHES cash_register(Register)              // Coherent sequence
        dyn216(Bill) APPEARS ON podium(Podium)                // of bringing bill.
        WAITRESS PICKSUP dyn216(Bill) FROM podium(Podium)
        WAITRESS GIVES dyn216(Bill) TO CUSTOMER

WAITRESS: "please pay"
CUSTOMER: "nicee!"
WAITRESS: "ok"
CUSTOMER: "one for me and one for you"
WAITRESS: "water please"

        dyn253(Water) APPEARS ON bar(Bar)

WAITRESS: "beer"

        dyn260(Beer) APPEARS ON bar(Bar)

WAITRESS: "water"
CUSTOMER: "did you know they dont have whiskey ...

        dyn268(Water) APPEARS ON bar(Bar)
```

Figure 4-5: Agent interaction with highest BLEU score, with commentary of issues on right.

Despite the issues identified above, anecdotally speaking, watching these NPCs dynamically interact and converse is certainly entertaining, and their responses to one another can often be quite witty. For example:

> WAITRESS: "smoking or non?"
> CUSTOMER: "WOW, you have a smoking section?"
> CUSTOMER: "I don't smoke, but I will start tonight. take me there!"

and:

> WAITRESS: "do you have a reservation?"
> CUSTOMER: "I have personal reservations, but I waive them when it comes to lust"

4.3 Toward Learning Deep Semantics and Higher-Level Structure

The previous section describes and evaluates a system that generates human-like social interaction based on a naive, simplistic representation of behavior and language. Deficiencies such as cyclical behavior, non-sequiturs, and lack of long-term memory highlighted in Section 4.2.3 suggest the need for more sophisticated representations that go beyond recurring patterns in surface text and action sequences. This section describes explorations into learning representations that capture the meaning of utterances, higher-level event structures, and relationships between words and abstract concepts.

4.3.1 Dialogue Act Classification

The simplistic approach of relying on utterance surface-forms (sequences of words) when trying to learn patterns of language and behavior is problematic for two reasons: 1) There are many ways to say the same thing, sometimes using entirely different words, and 2) Words are at a lower level of abstraction than actions, and cannot be combined seamlessly to learn behavioral patterns that interleave linguistic and physical actions. Addressing these issues requires clustering utterances semantically, to form dialogue acts -- linguistic actions used by players to affect changes in the game world (inspired by philosophies of Austin and Searle, as described in Chapter 2), that can co-exist as a common currency side-by-side with physical actions, leading to an integrated model of social interaction. This section details the implementation and evaluation of a dialogue act classifier, capable of transforming an arbitrary sequence of words into a dialogue act triple, which specifies the {speech act, content, and referent} of an utterance. The possible labels for elements of a triple are presented in Table 4-4. Note that each axis includes a label for OTHER covering a significant percentage of the utterances. One of the challenges for classifying dialogue acts from open-ended input is screening out off-topic or nonsensical utterances. It may be worthwhile to explore alternative approaches to filtering out these utterances in the future, perhaps classifying relevant versus irrelevant utterances in a first pass prior to classifying dialogue acts.

Speech Act			Content			Referent		
	Dist.	Pr / Re		Dist.	Pr / Re		Dist.	Pr / Re
ASSERTION	338	0.6 / 0.5	APOLOGIZE	71	0.8 / 0.9	AGE	19	0.6 / 0.5
CONFIRMATION	354	0.9 / 0.8	APPROVE	267	0.7 / 0.6	BILL	106	0.9 / 0.9
DENIAL	90	0.7 / 0.7	BRING	413	0.8 / 0.8	CUSTOMER	5	1.0 / 0.2
DIRECTIVE	1,217	0.8 / 0.9	COMPLAIN	88	0.4 / 0.1	DIET	8	0.0 / 0.0
EXPRESSIVE	724	0.8 / 0.8	CONSOLE	11	0.8 / 0.3	FLOWERS	31	1.0 / 0.8
GREETING	302	0.9 / 0.9	CORRECT	11	0.5 / 0.2	FOOD	1,394	0.9 / 0.9
OTHER	517	0.5 / 0.4	DESIRE	363	0.8 / 0.8	GEOGRAPHY	51	0.9 / 0.3
PROMISE	136	0.9 / 0.8	EXCUSEME	25	0.8 / 0.8	MENU	52	0.9 / 0.9
QUESTION	617	0.8 / 0.9	FAREWELL	110	0.8 / 0.7	MONEY	75	0.8 / 0.6
			FOLLOW	24	0.9 / 0.8	NAME	24	1.0 / 0.3
			GIVE	170	0.8 / 0.7	OTHER	651	0.6 / 0.4
			HELLO	167	0.9 / 0.9	RESTAURANT	20	0.8 / 0.6
			INFORM	176	0.6 / 0.3	SPECIALS	12	0.9 / 0.6
			LAUGH	76	0.8 / 0.9	STAFF	22	0.9 / 0.5
			MOVE	32	0.4 / 0.2	TABLE	37	0.9 / 0.9
			OTHER	643	0.5 / 0.7	TIME	107	0.9 / 0.7
			PICKUP	29	0.5 / 0.3	WAITRESS	21	0.8 / 0.7
			PREPARE	627	0.9 / 0.9			
			REPREMAND	24	0.4 / 0.3			
			SIT	74	0.9 / 0.9			
			STATUS	149	0.7 / 0.4			
			THANK	290	0.9 / 0.9			
			UNDERSTAND	25	0.8 / 0.4			
			YRWELCOME	28	0.8 / 0.8			
CORRECT:	77.3%		CORRECT:	75.3%		CORRECT:	81.1%	
BASELINE:	28.3%		BASELINE:	15.0%		BASELINE:	38.6%	
OVERALL CORRECT:		60.9%	OVERALL BASELINE:		14.3%			

Table 4-4: Label distributions and classification accuracy, precision (Pr), and recall (Re).

The dialogue act classifier is actually composed of three independent Hidden Markov Model (HMM) classifiers (Baum & Petrie 1966), one for each axis (speech act, content, and referent). An HMM classifier exploits transition probabilities in the temporal patterns that emerge in human dialogue to boost classification recognition beyond that of individual utterances. Specifically the SVMhmm classifier (Joachims 2008) was employed, which combines a Support Vector Machine (SVM) (Corinna & Vapnik 1995) for observation classification with an HMM for learning temporal patterns of hidden states. Words and contextual features function as observations, and the labels themselves are the hidden states. This combination of an SVM and HMM has proven successful for dialogue act classification previously (Surendran & Levow 2006).

Training the classifier requires each line of dialogue to be transformed into a vector consisting of features derived from the surface text, and contextual features based on the physical situation of the speakers. Contextual features include the social role of the speaker (waitress or customer), the posture

of the speaker (sitting or standing), who the speaker is facing (one or more of: customer, waitress, bartender, chef), and the containing spatial region of the speaker (one or more of the possibly overlapping regions: inside-the-restaurant, outside-the-restaurant, entrance, podium, table, counter, bar, behind bar, kitchen). The physical state of the players is reported explicitly in the game logs. The text-based features primarily consist of indicators for the presence of unigrams, bigrams, and trigrams of words observed to be salient for particular labels, as well as a smaller number of indicators for symbols and punctuation ('?', '!', '$', emoticons, and digits). Salience is computed based on the mutual information between n-grams and labels, where mutual information is a measure of statistical dependence (Cover & Thomas 1991). Salient indicators of the text-based feature set are customized for each axis by computing the mutual information between every label and every unigram, bigram, and trigram. The feature set for a classification axis is the compilation of the top 50 unigrams, bigrams, and trigrams for each label.

Despite the apparent freedom, players of TRG tend to constrain their dialogue to social conventions associated with the mutually understood "scripts" of restaurant interaction. This contributes to strong classification results given the challenge of correctly classifying three independent axes capable of producing 4,050 unique triples. Table 4-4 presents classification results, evaluated with 10 fold cross validation (each fold trained on 90 game logs and tested on 10). For each of the classification axes, precision and recall is reported for each label, followed by the percentage classified correctly and a comparison baseline. All of the axes perform significantly better than baseline, contributing to 60.9% of the utterances being classified entirely correctly – correct on all three axes.

4.3.2   Learning Other Semantics

There are numerous other possible ways to exploit data from TRG to learn aspects of language and behavior. As two examples of these possibilities, this section briefly summarizes approaches taken by colleagues to automatically learn higher-level event structure (Smith 2011) and words for concepts (Reckman et al. 2010).

Smith (2011) applied an iterative, bottom-up clustering and sequence mining approach to discover strongly recurring patterns at increasing levels of abstraction, from sequences of words up to abstract events composed of actions and utterances. The process begins by clustering words based on their linguistic environments (surrounding words) employing the Affinity Propagation algorithm (Frey & Dueck 2007). Affinity Clustering was chosen because it does not require specification of the number of clusters apriori. Once words have been clustered, sequences of words (n-grams) are mined using the Pre-Order Linked Web Access Pattern (PLWAP) algorithm (Ezeife et al. 2005). The process is repeated, now clustering sequences of words. The process continues, working up to clustering entire utterances, based on surrounding actions and utterances, which leads to mining sequences of interleaved actions and utterances, representing events. Iterating over the entire process incrementally improves results, as previously discovered clusters at higher levels of abstraction provide valuable information about the linguistic or behavioral environment, useful for re-clustering at lower levels. This approach proved successful in discovering exemplars -- prototypical examples of events one would expect to observe in a restaurant interaction. While many subtle variations of behavior and language get discarded due to sparse data, automatically discovering exemplars could be a useful way to explore a new dataset in an unfamiliar domain.

Reckman (Reckman et al. 2010) demonstrated a simple, unsupervised, data-driven approach to learning words and phrases for concepts, where concepts are physical objects that players interact with, such as different types of food. Dialogues were segmented according to the procedure described in Section 4.1.2, and a χ^2 correlation (Manning & Schutze 2000) was computed between each word, and the physical object(s) interacted with in the action that followed the last utterance of the dialogue. This process was then repeated for bigrams and trigrams. Ultimately this approach was successful at learning common words and phrases used to refer to items on the menu (e.g. steak, filet mignon, soup du jour, red wine, coffee). It remains for future work to determine if success depends on characteristics of the TRG dataset, and how well the approach generalizes to other domains.

4.4 Limitations and Opportunities for Statistical Approaches in CAI

While the results presented in this chapter are encouraging, and progress toward automatic pattern discovery is being made, ultimately this research changed direction toward a human-machine collaborative approach to pattern discovery. By 2010, it became clear that it was not going to be possible to automatically learn all of the elements of the narrative structure within the time frame of a single PhD. Some of the remaining challenges include disentangling interleaved events that overlap in time with other events, recognizing dependencies spanning arbitrary lengths of time, and clustering utterances semantically that share no common vocabulary. Some of these problems, such as the latter, may not be possible for a machine to solve without human intervention, due to required common sense or domain specific knowledge that does not exist in the data itself (though an automatic system may be able to mine relevant information from external sources of data on the internet). This section discusses arguments in favor and against the continuing pursuit of completely automating the discovery process.

Machine learning (ML) is a controversial topic among professional game developers. While it is widely recognized that there is an opportunity to exploit data from players that is increasingly easy to record, moving from carefully hand-crafted experiences from designers, to more emergent experiences automated by machines raises concerns. The most frequently voiced objections are the lack of designer control, and difficulty in debugging and modifying learned behavior[7]. Designers are not typically well versed in ML, thus ill-equipped to tune parameters or tweak feature selection. ML algorithms may learn the wrong things, and retraining the algorithms may fix one problem, while introducing a variety of undesirable side-effects. Statistical approaches estimate the probability or likelihood of a decision, while developers desire control over selecting the correct action for an NPC. Developers rely on certainty, while ML algorithms exist in a world of uncertainty. In other words, in order to guarantee an intended end-user experience, developers prefer systems that make discrete decisions, which can be debugged and definitively corrected if necessary.

In addition to the practical arguments above, there are considerations regarding ML that are more closely related to the goals of this thesis. The motivation to collect large amounts of data from human players is to create NPCs who's performances capture the nuance and diversity of human behavior and

---

[7] For example, see Kevin Dill's post on the Game/AI industry developer blog: http://www.ai-blog.net/archives/000178.html

language, and who are able to leverage this data to understand human partners. Examples of nuance and variety are inherently sparse -- some of the most interesting ways to say or do something are observed only a few times in 10,000 games, or even only once (as illustrated by the histograms in Figures 3-13 and 3-15). While collecting more data will alleviate the problem, there will always be a long tail, and at some point it may become impractical to continue collecting data in hopes of covering all gaps. This desire to capture nuance and variety despite its sparsity led to the human-machine collaborative approach described in Chapter 5, where there are humans in the loop, not only creating content, but also interpreting data, enabling systems that can exploit demonstrations seen only once in 10,000 games (or more).

This is not to say that ML has no role in CAI. There are many opportunities for statistical methods to improve the authoring and debugging processes. Chapter 5 describes an approach to processing data where humans are employed to annotate events, long-term dependencies, and attitudes. This approach relies on humans to establish lists of possible labels for the data, as well as exemplars to guide annotators. As suggested in Section 4.3.2, unsupervised algorithms can be effective in discovering patterns in new data sets from unfamiliar domains, facilitating the process of establishing the set of possible labels for the data. Automatically learned patterns can also be exploited to assist a human in semi-automating the labor-intensive process of tagging data. Human data annotation is likely to produce many errors and inconsistencies, which a statistical system can catch preemptively before interactive debugging begins.

In my experience, executing the process described in Chapter 5 to annotate the data, and completing the loop to automate a coherent interaction from this meta-data, helped clarify exactly what would be useful to learn automatically in the future, hinting that the approaches described in Chapters 4 and 5 might be more complementary than mutually exclusive, and there is likely potential for a hybrid process leveraging strengths of each approach in the future. For example, a small amount of annotated data could establish confidence metrics for an automatic system. Where the automatic system lacks confidence, it could delegate annotation tasks to humans online.

5 Human-Machine Collaborative Pattern Discovery

There are two revolutions occurring simultaneously, related to the increasing availability of processing power and storage required to collect, store, and work with large data sets. On the one hand, powerful machine learning and data mining algorithms exist to automatically discover patterns in data, as explored in the last chapter. On the other hand, realities of today's connected digital world have enabled viable alternatives to clever algorithms. Human Computation (von Ahn & Dabbish 2004) refers to an approach in which problems that are difficult for machines but easy for humans (e.g. image labeling) can be divided into small manageable tasks, and distributed online for humans to solve. There are pros and cons to each approach, and combining them is likely a fruitful direction. While there are obvious benefits to a completely automated process, unsupervised machine learning algorithms can be difficult to control (especially by non-experts), and their performance degrades in the face of sparse data. In contrast, humans can generalize from few examples, bringing to bear background knowledge from a lifetime of experience.

This chapter explores a human-machine collaborative approach to pattern discovery, as part of the Collective AI (CAI) process. CAI begins by recording thousands of human-human performances in multiplayer games. The motivation for recording humans online is to capture the nuance and variety of behavior and language, subtleties that wash away statistically due to sparse data. Employing humans to interpret data can capture valid examples of interaction that may have only been observed in few games, or even only once. Tagged data is automatically mined to discover patterns in the data, and these patterns are exploited at runtime to recognize behavior, and retrieve similar cases (recorded game logs) to help select the next action. This chapter describes EAT & RUN, an end-to-end process for tagging data, mining patterns the data, and exploiting these patterns at runtime to drive interactions with humans or other NPCs. EAT is an acronym for the *Event Annotation Tool*, while RUN (not an acronym) refers to the runtime planning architecture.

5.1   Tagging with the Event Annotation Tool (EAT)

The *Event Annotation Tool* (EAT) is a browser-based data annotation tool. EAT enables hiring non-experts online to interpret and tag data. Game logs are translated into a human-readable transcript, presented to annotators as horizontal timelines of nodes representing actions and utterances. Annotators use EAT to tag nodes with several varieties of meta-data: events, event hierarchies, causal chains, references, personalities, and domain-specific tags. EAT is written in Adobe Flex and Action Script 3, which stores and retrieves data from a WAMP web server via a PHP script.

Annotators do not require specialized skills, aside from English fluency. After reading a short web-based tutorial (Appendix A), people applied for work by annotating one sample gameplay transcript. Seven people were hired to annotate the TRG data via oDesk.com, residing in the Philippines, India, Pakistan, and the U.S. It took this team 415 person-hours total to complete annotation of 1,000 logs[8], and it cost

---

[8] The 415 hours covers the most significant tagging effort, but does not include tagging of an auxiliary set of additional lower-priority events that were added later, or tagging of personalities and domain-specific tags.

about $3,000. Tags were manually spot checked for quality, iterating on corrections with annotators. EAT includes an integrated system for leaving notes to taggers for corrections. Iterating to correct their own errors helps educate taggers about desired conventions, improving future work, and decreasing time required for quality assurance. Work was spread over two months, but seven people working 40 hours/week could complete 415 hours of work in 1.5 weeks. In the process of annotating, annotators are also asked to identify game logs where the human behavior has no resemblance to what people do in a restaurant, and manually flag these logs as junk. For example, games where players spend the entire game ordering beers, trying to fill the restaurant. 227 logs out of the 1,000 annotated logs were flagged as junk by humans (22.7% of the annotated corpus).

Strictly speaking, this online annotation process is really outsourced, rather than crowdsourced. Crowdsourcing refers to dividing a task among large numbers of anonymous workers hired through a platform such as *Amazon's Mechanical Turk*. The work described here can be considered a pilot test of what should be possible to crowdsource in the future, based on the quality of work completed by people hired online with minimal vetting or training. However, the tasks might be further sub-divided for crowdsourcing. For example, each "Human Intelligence Task" might require tagging only one type of event, instead of the entire set. In addition, scaling up the number of annotators would require automating some of the quality control, by comparing redundant tags from different annotators, and comparing a portion of work to gold standard annotations.
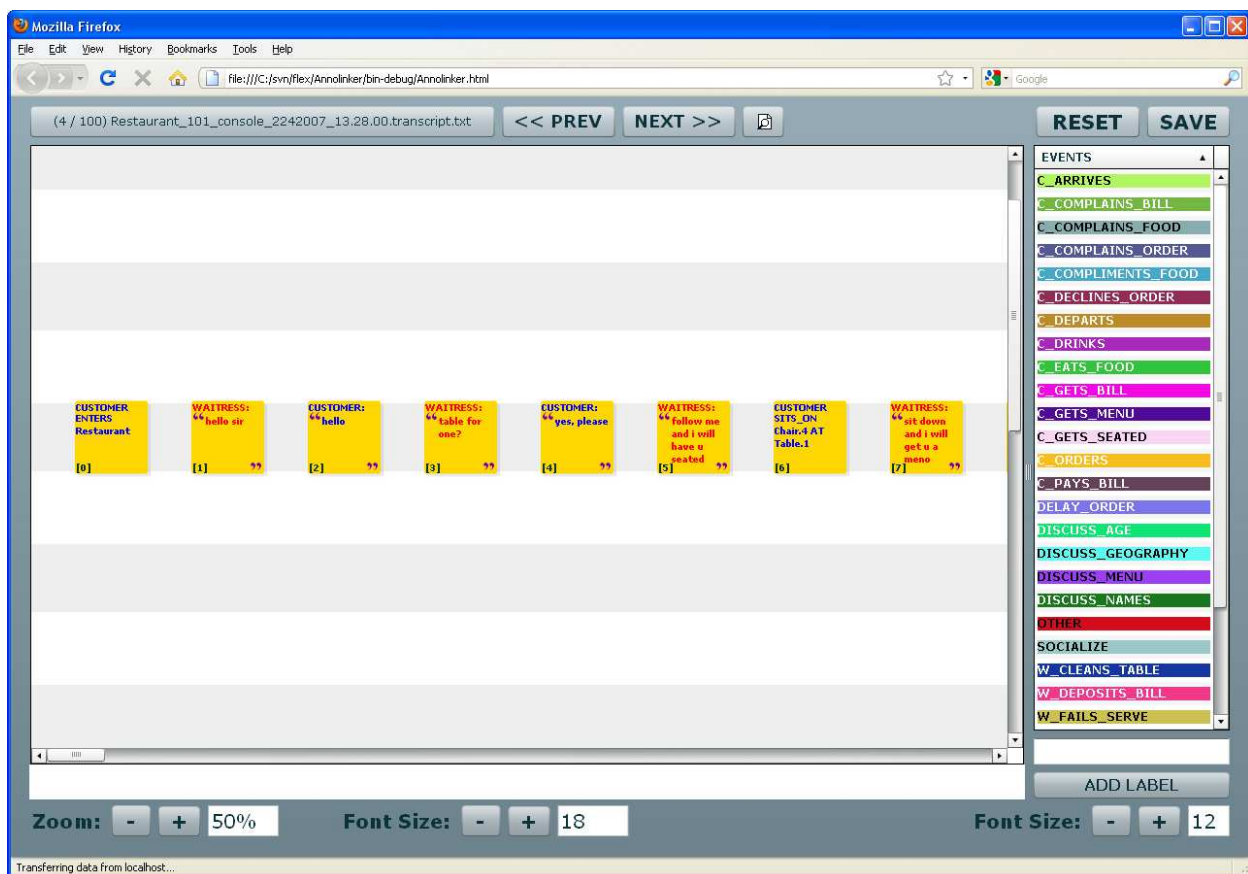
Figure 5-1: Screenshot of the Event Annotation Tool (EAT), before applying any tags.

5.1.1   Tagging Events

Annotators draw boxes around sequences of nodes to tag events, and draw bigger boxes around multiple events to tag event hierarchies. Events may contain nodes representing both actions and utterances, arbitrarily intermixed. Annotators can move nodes vertically, to separate events that overlap in time. Each node may only be contained by one low-level event. The core set of events for TRG includes 31 low-level events (e.g. C_GETS_SEATED, C_ORDERS, C_PAYS_BILL), grouped into five higher-level events (e.g. BEGIN_DINING, CONCLUDE_DINING, FULFILL_ORDER). Note that the prefixes refer to Customer (C_) or Waitress (W_). An auxiliary set of events was added after tagging of the core set was completed. A domain expert defines the list of event labels, and provides examples for annotators on a web page. See Table 5-1 for the complete list of event labels. Section 4.3.2 suggests the potential to automatically discover events in the future, though a human would still be required to associate labels with the discovered events. A study of inter-annotator agreement found substantial agreement, on a ten game subset, between event annotations of an expert and five novice annotators (mean kappa 0.81) (Orkin et al. 2010).

Figure 5-2: Timeline before applying any tags.

Figure 5-3: Same timeline as Figure 5-2, after tagging low-level events.
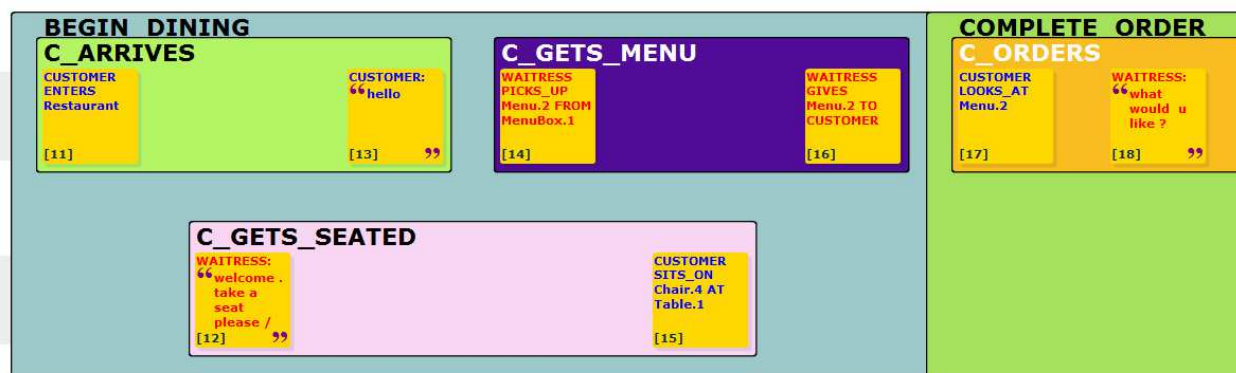
Figure 5-4: Same timeline as Figure 5-3, after tagging high-level events.

Core Low-Level Events	Auxiliary Low-Level Events
C_ARRIVES	BITES_PLAYER
C_COMPLAINS_BILL	C_ASKS_MANAGER
C_COMPLAINS_FOOD	C_BITES_DISH
C_COMPLAINS_ORDER	C_DINES_DASHES
C_COMPLIMENTS_FOOD	C_EATS_FLOWERS
C_DECLINES_ORDER	C_EATS_TRASH
C_DEPARTS	C_GIVES_FLOWERS
C_DRINKS	C_PICKSUP_APPLIANCE
C_EATS_FOOD	C_SITS_FRIDGE
C_GETS_BILL	C_SITS_STAFF
C_GETS_MENU	C_SITS_TABLE
C_GETS_SEATED	C_STEALS_REGISTER
C_GETS_UP	DISCUSS_BATHROOM
C_ORDERS	DISCUSS_DATE_ACCEPT
C_PAYS_BILL	DISCUSS_DATE_REJECT
DELAY_ORDER	DISCUSS_GENDER
DISCUSS_AGE	DISCUSS_JOIN_ACCEPT
DISCUSS_GEOGRAPHY	DISCUSS_JOIN_REJECT
DISCUSS_MENU	DISCUSS_OCCUPATION
DISCUSS_NAMES	W_CALLS_POLICE
OTHER	W_COMPLAINS_TIP
SOCIALIZE	W_DRINKS
W_CLEANS_TABLE	W_DROPS_ITEM
W_DEPOSITS_BILL	W_EATS_FOOD
W_FAILS_SERVE	W_PAYS_BILL
W_ITEMIZES_BILL	W_SITS_STAFF
W_PLACES_MENUS	
W_PREPARES_BILL	**High-Level Events**
W_RETURNS_MENU	BEGIN_DINING
W_SERVES_DRINK	COMPLETE_ORDER
W_SERVES_FOOD	CONCLUDE_DINING
	FIX_BILL
	FIX_ORDER

Table 5-1: Event labels.

5.1.2 Tagging Long-term Dependencies

Using the same codebase, the event annotation interface was adapted to support tagging long-term dependencies -- forward and backward. Arrows from one node to another tag causal chains (forward) and references (backward). Causal chains explain, for example, that the customer asking for steak caused the waitress to bring back a steak from the kitchen. Any node may be the source or destination of multiple arrows. A reference explains that a waitress who asks "How was your lobster?" is referring to the previously served lobster. The number of nodes between the beginning and end of a dependency is unconstrained. Someone at the end of the game may refer back to something that happened hundreds of actions earlier. For this reason, the interface for tagging dependencies is split-paned, showing the same timeline in each pane, allowing the annotator to scroll to different sections of the timeline, and see both at once. Figure 5-5 presents a screenshot of tagging a causal chain, where the waitress asks the chef for salad and lobster because the customer ordered salad and lobster.
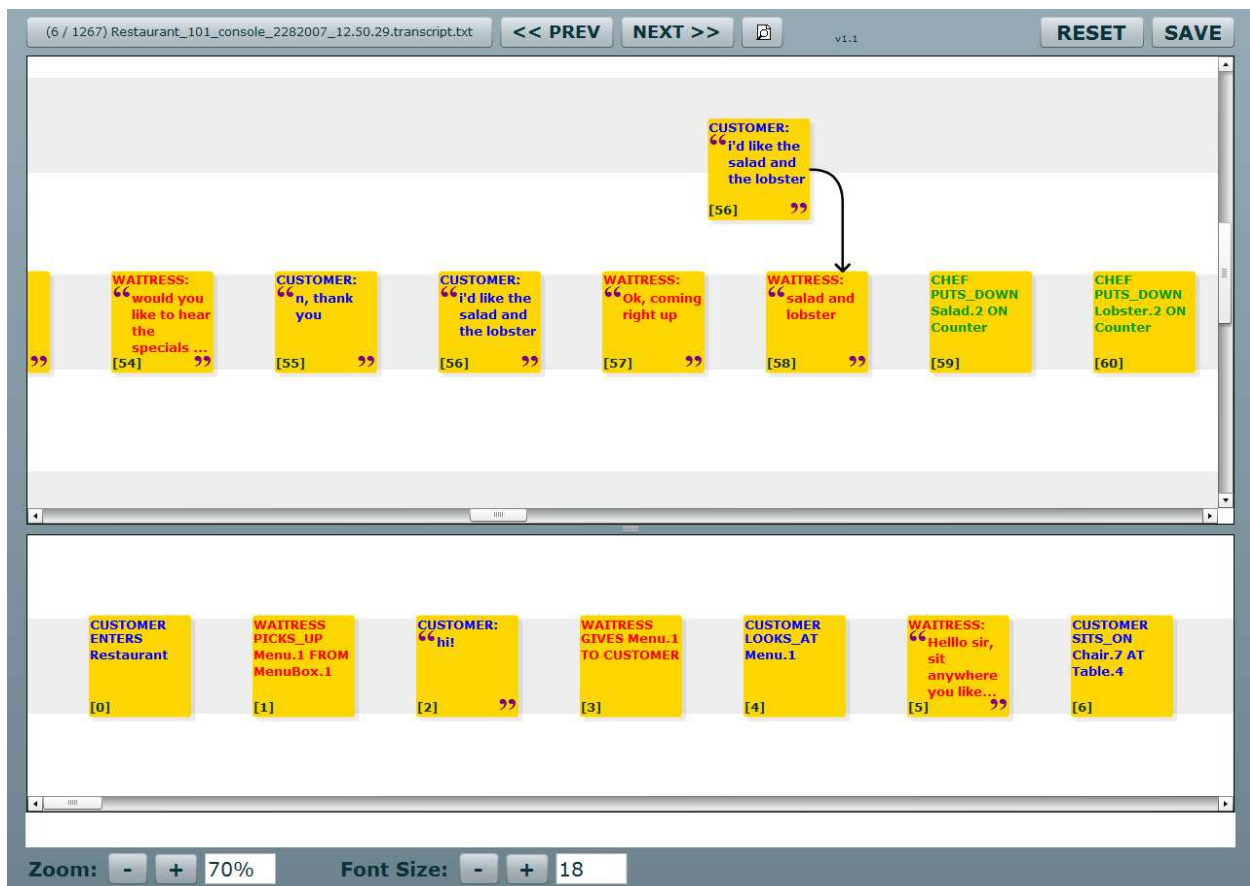
Figure 5-5: Interface for tagging long-term dependencies.

5.1.3   Other Tags

An arbitrary number of additional types of tags may be applied to a dataset, for use in biasing behavior at runtime. Two additional types of tags were applied to the TRG data, attitude tags, and domain-specific tags identifying utterances that refer to a specific course of the meal (appetizer, entree, or dessert). A modified version of the interface for tagging dependencies enables applying one of more tags to a node. Attitude tags identify utterances that humans consider to be: polite, rude, flirtatious, or drunk. These tags can be exploited by an attitude *Critic*, which is biased according to parameters configured by a designer, to influence next action selection so that an NPC tends to act rude when such an utterance exists. Otherwise, the system can fall back on untagged utterances.

The domain-specific tags provide additional semantic information required by the domain *Critic*, described in Section 5.3.5, to ensure the coherence of NPC dialogue by suppressing utterances related to ordering courses which have already been ordered or served. For example, a waitress should not ask if the customer wants dessert after he has already eaten a slice of pie. It might seem like the event structure should prevent such problems, however the fluid nature of human behavior and language leads to examples with a variety of structures -- in some cases customers order one course at a time, and in others they order all of their courses at once. There needs to be some means of bookkeeping outside of the event structure to keep track of which courses have been ordered or served, which in turn factors into selection of future utterances.

5.2   Learning Structure from Annotated Data

Human annotation described in 5.1 provides the building blocks necessary to learn hierarchies of discrete patterns representing events. Events, in turn, give context to utterances, facilitating semantic clustering by humans. This section describes how utterances are clustered, how a hierarchical *Event Dictionary* (ED) is learned from tagged data, and how the ED is employed to construct the *Event Log Index* (ELI). The ELI is a comprehensive index table which enables efficient retrieval of game logs at runtime, for purposes of plan recognition and case-based planning, based on matching discrete patterns representing events or event subsequences. Appendix B presents the script used to generate the ELI from game logs and annotations, with comments describing the various steps of preprocessing.

5.2.1   Manual Semantic Clustering of Utterances

Events are composed of nodes, where each node may be an action or utterance, intermixed arbitrarily. In order to store events as a discrete sequence of tokens, action can be tokenized as ACTIDs (introduced in Section 4.1.1), and a compatible representation is required for utterances. Manually grouping utterances into semantically meaningful clusters produces linguistic tokens that can serve as a common currency for combining with physical actions. This section describes how utterances are encoded, and grouped into sets of semantically meaningful clusters called *Utterance Sets* (U-SETs), stored in an *Utterance Library* (U-LIB).

Once logs have been annotated, all unique utterances included in events are extracted. Utterances perceived to serve the same purpose are dragged into a folder together. An annotator might drag "Hi"

and "Hello" into one folder, and "I'm ready for the bill" and "Check please" into another folder. Developing a browser-based tool for clustering utterances remains for future work. For the current study, I clustered utterances myself, using Windows Explorer as a surrogate user interface. Folders were named with the text from utterances, which could be dragged into parent folders to cluster them semantically with similar utterances. See Figure 5-6 for an example of clustering different ways of saying "hello." It took about two weeks of full-time work to manually group 18,907 utterances into 3,568 clusters. Manually grouping utterances is an alternative to more traditional dialogue act tagging, described in Section 4.3.1. The decision to take this more labor-intensive manual approach was based on the fact that manual clustering can be accomplished with minimal training or specialized knowledge, and allows for flexible, fine-grained groupings. Training a classifier to apply 3,568 different dialogue act labels would be difficult, especially due to the fact that many of these clusters have a small number of members, and the distinction between two clusters may be subtle.
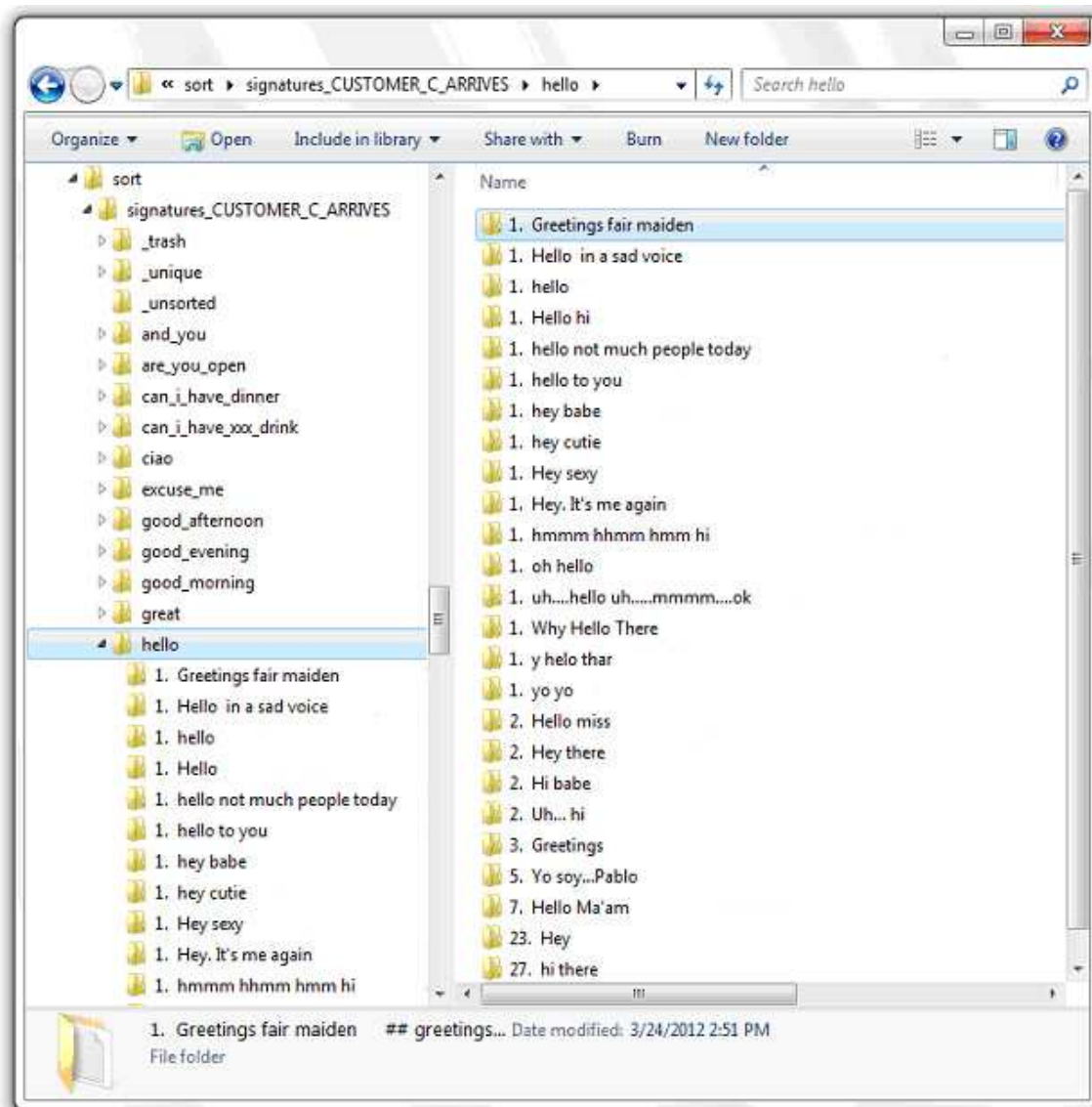
Figure 5-6: Manually clustering utterances using Windows Explorer.

Prior to clustering, utterances are encoded as keyword strings, and variables are collapsed based on a hand-crafted, domain-specific ontology (e.g. "Can I have steak?" and "Can I have salmon?" merge into "Can I have [FOOD]?"). Figure 5-7 illustrates a fragment of the ontology, with hierarchical concepts as branches, and labels observed in human utterances at the leaves. Note that leaves include misspellings (e.g. "salmon" vs. "slamon"). A keyword is any word observed in at least 25 game logs. All non-keywords are omitted from the encoded utterances. Encoded utterances grouped into the same folder form an *Utterance Set* (U-SET), and all U-SETs are stored in an *Utterance Library* (U-LIB). The U-LIB is implemented as a *Trie* data structure (Fredkin 1960), for efficient retrieval of the U-SET ID for any observed utterance.
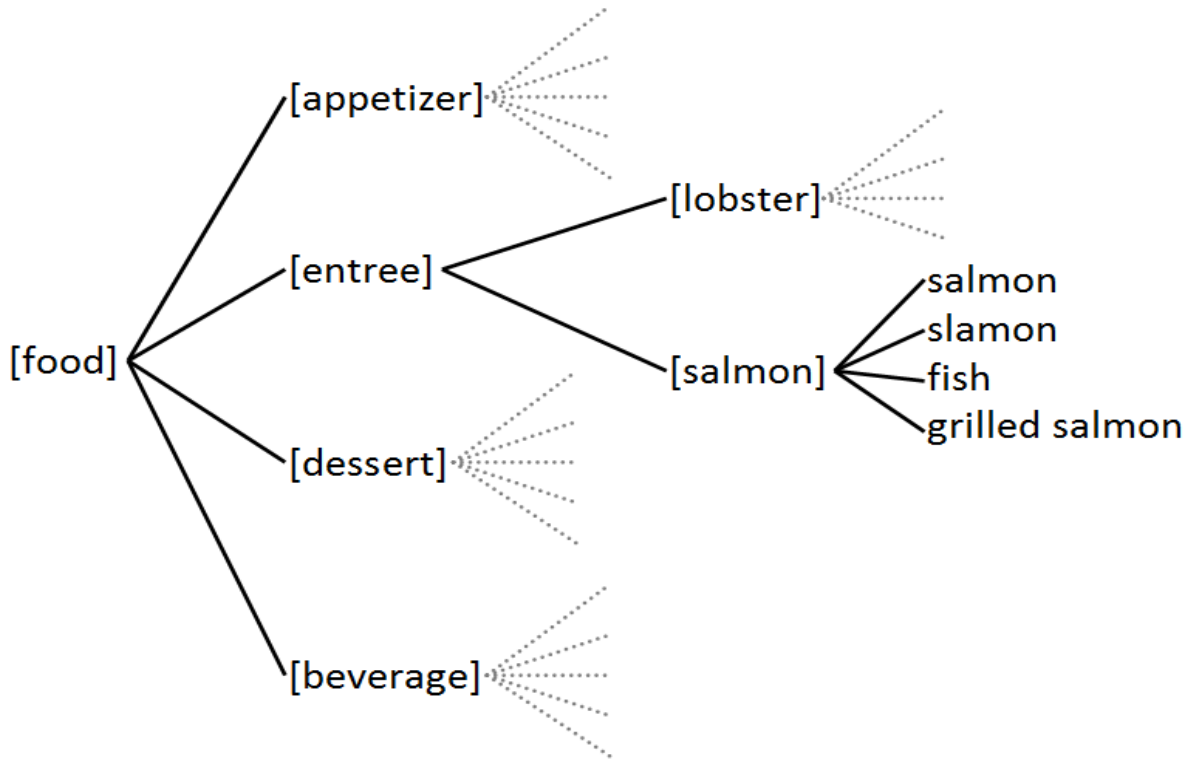
Figure 5-7: Fragment of manually constructed ontology of concepts, with human labels at leaves.

5.2.2   Learning the Event Dictionary and Event Log Index

The *Event Dictionary* (ED) contains discrete sequences, representing events, learned from the human annotations. Actions and utterances from the text-based logs are transformed into discrete tokens, ACTIDs indexing into the A-LEX, and U-SETIDs indexing into the U-LIB.

Using the A-LEX and U-LIB, logs are compiled into discrete sequences of time-coded ACTID and U-SETIDs. Time codes associate annotations with tokens in compiled logs, allowing extraction of each unique event pattern. Low-level events are stored in the *Event Dictionary* (ED) as sequences of ACTIDs and U-SETIDs. Higher-level events are stored as sequences of event start points. The annotated 1,000 game subset

contains 675 patterns for high-level events, and 9,460 for low-level events, composed of 1,414 unique ACTIDs, and 3,568 U-SETIDs representing 18,907 unique utterances. Figure 5-8 illustrates four different sequences in the ED to represent the event C_GETS_SEATED.
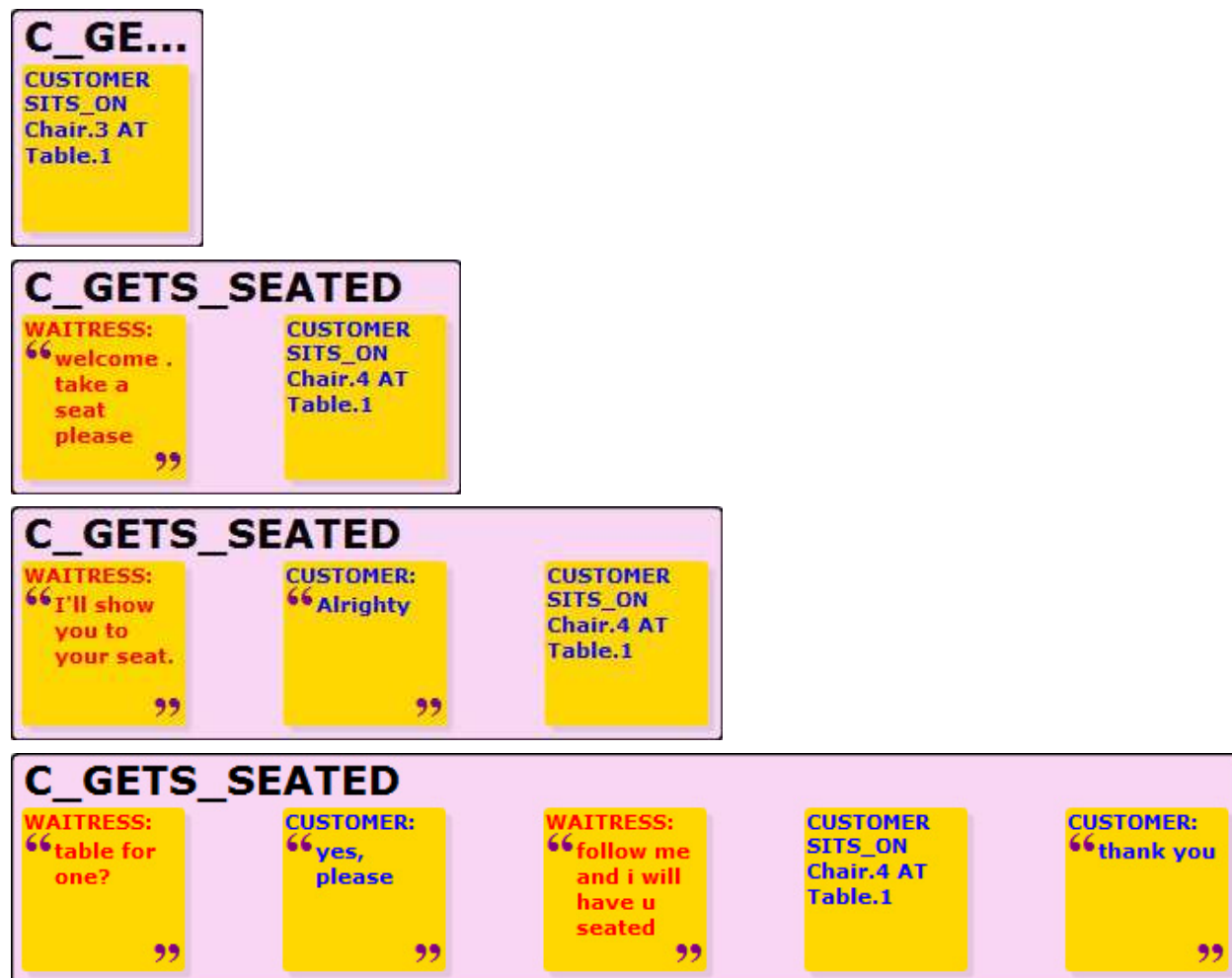
Figure 5-8: Four different sequences to represent C_GETS_SEATED in the Event Dictionary.

The compiled logs and ED are stored in the *Event Log Index* (ELI), along with a lookup table indicating the start points of event instances within log files. This table maps specific event patterns to instances within logs, allowing an agent to efficiently find logs that match observation sequences (or subsequences) at runtime. The ELI also stores associated meta-data, such as references and causal chains.

5.3 The Runtime Data-Driven Planning Architecture (RUN)

This section describes how NPCs select actions based on recent observations of interactions in the game world. NPCs are driven by a planner, which selects actions and utterances through a process that combines *plan recognition* (Kautz & Allen 1986) with *Case-Based Planning* (Hammond 1990). *Cases* refer to entire recorded episodes, in the form of annotated game logs. Each NPC observes actions and utterances, infers an event hierarchy (plan recognition), proposes games with similar event histories (case retrieval), and critiques proposals until one is found with a valid next action.
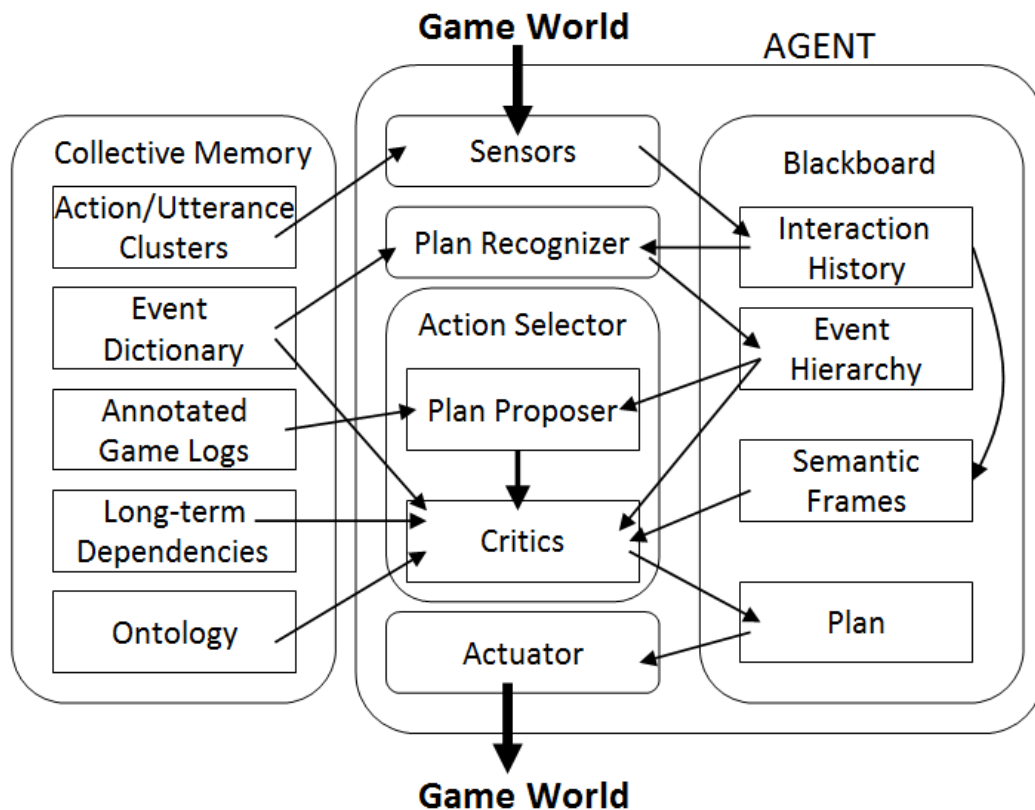
Figure 5-9: Diagram of the RUN planning architecture.

Figure 5-9 illustrates a high-level overview of the planning architecture (adapted from the architecture in Figure 4-3, with several differences). There is an agent on the CAI server associated with each NPC in the game world. Agents have access to *Collective Memory*, which stores the annotated game logs, as well as various data structures generated from these logs. The agent has a set of *Sensors* which receive observations from the game world. *Sensors* leverage information in *Collective Memory* to semantically cluster actions and utterances. Clustered observations are recorded in the *Interaction History* on the agent's *Blackboard*. The *Plan Recognizer* employs the *Event Dictionary* to infer an *Event Hierarchy* from the *Interaction History*. Next, the *Action Selector* selects the next action or utterance through a case-based planning process. The *Plan Proposer* retrieves game logs containing event hierarchies similar to

that stored on the *Blackboard*, and passes these proposals to a set of *Critic* processes. *Critics* exploit a variety of information to scrutinize these proposals, validating those expected to lead to a coherent next action. Assuming a proposal is found that gets approved by all *Critics*, the proposal is stored on the *Blackboard* as the current *Plan*. Finally, the *Actuator* executes the next action or utterance in the current *Plan* by sending a command to the game engine. Note that in some situations, the next action or utterance in the plan is actually taken by a different NPC, in which case it is considered an expectation, and the agent will wait to see if the expected action occurs before some expiration time.

5.3.1   A Motivating Example

This process becomes more clear with a concrete motivating example. Imagine an AI-controlled customer needs to respond to a waitress, recently observed to say "What can I do for you?" This is a sensible thing to say while working at a restaurant, however it is highly ambiguous. If the customer takes a naive approach, and simply searches for game logs that include identical, or similar, utterances, this may lead him to respond in a variety of ways: "Table for one please?", "Can I see the menu?", or "I'll start with the salad." Either response is equally likely, and the right choice depends on context.

Context can be formally represented as an event history. The history is hierarchical, and continually updated with each observation received since the beginning of the game. Each time the customer receives a new observation, he updates a mental model by adding or extending an event. Figure 5-10 illustrates the customer's mental model, which leads him to ask for salad.

BEGIN_DINING

C_ARRIVES
0: CUSTOMER_ENTERS
1: CUSTOMER: "Hello"

C_GETS_SEATED
2: WAITRESS: "Have a seat over here"
3: CUSTOMER SITS_ON chair04

C_GETS_MENU
4: WAITRESS GIVES menu01

C_ORDERS
5: WAITRESS: "What can I do for you?"
6: CUSTOMER: "I'll start with the salad"
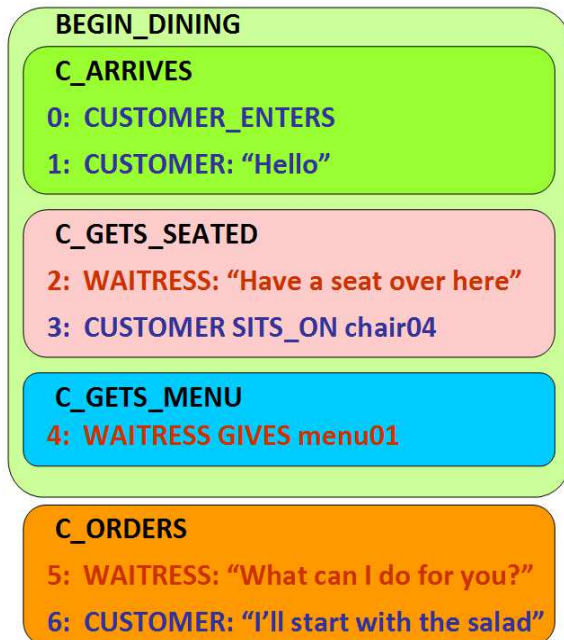
Figure 5-10: The customer's mental model after responding to "What can I do for you?"

The observation "What can I do for you?" is ambiguous, and could be incorporated into the event hierarchy in multiple ways. This utterance could begin a new C_GETS_SEATED, C_GETS_MENU, or C_ORDERS event. The customer tries each of these options.

Candidate game logs for each option are retrieved from *Collective Memory*, and passed to *Critic* processes to scrutinize. The *Critics* reject the candidates which propose to begin a new C_GETS_SEATED or C_GETS_MENU event, because BEGIN_DINING already includes both of these events, and extending BEGIN_DINING with either of these will add a duplicate, leading to an incoherent structure. In other words, *Collective Memory* does not contain any examples where someone got seated twice or got the menu twice while beginning dining, prior to ordering.

In contrast, based on *Collective Memory*, beginning a C_ORDERS event after a complete instance of BEGIN_DINING is valid. The customer is able to find a game log that begins C_ORDERS with "What can I do for you?" (or something clustered with the same U-SETID), and increments the pointer to the next action. This next action extends the C_ORDERS event, is validated by the *Critics*, and sent to the game engine for execution.

5.3.2   Integrating Planning with the Game Engine

For each NPC that exists in the world, an associated agent is running on an AI server (implemented in Java), networked with the game engine. As the game engine logs players' actions, resulting state changes, and utterances to a file, the engine broadcasts the same data over the network. The agent uses the A-LEX and U-LIB to process incoming data into discrete observations – ACTIDs and U-SETIDs. Agents process all observations through the same channel, regardless of whether they are associated with another player or the agent itself. Based on these observations, the agent makes decisions about what to do next, which are transmitted as commands for the NPC to execute in the game engine. The embodied NPC is responsible for all low-level behavior, including pathfinding, animation, and manipulating objects (e.g. CAI sends a command to a waitress to pick up a dish, and the NPC in the game world actually walks to the dish, and attaches it to her hand). This separation between high-level and low-level behavior leads to an arrangement where CAI is agnostic about the implementation of the low-level AI system for the NPC, thus CAI complements rather than replaces current approaches.

5.3.3   Plan Recognition

Plan recognition refers to the process of inferring an event hierarchy from a discrete sequence of observations. Sequences of observed actions and utterances are compared to sequences representing events in the *Event Dictionary* (ED), stored in *Collective Memory*. Low-level events are directly constructed from directly observable actions and utterances, while higher-level events are constructed from recognized low-level events. Multiple recognized sequences can overlap in time, and each sequence may contain gaps between observations.

In many instances, there is more than one possible event that matches an observed sequence (or subsequence). Plan recognition employs a case-based process for disambiguating events, and ensuring the inferred event hierarchy is coherent. In fact the same case-based system is employed for both plan recognition and next action selection (covered in the next section). The primary difference is that plan recognition uses case-based reasoning to explain observations, while action selection uses case-based planning to imagine a future observation, and ensure a coherent explanation exists for such an imagination.

As described in Chapter 2, *Case-Based Reasoning* (CBR) / *Case-Based Planning* (CBP) involves four steps:

(1) Case retrieval:
Retrieve cases from a corpus which are similar to the problem one is trying to solve.

(2) Ballpark solution proposal:
Propose one or more cases that could be used to solve the problem.

(3) Adaptation:
Adapt variables of the proposed case(s) to align the problem.

(4) Critique:
Scrutinize the adapted case(s) to validate whether a solution has been found.

Overview:

The agent tries to understand each new observation (an action or utterance) within the context of what it has observed previously. The *Plan Recognizer* infers how new observations extend the event hierarchy recognized so far. Plan recognition involves retrieving cases that can explain the most recent observation, in terms of how the observation extends an existing event, or adds a new event to the hierarchy. Cases are critiqued them until one can be found that is validated as a coherent explanation.

The application of CBR to CAI treats each entire recorded performance (stored as a game log) as a *Case*. Cases are indexed for retrieval by the events annotated within the game log. Each log may have many indices, allowing the agent to propose different entry points for using a particular action from a recorded performance to explain an observation. These event instances within the game log are referenced from the Event Log Index (ELI). The ELI allows the *Plan Recognizer* to efficiently retrieve game logs that match an observed sequence or subsequence.

Case Retrieval:

The event hierarchy is composed of token sequences representing events, some of which may be *incomplete*. Each token is a previously observed action or utterance, and a sequence is considered *complete* if it exactly matches a pattern in the ED. The agent is driven to try to complete sequences that are currently incomplete. Each new observation may (in order of preference):

        (1) Extend an incomplete sequence

        (2) Extend a subsequence of an incomplete sequence

        (3) Start a new sequence

        (4) extend a complete sequence.

Modified sequences that result from these options are referred to as candidate sequences. The *Plan Recognizer* iterates over candidates, and repeats the CBR process until a valid case is found. Each candidate sequence is used as the index to **retrieve** cases, which propose explanations for recent observations.

The ELI is used to retrieve all cases (annotated game logs) which contain an instance of an event that matches, or begins with, the candidate sequence. For example, if the agent has observed the sequence:

        WAITRESS: "What can I do for you?"   // U-SETID = 689
        CUSTOMER: "I'll start with the salad"   // U-SETID = 553

All cases will be retrieved that begin with a two-utterance sequence with the U-SETIDs 689, 553. Some of these cases will continue extending the event (e.g. the waitress asks "Would you like a drink with that?", or says "That will be right out"), in others, these two utterances form a complete instance of an event, and the log moves onto a new event immediately after (e.g. serving the food that was ordered).

Ballpark Solution Proposal:

One by one, each proposal is **applied** by truly extending the corresponding sequence in the hierarchy, or inserting a new sequence into the hierarchy. Initially, only the low-level event sequence is added or extended. Next, higher levels of the hierarchy are modified to match that in the proposal. For example, if the proposal begins a new C_GETS_SEATED event, and this event has the parent BEGIN_DINING in the proposed annotated game log, then the corresponding sequence added to the hierarchy in the *Plan Recognizer* also becomes a child of the BEGIN_DINING event. If the BEGIN_DINING event does not exist yet, it is created at this point. As the *Plan Recognizer* modifies the event hierarchy, it is responsible for all book-keeping which will allow these modifications to be reversed if the proposal is rejected by the *Critic* processes.

Adaptation:

Once applied, the sequence can be optionally **adapted**. CAI supports adaptation, allowing an action or utterance to swap an associated concept with another concept at the same level of the ontology described in Section 5.2.1. For example, a proposed game log where someone ordered fish can be adapted to explain an observed utterance where someone ordered steak. Currently, adaptation is used

sparingly in TRG, only applied to utterances for ordering food as just described. The discussion of adaptation in *Minstrel* and *Mayor* in Chapter 2 illustrates the complexities and potential pitfalls of adaptation. For these reasons, this work favors exploiting larger corpora rather than relying on clever adaptation. In other words, in many situations it is possible to reduce or eliminate the need for adaptation by providing alternate cases in corpora. However, some situations do exist where adaptation is necessary. For example, the corpus will never cover all possible ways of ordering food, due to combinatorial explosion of dishes and drinks ordered within the same utterance.

Critique:

The applied (possibly adapted) proposal is next sent to a set of **Critic** processes, to scrutinize the new or modified sequence (and parents). The same *Critics* are used in plan recognition and case-based planning. (*Critics* are described in detail in Section 5.3.4). A proposal that passes all of the *Critics* is considered valid. If any *Critic* rejects the proposal, the process repeats by applying the next proposal. If all proposals fail, the candidate is rejected, the modification is reversed, and the next candidate is considered if one exists. Once a candidate has been validated, the process is complete -- the observation has been recognized, and the modified structure of the hierarchy persists. If none of the candidates can be validated the observation is discarded as unrecognizable.

Whenever a sequence is added or extended, it requires an event label. Possible event labels for a candidate are determined by matching patterns in the ED. Labels are non-committal, and may be disambiguated as new information arrives. In Figure 5-11, the event beginning at node 5 is initially labeled as a C_ORDERS event, but later revised to a C_GETS_MENU once more information arrives in a subsequent observation.
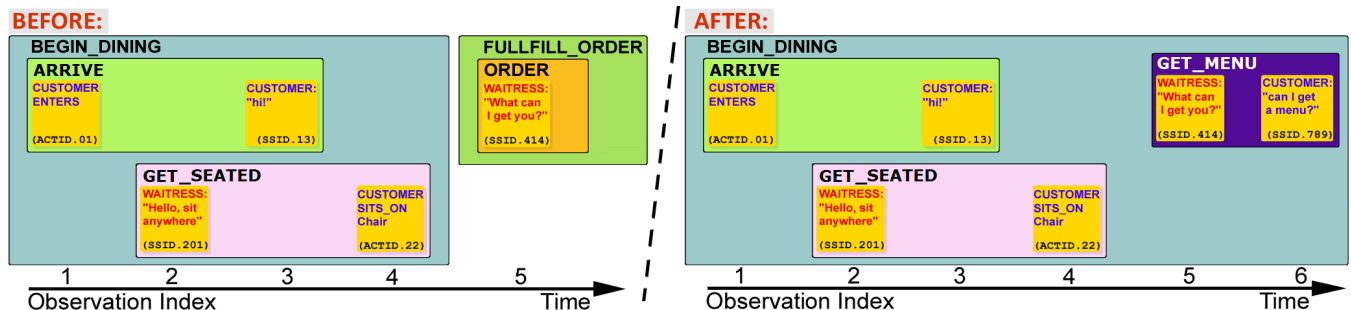
Figure 5-11: Label for event beginning at node 5 is revised as new information arrives.

Once an observation has been recognized, the agent selects the next action by either advancing the current plan, or searching for a new plan. A plan is a compiled game log, and the agent continues following the same plan as long as new observations continue to match the next token in the log, and are validated by the *Plan Recognizer*. If the observation does not match the next token, or is unrecognizable, the plan is invalidated and the agent re-plans, as described in Section 5.3.4.

114

5.3.4   Case-Based Planning

The previous section describes a case-based reasoning process for recognizing observations. This section explains how the CBR process is extended to support *Case-Based Planning* (CBP), where CBP is applied to selecting a coherent next action. CBP is an approach for selecting actions to respond to recent observations by finding examples of similar observations in the past. As described in Section 5.3.3, the application of CBP to CAI treats each entire recorded performance (stored as a game log) as a *Case*. cases are indexed for retrieval by the events annotated within the game log. Each log may have many indices, allowing the agent to propose different entry points for replaying from a particular action of a recorded performance.

Planning begins by iterating over a set of prioritized interaction *Goals*. *Goals* employ a variety of strategies to retrieve annotated game logs containing actions expected to move the interaction forward coherently. This may mean responding to something someone else has said or done, or pro-actively beginning a new event that is contextually appropriate. Using the ELI, *Goals* retrieve game logs and generate proposals – pointers into logs that begin after matching a particular sequence of tokens, or at the start of a specified type of event. Seven *Goals* have been implemented, detailed in Table 5-2. Appendix C presents the prioritization of *Goals* configured for TRG.

Goal	Description
GoalCompleteEvent	Find logs that contain the most recent incomplete sequence, and point to a subsequent token that can extend that sequence. Skip tokens that are part of other events.
GoalCompleteCausalChain	Find logs with events that could complete an initiated, but unresolved, causal chain (e.g. W_SERVES_FOOD or W_SERVES_DRINK if open orders exist).
GoalExtendCompletedEvent	Find logs that contain the most recently completed sequence, and point to the subsequent token that can extend that sequence. Skip tokens that are part of other events.
GoalExtendScenario	Find logs with events that could begin a contextually appropriate higher-level event.
GoalExtendStructure	Find logs with events that could extend an incomplete higher-level event.
GoalRespondToCompletedEvent	Find logs that contain the most recently completed sequence, and point to the subsequent token.
GoalRespondToSequence	Find logs that contain the most recently extended sequence, and point to the subsequent token.

Table 5-2: Description of Goals.

For each *Goal*, the agent iterates over the proposals, and re-runs the *Plan Recognizer*, treating the proposed next action as an imagined observation. Proposals with next actions that cannot be recognized are rejected. Remaining proposals must be validated by a set of *Critic* processes. *Critic* processes ensure future actions maintain coherence, with respect to past observations. Ten *Critics* have been implemented, detailed in Table 5-3. Planning is complete when a proposal is found that is approved by

all *Critics*, or when all *Goals* have been evaluated, and no valid proposal has been found. In the case of failure, the agent repeats the process, iterating the focus of attention backward in time to respond to earlier observations.

Critic	Description
CriticAttitude	If a desired attitude has been configured, this *Critic* biases behavior toward behavior that has been tagged with the specified attitude type.
CriticCausalChain	Prevents executing an action that completes a causal chain in a log file, if a corresponding chain has not been established in the currently running game, or has already been completed. For example, do not serve steak, if a steak was never ordered, or has already been served. This *Critic* relies on tagged Causal Chain meta-data.
CriticDomain	Prevent executing an action for domain-specific reasons. More details about domain knowledge in Section 5.3.5.
CriticInvalidatedAction	Prevents repeatedly trying to execute an action that the game engine reports has failed.
CriticNoReruns	Prevents repeating the exact same utterance from the same game log more than once in the same game.
CriticReference	Prevents executing an utterance that refers to something that has never been observed in the current game. This *Critic* relies on tagged Reference meta-data.
CriticRequiredRole	Prevents waiting for someone else to execute an expected action. After an agent has failed to execute an action, this *Critic* forces the same agent to try to take a different action as soon as possible.
CriticReservedLog	When multiple NPCs are interacting (as opposed to a human-NPC interaction), this *Critic* prevents both agents from locking onto the same log file, which would result in a replay rather than a dynamic interaction. Each time an agent executes an action, it reserves the action to prevent the other agent from selecting it in the future.
CriticResourceConflict	Prevents trying to execute an action that requires a resource already in use. For example, a waitress should not plan to pick up a menu if there is already a dirty dish in her hand. This critic will encourage her to return the dirty dish to the kitchen before pursuing giving the customer a menu.
CriticStaleDialogue	Prevents continuing a dialogue many steps later, if the dialogue could already be considered complete. This reduces the occurrence of utterances that seem to be non-sequiturs.
CriticStructure	Prevents adding low-level events which create invalid structure at higher levels of the hierarchy, according to the Event Dictionary.

Table 5-3: Description of Critics.

Critics can be configured as *required* or *preferred*, where preferred *Critics* can be ignored if no valid proposals can be found to satisfy them. For example, if no proposals can be found to satisfy a specified attitude type, *CriticAttitude* can be ignored. In addition, there is a notion of pressure that rises over time, and *Critics* may optionally specify a maximum pressure, above which they are ignored. This ensures that an NPC will eventually do something if enough time has passed. Appendix C presents the prioritization of *Critics* configured for TRG.

5.3.5   Domain Knowledge

Critics are domain independent, with one exception -- *CriticDomain*, a special *Critic* which can invalidate a proposal based on domain-specific information. Each event type may optionally have a hand-encoded validation function, which constrains when that type of event may be extended, or introduced into the hierarchy. These functions contain arbitrary Java code, and are stored in the *Domain Knowledge Manager*. Hand-crafting domain knowledge, in an otherwise data-driven system, is a concession required for two reasons: (1) sparse data, and (2) discrepancies between recorded human behavior, and what we desire to see from NPCs.

A concrete example illustrates both of these motivations. Imagine a customer has finished eating a salad, salmon, and cherry cheesecake, and an AI-controlled waitress needs to decide what to say next. With a wide variety of dishes on the menu, there is no guarantee that a recorded game log exists that matches the dishes ordered exactly. Abstracting dishes in a domain-specific way -- into courses (appetizer, entree, dessert) -- compensates for sparse data, and domain knowledge associated with C_ORDERS leads the waitress's *CriticDomain* to invalidate initiating events from game logs where the history of consumed courses does not match what has been observed. Furthermore, there may be examples of human-human game logs where the waitress continues to offer dessert after dessert. An AI-controlled waitress who offers dessert after serving dessert is interpreted as buggy by humans, even if she is imitating an actual recording of playful human behavior. *CriticDomain* can leverage hand-crafted domain knowledge to constrain an NPC to behavior desired by designer, even if humans have broken these rules. Programmatically restricting these behaviors allows them to remain in the corpus for recognition, without risk of execution.

Domain knowledge is also used to perform adaptation of proposed cases during the CBP process, as mentioned above in Section 5.3.4. Each event type has an optional associated adaptation function. For example, domain knowledge for the C_ORDERS event includes a function that detects when an utterance from an annotated game log begins a causal chain, indicating that someone is ordering a food or drink. In these cases, the function replaces the item(s) mentioned in the utterance from the game log with the items mentioned in the associated utterance in the actual interaction. The ontology is employed to recognize words for food and drink items within utterances.

In the current implementation, encoding domain knowledge is a time-consuming iterative process (iterating with testing). Future work should focus on formalizing the encoding of rules from ad hoc Java code to something more principled and more strictly structured. The expectation is that this first implementation will reveal design patterns in rules, which can be formalized into event configuration files, rather than procedural code. In addition, Section 5.5 suggests ways to simplify debugging, and preemptively detect problems.

5.3.6   High-Level Goals & Directability

Automating NPCs from a large corpus of human performances produces digital actors who can improvise and adapt their behavior, in response to interactions in the virtual world. While this autonomy helps support a player-directed experience, it may also be desirable for the designer to be able to direct these NPCs at a high-level to behave in some specified way. For example, a designer may wish for an NPC to dramatize a scene exhibiting a particular attitude (e.g. a rude waitress). The planning architecture provides two mechanisms that can be used to direct an NPC with high-level goals that bias his or her behavior in some desired way. Specialized *Critics* can be introduced which attend to specific layers of meta-data, and domain knowledge for specific events can be augmented to support some high-level goal.

Two high-level goals have been implemented. The first biases an NPC to pursue a specified attitude. The second encourages an NPC waitress to upsell. The attitude bias in the simpler of the two to implement, though upselling is only slightly more complex. In order to bias an NPC toward a specified attitude, game logs are annotated with tags indicating actions and utterances that exhibit attitudes such as polite, rude, and flirtatious. When a desired attitude is specified for an NPC (via a configuration file), the attitude *Critic* rejects proposals for actions and utterances that are not tagged with the specified attitude. If no valid proposals are found, on the second pass the attitude *Critic* only rejects actions and utterances tagged with a different attitude, allowing untagged actions and utterances to be proposed as defaults. Upselling combines an attitude tag for upselling with a small modification to the domain knowledge for the W_SERVES_FOOD, W_SERVES_DRINK, W_PREPARES_BILL events. Validation functions for these events are modified such that the waitress will not begin serving an entrée until an appetizer has been ordered, and will not bring the bill until dessert has been served. These modifications combined with the upsell attitude tag, encourage the waitress to continue extending the C_ORDERS event, producing additional utterances that ask the customer if he would like to order more food, expressed in different ways. Ideally these restrictions would eventually time-out if the customer never orders more items, but a time-out is not currently implemented, producing a very persistent waitress. The next section includes transcripts illustrating the effect of these high-level goals on human interaction.

5.4   Human Interaction

Human players can type text or speak into a microphone to converse with NPCs. Rather than passing the natural language input directly an NPC, the input is intercepted, and used to dynamically generate a list of contextually relevant dialogue options, as semantically similar as possible to the human input.

Figure 5-12: Intelligent interface compensates for challenges in natural language understanding.

Presenting dialogue options is a practical approach to alleviate the difficult problem of natural language understanding (NLU). NLU is a challenge for reasons of coverage and context. The enormous potential variation in open-ended spontaneous natural language input makes it likely that any future human-NPC game will encounter utterances never previously observed in the recorded game logs. While many of the words may be familiar, unfamiliar ordering, or the introduction of out-of-vocabulary words, may lead to an utterance with a completely different meaning from its closest match in the corpus. Furthermore, an NPC may misunderstand a familiar utterance used in an unfamiliar context. Finally, speech recognition adds additional uncertainty, as the system cannot be entirely sure that the words were recognized correctly. Presenting the player with dialogue options makes the process transparent, and increases the likelihood that the player will be able to move the narrative forward as intended.[9] In

---

[9] There is evidence from the commercial game industry that making intelligent AI systems transparent to the player improves usability. Black & White (Lionhead 2001) featured creatures who learned from positive and negative reinforcement, but it was not always clear to the player what the creatures was learning, and sometimes the creature could learn the wrong things. In the sequel, Black & White 2 (Lionhead 2005), critics and press responded positively when the developers made the learning system transparent, so that players could see what was being learned, and intercept and correct the creature when it learned incorrectly.

this respect, NLU is handled in much the same way as a search engine, which returns multiple query results, sorted by relevance. The current interface, in the figure above, is a rough first implementation. One can imagine more polished interfaces. Perhaps the interface could automatically choose the top option in the list, and give the user one second to hit a button to intercept if options are desired, or perhaps the interface could present options in real-time as the user types, similar to Google's auto-complete in search engine queries. There are many possibilities for interfaces that could operate on top of the foundation laid by an underlying CAI system.

The following figures present transcripts from actual interactions between a human customer and an NPC waitress. Each transcript highlights a different configuration for interaction. In these transcripts, the utterances recorded for the customer represent the final dialogue option selected by the human player, based on typed text input (not shown). All dialogue options come from recorded human performances, including typos, misspellings, and colloquialisms. Figure 5-13 is a transcript with a default configuration, while the next two transcripts capture interaction with waitresses directed to achieve high-level goals of being rude and upselling respectively, using the approach described in Section 5.3.6. The transcript from the default configuration includes examples of the NPC responding to out of the ordinary behavior, such as sitting on top of the table and stealing the cash register. This transcript also contains two utterances that were selected based on the human input of "can I have," illustrating that the system provides different options based on context. At the start of the game, "can I have" produces options asking for a menu or a table. After sitting and eating some food, "can I have" produces options asking for the next course, a drink, or the bill. Clicking through several screens of options eventually leads to the more unusual option "can I have a microwave."

```
CUSTOMER: "Good evening, miss"
WAITRESS: "Good evening sir"
CUSTOMER: "can I have a table?"
WAITRESS: "sure follow me"

        CUSTOMER SITSON table1(Table)

WAITRESS: "SIR do not sit on the table!"

        CUSTOMER STANDSUP FROM table1(Table)
        CUSTOMER SITSON chair1(Chair)
        WAITRESS PICKSUP dyn026(Menu)

WAITRESS: "a menu sir"
CUSTOMER: "THANKS"

        WAITRESS GIVES dyn026(Menu) TO CUSTOMER

WAITRESS: "drink?"
CUSTOMER: "no"
CUSTOMER: "so whats good?"
WAITRESS: "another restaurant"
WAITRESS: "no but I would go for the salmon"
CUSTOMER: "I'll take it"
WAITRESS: "ok"

        CUSTOMER PUTSDOWN dyn026(Menu) ON table1(Table)

WAITRESS: "salmon"

        dyn087(Salmon) APPEARS ON counter2(Counter)
        WAITRESS PICKSUP dyn087(Salmon)
        WAITRESS PUTSDOWN dyn087(Salmon) ON table1(Table)
        CUSTOMER EATS dyn087(Salmon)
        CUSTOMER EATS dyn087(Salmon)
        CUSTOMER EATS dyn087(Salmon)

CUSTOMER: "can I have an microwave?"
WAITRESS: "yes..."
WAITRESS: "ok lol"

        WAITRESS PICKSUP microwave(Microwave)
        WAITRESS GIVES microwave(Microwave) TO CUSTOMER

CUSTOMER: "thank you!"

        CUSTOMER PUTSDOWN microwave(Microwave) ON table1(Table)
        CUSTOMER STANDSUP FROM chair1(Chair)
        CUSTOMER PICKSUP cash_register(Register)

WAITRESS: "a robbery!"

        CUSTOMER EXITS RESTAURANT
```

Figure 5-13: Transcript from actual interaction between human customer and NPC waitress.

```
CUSTOMER: "hello"
WAITRESS: "welcome to the hell hole"
WAITRESS: "SIT BOY"

        CUSTOMER SITSON chair1(Chair)

WAITRESS: "what you want"
CUSTOMER: "What are the specials?"
WAITRESS: "specials? read 'em on the wall"
CUSTOMER: "Ill have the salmon"
WAITRESS: "Ill see what I can do"
WAITRESS: "fish"

        dyn022(Salmon) APPEARS ON counter3(Counter)
        WAITRESS PICKSUP dyn022(Salmon)
        WAITRESS GIVES dyn022(Salmon) TO CUSTOMER
        CUSTOMER PUTSDOWN dyn022(Salmon) ON table1(Table)
        CUSTOMER EATS dyn022(Salmon)
        CUSTOMER EATS dyn022(Salmon)
        CUSTOMER EATS dyn022(Salmon)

CUSTOMER: "bill please"
WAITRESS: "HANG ON"

        WAITRESS TOUCHES cash_register(Register)
        dyn023(Bill) APPEARS ON podium(Podium)
        WAITRESS PICKSUP dyn023(Bill) FROM podium(Podium)

WAITRESS: "pay it!"

        CUSTOMER PICKSUP dyn023(Bill)
        CUSTOMER LOOKSAT dyn023(Bill)
        CUSTOMER PAYS dyn023(Bill)
        CUSTOMER GIVES dyn023(Bill) TO WAITRESS

WAITRESS: "You cheapskate"
WAITRESS: "No tip!"

        WAITRESS PUTSDOWN dyn023(Bill) ON cash_register(Register)

WAITRESS: "out you go"

        CUSTOMER STANDSUP FROM chair1(Chair)

CUSTOMER: "bye"
WAITRESS: "bye"

        CUSTOMER EXITS RESTAURANT
```

Figure 5-14: Transcript from interaction between human customer and waitress directed to be rude.

WAITRESS: "can I get you an appetizer or something to drink?"
CUSTOMER: "Ill just have the steak"
WAITRESS: "anything else?"
CUSTOMER: "no thank you"
WAITRESS: "salad, soup, desert"
CUSTOMER: "Ok Ill start with the salad"
WAITRESS: "would you like some dessert too?"
CUSTOMER: "no thank you"
WAITRESS: "salad"

       dyn156(Salad) APPEARS ON counter1(Counter)
       WAITRESS PICKSUP dyn156(Salad)
       WAITRESS PUTSDOWN dyn156(Salad) ON table1(Table)
       CUSTOMER EATS dyn156(Salad)
       CUSTOMER EATS dyn156(Salad)
       CUSTOMER EATS dyn156(Salad)

WAITRESS: "steak"

       dyn157(Filet) APPEARS ON counter2(Counter)
       WAITRESS PICKSUP dyn157(Filet)
       WAITRESS PUTSDOWN dyn157(Filet) ON table1(Table)
       CUSTOMER EATS dyn157(Filet)
       CUSTOMER EATS dyn157(Filet)
       CUSTOMER EATS dyn157(Filet)

WAITRESS: "can I get you a dessert"
CUSTOMER: "no just the bill please"
WAITRESS: "we have a great berry pie"
CUSTOMER: "can I get the bill?"
WAITRESS: "can I interest you in dessert? The cheesecake is great"
WAITRESS: "The cheesecake is wonderful"
CUSTOMER: "Ok I think I'll have cheesecake"
WAITRESS: "cheesecake"

       dyn158(Cheesecake) APPEARS ON counter2(Counter)
       WAITRESS PICKSUP dyn158(Cheesecake)
       WAITRESS PUTSDOWN dyn158(Cheesecake) ON table1(Table)
       CUSTOMER EATS dyn158(Cheesecake)

CUSTOMER: "check please"
WAITRESS: "will that be all?"
CUSTOMER: "yes"

       WAITRESS TOUCHES cash_register(Register)
       dyn159(Bill) APPEARS ON podium(Podium)
       WAITRESS PICKSUP dyn159(Bill) FROM podium(Podium)
       WAITRESS GIVES dyn159(Bill) TO CUSTOMER
       CUSTOMER LOOKSAT dyn159(Bill)
       CUSTOMER PAYS dyn159(Bill)
       CUSTOMER GIVES dyn159(Bill) TO WAITRESS

WAITRESS: "Come back again"
CUSTOMER: "THank you"
WAITRESS: "you bet"

       CUSTOMER STANDSUP FROM chair1(Chair)

CUSTOMER: "bye"

       WAITRESS PUTSDOWN dyn159(Bill) ON cash_register(Register)
       CUSTOMER EXITS RESTAURANT

Figure 5-15: Transcript from interaction between human customer and waitress directed to upsell.

The interface for human interaction re-uses the previously described planner. When a human controls a player, there is still an associated agent running in the background on the AI server. This agent passively runs plan recognition, but does not perform action selection. When the human interacts in the game world, physical actions are broadcast as usual. Utterances, on the other hand, are flagged for further processing by the human's agent, and are ignored by the NPC's agent. Human utterances originate from the text output of the Windows speech recognizer, running with a language model generated from the TRG corpus.

The human's agent is responsible for selecting a list of dialogue options from the corpus, semantically similar to the flagged human utterance. The agent begins by pruning non-key words from the utterance. Next, the agent retrieves a list of U-SETIDs from the U-LIB for all utterance sets that include an utterance containing the key words. The *Plan Recognizer* generates candidate sequences for all U-SETIDs that can be recognized as the next action. The agent then iterates over the candidates, applies each, and retrieves proposed plans from the ELI (which point to the U-SETIDs as the next action). Finally, the agent runs the critique process, but rather than stopping at the first approved proposal, the agent continues critiquing, collecting a list of all approved proposals.

If this process fails to generate at least five proposals (possibly zero if speech recognition fails), the agent uses context to compensate for failure to understand. In this case, the agent falls back to action selection driven by interaction *Goals*, like an NPC, as described in the last section. All proposals from all *Goals* are collected that are not rejected by *Critics*.

C: Can I get table for one?
W: Sure right this way
W: would you like a drink to start?
C: can I get some water please
W: ok
W: would you like more time or
    are you ready to order?
C: can I have a menu?

. . .

W: our specials tonight are salmon,
    a vegtable soup, and a nectari
C: ill have grilled salmon
    please

Actual transcript excerpt and
dialogue options from
human customer with
waitress NPC.
Selected utterance in **bold**.

Human said: "can I have a menu please?"
Speech Rec heard: "banana nine yep"

Dialogue Options:
- "can ican i have a cobb salad please"
- "yes nectarine tarts sound good for desert"
- "maybe I should just start with the soup"
- "can I have another water, please?"
- **"can I have a menu?"**
- "I'll have the lobster and cheesecake"
- "and with that some spaghetti"
- "May i please have the Filet please?"
- "I'll have the... uh seafood I mean, salmon"
- "may i get the soup of the day please"

Human said: "I'll have grilled salmon"
Speech Rec heard: "of grilled salmon"

Dialogue Options:
- "may i get the grilled salmon of the day please"
- "yes the grilled salmon"
- "a grilled salmon please"
- **"ill have grilled salmon please"**
- "and with that some grilled salmon"
- "can have grilled salmon and soup"
- "Ill start with the grilled salmon"
- "beery pie beer and grilled salmon"
- "I think grilled salmon sounds lovely"
- "can i have some grilled salmon?"

Figure 5-16: Top 10 dialogue options found for player inputs.

The agent now has a list of proposals for utterances deemed valid by the *Plan Recognizer* and *Critics*. Proposals are sorted to best match the human input, discussed further below. The top five utterances are sent to the game engine, for display to the player as dialogue options (Figure 5-16 shows the top 10). The human can repeatedly click SHOW ME MORE to retrieve the next five options, or CANCEL to abort if

none of the options are satisfactory. When the human selects a dialogue option for execution, the selected utterance is broadcast as an ordinary unflagged utterance, for processing by agents through ordinary channels.

A number of factors complicate retrieval and sorting of dialogue options, making it a less straight forward process than one might expect. The system can only present the player with five dialogue options at a time (limited screen real estate), and ideally the options at the top of the list will be as close as possible to the human input, both in terms of surface form and meaning. The best-case scenario is when the words of the human input directly map to a U-SET, where retrieval is simply a process of collecting all utterance proposals with the specified U-SETID(s) deemed valid by all *Critics*, and sorting them to maximize overlap with the input. In this case, sub-optimal dialogue options can result from exceeding limits placed on search time for valid proposals. It is possible that the best matching utterance in the corpus will not be found within a reasonable time, leading to dialogue options that share the same meaning, but do not match the words quite as well. Bigger problems arise when speech recognition fails, and words are misrecognized or not recognized at all, leading the system to rely on retrieval by context. In these cases, rather than optimizing the sorting to best match human input, sorting is optimized to increase the likelihood that a desirable option will appear on the first page, or soon after. This is accomplished by interleaving proposals associated with different event types, so that each page of dialogue options offers a variety of utterances with different purposes.

There are numerous tradeoffs to consider when selecting dialogue options, and further experimentation remains for future work. The current system displays multiple utterances of the same U-SETID, often producing redundant options. Addressing this issue is complicated by the fact that concepts are clustered when encoding utterances (see Section 5.2.1). For example, when retrieving proposals by context, it may be desirable to display multiple utterances for ordering food, if each utterance refers to different menu items. In addition, all words are currently weighted equally. It might improve sorting if words that are unique to an event or U-SETID are weighted more heavily.

5.5   Best Practices

Data-driven planning is a non-traditional approach to automating NPCs, which requires rethinking both how games are authored, and how they are debugged. The experience of automating NPCs in TRG has led to some insights about how best to approach this development process in the future. Below are some suggestions for tools and features that future developers of similar systems might consider.

Integrated Search Tools: Bugs in behavior and dialogue may result from problems in the domain knowledge code, or incorrect / inconsistent annotation of the data. When issues exist in the annotation, it is likely that the same issue exists in other game logs within the corpus. It is important to have tools in place to search for all instances of a tagging pattern (e.g. sequence of actions tagged as some type of event), generate human comprehendible reports, and are ideally integrated with the annotation tools to streamline applying corrections. Currently, searching the corpus for instances of tagging patterns requires hand-crafting scripts which generate reports of search results. A tool exists to convert plain text-based reports into HTML, with links from reported instances to the original data files containing the potential tagging errors. Design patterns have emerged for frequently required types of searches. Ideally, rather than writing scripts, a configurable tool would exist for different types of searches, integrated with other tools for viewing instances in the appropriate authoring tool for making tagging corrections.

Pre-emptive Debugging: Some annotation errors may be identifiable without running the game at all due to statistical anomalies. It is possible to generate reports of action sequences that are likely mislabeled by comparing recurrence counts of sequences within different event labels. While this will generate some false positives, the cumulative debugging time saved is worthwhile. Again, the existing tool for pre-emotively searching for tagging errors requires writing hand-crafted scripts to search for different types of errors, and ideally could evolve into a configurable tool.

Synthetic Data Generation: If the planning system can play all of the NPCs in the game, it is possible to generate game logs by running the system repeatedly without any human players. Hiring people online to inspect human-readable transcripts of these games is a means of scaling up the quality assurance process. There are some bugs that will never be exposed without human players, but fixing issues observed in NPC-NPC games first can maximize the debugging gains from human tests. The current system was debugged for several months with NPC-NPC interactions prior to paying human testers, eliminating many bugs before consuming valuable human cycles.

Replay System: One of the features of a system driven by thousands of recorded game logs is that the same scenario can play out differently every time. This can make debugging an observed problem difficult. It is extremely helpful to have a system that can generate a script from game log, and replay the exact same sequence of decisions. It may also be useful to be able to begin a replay from any step in a previously recorded game. Currently a replay system does exist, which proved critical for debugging. However, the current replay system could be improved by enforcing not only the game logs used to generate actions, but also the game logs that the system uses to understand observations. The CAI system ties all observations to a portion of a game log in the corpus, and the same sequence of observations may be interpreted differently depending on which log is bound to a sequence of observations. Forcing an NPC to understand observations using particular logs will better guarantee that the game plays out the same every time.

6 Evaluation

A two-part evaluation was designed to answer the questions: 1) Does the system robustly support open-ended interaction, and 2) Do players notice the increased freedom, and does it enhance their gameplay experiences? The first question is addressed through a quantitative evaluation, while the second is explored qualitatively via a focus group.

6.1 Quantitative Evaluation

The quantitative evaluation measures how well the system responds to the player, given an extremely challenging problem -- open-ended interaction with speech input. Speech adds additional complications to the already difficult problem of natural language understanding (NLU). Not only does the system need to be able to put utterances into the correct context, and handle out-of-vocabulary words, but it must also compensate for the uncertainty of speech recognition, where words may be misrecognized (faulty information), or the speech recognizer may fail to recognize any words at all (no information).

6.1.1 Experimental Setup

Evaluation data was collected from 20 people (with no previous exposure to TRG) playing as a customer, using speech to interact with an NPC waitress. The waitress behavior, and customer dialogue options were driven by 1,000 recorded games. These games were randomly selected from the corpus, and annotated through the process described in Chapter 5. Subjects were divided into four groups of five, each playing under one of four conditions for populating the list of dialogue options:

(1) Text+Context:

Text+Context refers to the system described in Section 5.4, which selects U-SETIDs based on key words from typed input, and falls back to interaction *Goals* to compensate for failure to find valid proposals. Text+Context simulates perfect speech recognition, and provides an upper-bound on how well this system works if speech recognition never fails at recognizing words (because the typed words are directly observable).

(2) Speech+Context:

Speech+Context works the same way as Text+Context, using the words returned by the speech recognizer as input.

(3) Speech-Only:

Speech-Only presents a sorted list of all utterances in the corpus that match any of the words in the speech input, without using the *Plan Recognizer* or *Critics* for filtering.

(4) Context-Only:

Context-Only completely ignores human input, and only relies on the inferred event hierarchy and interaction *Goals* to select the list of relevant utterances.

Each subject played through one entire game as a customer, interacting with an NPC waitress. Subjects were given a brief tutorial of the game controls, and then were simply told to go have dinner. The game begins when the customer walks through the door of the restaurant, and ends when the customer exits. In between, interaction is open-ended, as the customer gets seated, consumes a meal (with a variable number of dishes and drinks), and pays a bill. Each session took about 10-15 minutes.

Every time the customer spoke to the waitress (via typed text or speech, depending on condition), the subject was presented with a list of dialogue options, populated according to condition as described above. The subject was shown five dialogue options, and asked to select the option that best matches the meaning of what s/he was actually trying to say. If none of the options were satisfactory, the subject could repeatedly click SHOW ME MORE to see the next five options, cycling back to the start of the list if the subject reached the end of the list.

The total number of options per input varied depending on condition and input. If the subject failed to find any satisfactory dialogue option, s/he could choose to click CANCEL to abort the interaction, and either try the input again, do something else, or wait for the NPC to do something. For each input, the subject was asked to mark the first 10 dialogue options, to indicate *relevant* options, which the subject felt had same meaning as what s/he was actually trying to say. In the Speech-Only condition, when the speech recognizer failed completely, the subject was given only a failure message, and CANCEL, due to lack of any other means to select dialogue options, given no input words. In the other conditions, the system could fall back to selecting options by context when speech recognition failed.

A variety of different kinds of data was generated from transcripts of these gameplay sessions:

Selection Rank:

> For each input in which the subject was able to find a satisfactory dialogue option, the system recorded the selection rank of the option actually selected. The dialogue option at the top of the list is rank 1, the next option is rank 2, and so on. Thus, lower rank is better, indicating that the subject was able to find the desired option sooner. Dividing the rank by five determines how many pages the subject needed to click through before finding the desired option. The selection rank is undefined for inputs where the subject clicked CANCEL.

Selection Rank Likelihood:

> Separately for each condition, based on the cumulative selection ranks across all inputs for all subjects, the likelihood that a selected dialogue option will be of rank N or higher was computed for N between 1 and 25.

Input Option Count:

> For each gameplay session, the input option count indicates the total number of typed text or speech inputs made by the subject.

Relevant Option Count:

> For each gameplay session, the relevant option count indicates the total number of options the subject marked as relevant, out of the top 10 options per input.

Plan Recognition Failure Count:

> For each gameplay session, the plan recognition failure count indicates the number of times the subject selected a dialogue option, and the NPC could not understand the utterance. This can only occur in the Speech-Only condition, because the other conditions filter out options deemed contextually irrelevant by the system.

Speech Recognition Failure Count:

> For each gameplay session, the speech recognition failure count indicates the number of times the speech recognizer failed completed, and no words were recognized from the input. Speech recognition failure is external to the implemented CAI system (and is a failure of the off-the-shelf speech recognizer), but is measured to show that the number of recognition failures is comparable in both speech conditions, ruling out a possible confound in the results.

6.1.2 Results

The number of speech inputs varies per game. The evaluation looks at the first 10 inputs in each game, 50 total per condition, for a fair comparison. The following four tables present the raw data associated with the 50 inputs for each condition. The Text+Context and Context-Only conditions did not employ speech recognition, thus the ASR Recognized column specifies "N/A." For inputs where the subject clicked CANCEL to abort, the "Selection Rank" is specified as "--".

Input Index	Selection Rank	ASR Recognized	Num. Options	Num. Relevant
1	3	N/A	10	10
2	1	N/A	10	10
3	7	N/A	10	6
4	--	N/A	10	10
5	3	N/A	10	10
6	1	N/A	10	7
7	2	N/A	10	2
8	--	N/A	10	0
9	1	N/A	10	9
10	1	N/A	10	10
11	--	N/A	10	0
12	4	N/A	10	7
13	18	N/A	10	3
14	1	N/A	10	8
15	1	N/A	10	3
16	--	N/A	10	0
17	8	N/A	10	10
18	2	N/A	10	7
19	1	N/A	10	1
20	1	N/A	10	10
21	1	N/A	10	7
22	1	N/A	10	7
23	1	N/A	10	1
24	2	N/A	10	7
25	1	N/A	10	1
26	1	N/A	10	9
27	2	N/A	10	9
28	1	N/A	10	8
29	4	N/A	10	5
30	1	N/A	10	1
31	1	N/A	10	1
32	1	N/A	10	3
33	2	N/A	10	0
34	3	N/A	10	9
35	8	N/A	10	0
36	9	N/A	10	3
37	1	N/A	10	6
38	3	N/A	10	8
39	1	N/A	10	10
40	1	N/A	10	1
41	1	N/A	10	10
42	3	N/A	10	10
43	1	N/A	10	9
44	1	N/A	10	10
45	--	N/A	10	1
46	1	N/A	10	7
47	3	N/A	10	10
48	3	N/A	10	9
49	1	N/A	10	8
50	2	N/A	6	6

Table 6-1: Raw data for the Text + Context condition.

Input Index	Selection Rank	ASR Recognized	Num. Options	Num. Relevant
1	4	FALSE	10	1
2	1	TRUE	10	10
3	1	FALSE	10	3
4	1	TRUE	10	10
5	1	TRUE	10	9
6	7	FALSE	10	4
7	19	FALSE	10	0
8	13	FALSE	10	0
9	2	TRUE	10	6
10	1	TRUE	10	5
11	1	FALSE	10	3
12	1	TRUE	10	8
13	--	TRUE	10	0
14	--	FALSE	10	0
15	--	TRUE	10	0
16	1	TRUE	10	3
17	6	TRUE	10	3
18	23	FALSE	10	1
19	1	TRUE	10	4
20	--	FALSE	10	0
21	7	TRUE	10	3
22	4	FALSE	10	1
23	6	TRUE	10	3
24	1	TRUE	10	5
25	4	FALSE	10	1
26	1	TRUE	10	9
27	1	TRUE	10	8
28	1	TRUE	10	5
29	--	FALSE	10	0
30	--	FALSE	10	0
31	2	TRUE	10	7
32	57	TRUE	10	5
33	1	TRUE	10	2
34	1	TRUE	10	4
35	1	TRUE	10	5
36	13	FALSE	10	1
37	2	TRUE	10	2
38	3	FALSE	10	2
39	3	TRUE	10	6
40	1	TRUE	10	9
41	1	FALSE	10	2
42	3	FALSE	10	2
43	1	TRUE	10	10
44	2	TRUE	10	4
45	4	TRUE	10	7
46	5	TRUE	10	6
47	4	TRUE	10	8
48	2	TRUE	10	6
49	1	TRUE	10	10
50	8	FALSE	10	2

Table 6-2: Raw data for the Speech + Context condition.

Input Index	Selection Rank	ASR Recognized	Num. Options	Num. Relevant
1	1	TRUE	8	6
2	1	TRUE	7	5
3	--	FALSE	0	0
4	--	FALSE	0	0
5	--	FALSE	0	0
6	--	FALSE	0	0
7	--	FALSE	0	0
8	1	TRUE	10	5
9	1	TRUE	10	6
10	1	TRUE	10	4
11	1	TRUE	7	5
12	--	FALSE	0	0
13	--	TRUE	10	2
14	1	TRUE	7	6
15	1	TRUE	10	8
16	1	TRUE	10	6
17	--	FALSE	0	0
18	1	TRUE	10	2
19	1	TRUE	10	6
20	1	TRUE	10	2
21	--	FALSE	0	0
22	1	TRUE	5	5
23	--	FALSE	0	0
24	--	FALSE	0	0
25	--	TRUE	10	1
26	--	FALSE	0	0
27	33	TRUE	10	1
28	--	FALSE	0	0
29	1	TRUE	10	3
30	--	FALSE	0	0
31	2	TRUE	6	4
32	1	TRUE	7	6
33	1	TRUE	1	1
34	5	TRUE	10	8
35	2	TRUE	10	7
36	3	TRUE	10	7
37	1	TRUE	10	7
38	2	TRUE	10	7
39	1	TRUE	3	2
40	1	TRUE	10	7
41	1	TRUE	10	6
42	1	TRUE	5	5
43	--	TRUE	10	0
44	1	TRUE	10	2
45	--	TRUE	10	0
46	--	TRUE	10	0
47	6	TRUE	10	3
48	--	TRUE	10	0
49	--	FALSE	0	0
50	4	TRUE	10	7

Table 6-3: Raw data for the Speech-Only condition.

Input Index	Selection Rank	ASR Recognized	Num. Options	Num. Relevant
1	1	N/A	10	3
2	4	N/A	10	2
3	1	N/A	10	1
4	--	N/A	10	0
5	--	N/A	10	0
6	34	N/A	10	0
7	25	N/A	10	0
8	23	N/A	10	0
9	4	N/A	10	1
10	14	N/A	10	2
11	1	N/A	10	2
12	--	N/A	10	0
13	--	N/A	10	1
14	25	N/A	10	0
15	65	N/A	10	0
16	--	N/A	8	0
17	24	N/A	10	1
18	8	N/A	0	0
19	--	N/A	10	0
20	12	N/A	10	0
21	8	N/A	10	3
22	1	N/A	10	2
23	1	N/A	8	2
24	1	N/A	10	4
25	--	N/A	10	0
26	3	N/A	10	3
27	33	N/A	10	0
28	--	N/A	10	0
29	5	N/A	10	1
30	2	N/A	10	2
31	1	N/A	10	2
32	3	N/A	10	2
33	20	N/A	10	0
34	8	N/A	10	1
35	111	N/A	10	0
36	74	N/A	10	0
37	--	N/A	10	0
38	50	N/A	10	0
39	--	N/A	10	0
40	--	N/A	10	0
41	1	N/A	10	2
42	--	N/A	10	0
43	24	N/A	10	0
44	4	N/A	10	3
45	--	N/A	10	0
46	--	N/A	10	0
47	--	N/A	10	0
48	--	N/A	10	0
49	--	N/A	10	0
50	3	N/A	7	1

Table 6-4: Raw data for the Context-Only condition.

The following two tables present statistics comparing the mean rank from each condition. Note that the sample size indicates the number of inputs (out of 50) in which a subject selected a dialogue option, rather than clicking CANCEL. The t-statistics in Table 6-6 show (in bold) that the mean rank of all conditions are statistically significant when compared to the mean rank from Context-Only, at the p = 0.05 level. The difference in mean rank between other conditions is not statistically significant.

	Text + Context	Speech + Context	Speech-Only	Context-Only
Mean Rank	2.58	5.07	2.63	18.00
Sample Size	45	44	30	33
Standard Deviation	3.12	9.34	5.88	25.03

Table 6-5: Mean selection rank and standard deviation for each condition.

	Speech + Context	Speech-Only	Context-Only
Text + Context	1.69	0.05	**4.10**
Speech + Context		1.27	**3.15**
Speech-Only			**3.28**

Table 6-6: t-statistics for independent groups t-test between mean ranks.

The following two tables present statistics comparing the percentage of relevant options generated for each condition. Note that the total number of options can fall below 500 (10 per 50 inputs), when the system fails to generate at least 10 options for some inputs. For example, in the Speech-Only condition, when speech recognition fails, there are no options provided for the input. The t-statistics in Table 6-8 show (in bold) that the difference between percentages for all conditions are statistically significant when compared to each other, at the p = 0.05 level.

	Text + Context	Speech + Context	Speech-Only	Context-Only
Total Num. Options	496	500	316	483
Num. Relevant Options	299	205	152	41
% Relevant Options	60.2	41.0	48.1	8.5

Table 6-7: Percentage of options flagged as relevant for each condition.

	Speech + Context	Speech-Only	Context-Only
Text + Context	**6.06**	**3.38**	**16.99**
Speech + Context		**1.99**	**11.76**
Speech-Only			**12.79**

Table 6-8: t-statistics for two sample t-test between percentages of relevant options.

The following two tables present statistics comparing the percentage of successful interactions generated for each condition. Note that all conditions include 50 interactions. A successful interaction is an interaction where the subject chooses a dialogue option, rather than clicking CANCEL. The t-statistics in bold in Table 6-10 indicate statistically significant difference between percentages, at the p = 0.05 level.

	Text + Context	Speech + Context	Speech-Only	Context-Only
Total Num. Interactions	50	50	50	50
Num. Successful Interactions	45	44	30	33
% Successful Interactions	90.0	88.0	60.0	66.0

Table 6-9: Percentage of successful interactions for each condition.

	Speech + Context	Speech-Only	Context-Only
Text + Context	0.32	**3.46**	**2.90**
Speech + Context		**3.19**	**2.64**
Speech-Only			0.62

Table 6-10: t-statistics for two sample t-test between percentages of successful interactions.

6.1.3   Discussion

The results demonstrate that the complete system (Text+Context and Speech+Context) is successful at using context to compensate for language understanding failures (due to failed speech recognition, or sparse coverage of possible inputs). A system that can effectively combine understood words with context can better support open-ended interaction, where a player to uses language as action. Quantitatively, the player can do what s/he wants to do more often. Figure 6-1 compares percentages of successful interactions visually, where subjects were able to find satisfactory dialogue options. The success of the system at supporting player choice is a direct result of crowdsourced imagination. The crowd has provided ample coverage of possible inputs (Section 6.1.4 shows that only six words in the Text+Context condition were not in the corpus, and most of these six are misspellings), and where words or phrases were not recognized, contextual understanding from annotated game logs led to suggesting options that subjects found satisfactory up (90% of the time for Text+Context, and 88% for Speech+Context). While a side-by-side comparison with a hand-crafted system would require reimplementing TRG without CAI, one can speculate that far fewer inputs would have been covered, and the system would have much more limited means of providing contextually relevant alternatives, if any.

Percentage of successful interactions
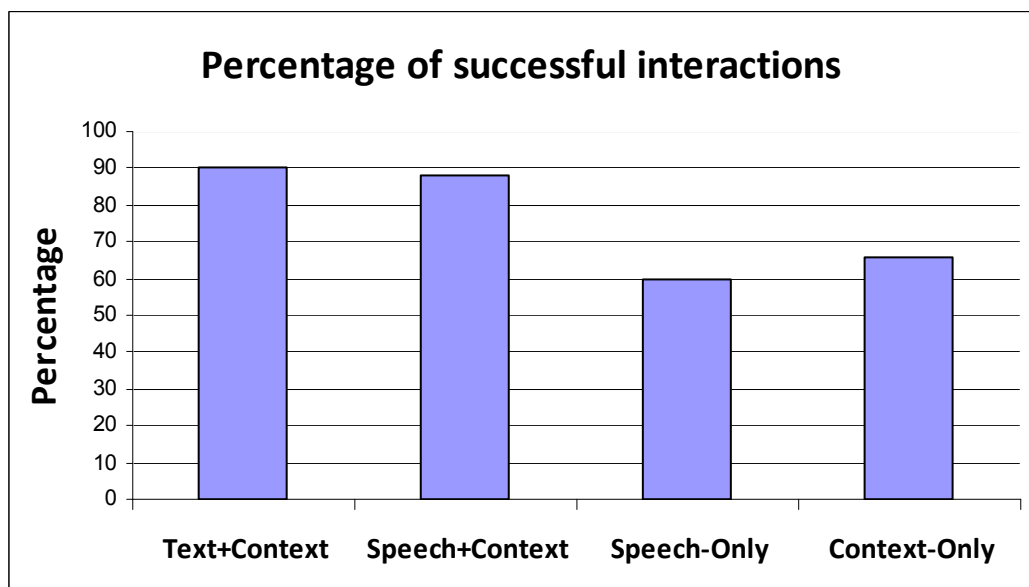
Figure 6-1: Comparison of percentages of successful interactions between conditions.

Table 6-11 presents a summarized view of the results, comparing various measures of system performance across conditions. For the speech conditions, this table reports that Speech-Only yields the highest percentage of relevant options (total for 50 inputs), and the lowest mean rank of the selected option (though the difference between means with Speech+Context is not statistically significant). In fact, Speech-Only achieves almost the same mean selection rank as Text+Context. However, looking only at these metrics is misleading, and does not tell the whole story.

When the speech recognizer fails completely, Speech-Only has no other means of selecting dialogue options, giving the subject only a failure message, and CANCEL. In the Speech+Context condition, subjects have 28% more successful interactions than with Speech-Only. Subjects were able to find a satisfactory dialogue option 88% of the time with Speech+Context versus only 60% of the time with Speech-Only. Looking at the converse, subjects aborted the interaction due to dissatisfaction with dialogue options 28% more often with Speech-Only versus Speech+Context, despite similar numbers of speech recognition failures. Also, there were two instances where Speech-Only allowed the subject to select an utterance that the *Plan Recognizer* could not understand in the current context (due to sparse data). Two times out of 50 is not a lot, but consider that this is an issue that never occurs in the other conditions where options are filtered by *Critics*. Speech+Context performs considerably better than Context-Only, validating that the words are important in this scenario, but context can compensate for failure to understand words. Text+Context performs best on all measures, demonstrating the potential for improvement as the quality of speech recognition improves.

	Text+Context	Speech+Context	Speech-Only	Context-Only
mean selection rank	**2.58**	5.07	**2.63**	18.00
% of opts flagged relevant	**60.2**	41.0	48.1	8.5
% of successful interactions	**90.0**	88.0	60.0	66.0
# of plan rec. failures	**0**	**0**	2	**0**
# of speech rec. failures	N/A	18	14	N/A

Table 6-11: Comparing four methods for populating dialogue options.

For any speech input, Figure 6-1 plots the likelihood that the subject's selected option will be rank N or less. Of the speech conditions, Speech-Only delivers the highest likelihood of providing the desired selection at rank 1, and plateaus after rank 5. If the spoken words are recognized correctly, and a similar utterance exists in the corpus, Speech-Only is most likely to provide a desirable option at the top of the list. For each method, the remaining likelihood in the space above the plateau represents the likelihood of aborting by clicking CANCEL. Speech-Only plateaus at 0.6 because this condition only has a 60% chance overall of providing a satisfactory dialogue option. Thus, no matter how many options the subject looks through beyond the first 5, there is always a 40% likelihood of clicking CANCEL. The conditions leveraging context have a higher likelihood of providing a desirable option, though possibly later in the list, rather than no satisfactory options at all, leading to fewer aborted interactions. Text+Context performs best overall, providing an upper-bound on selection rank, as it relates to speech recognition quality. Context-Only serves as a baseline, illustrating that recognizing words plays an important role in generating high-quality dialogue options.
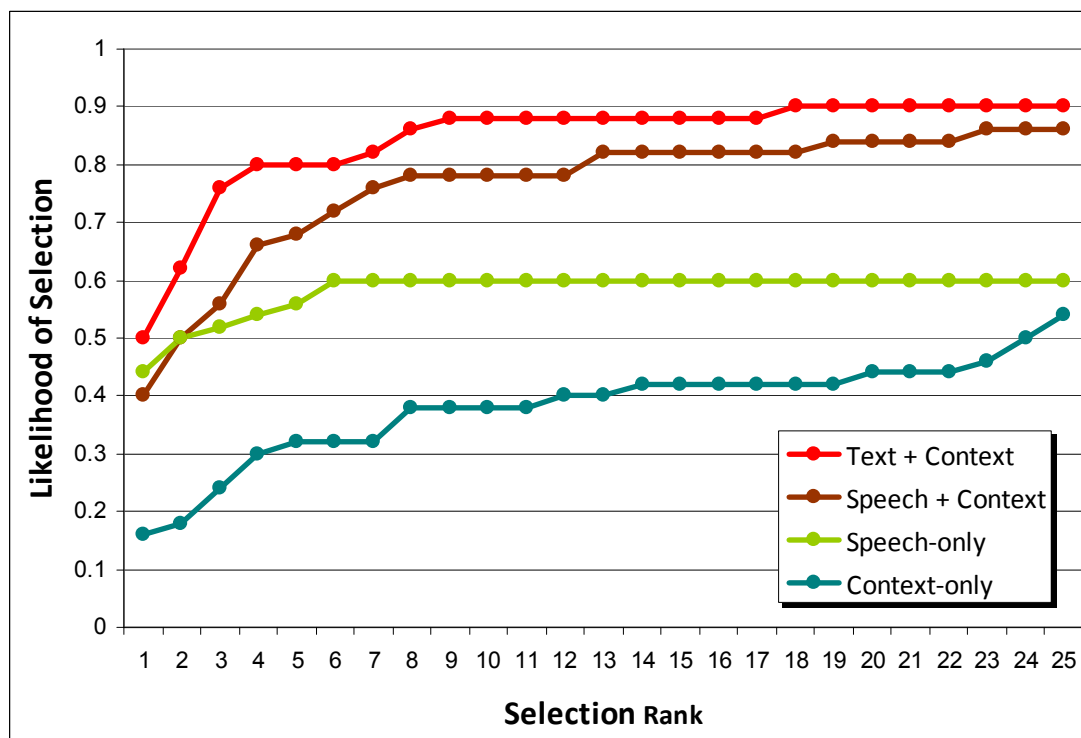
Figure 6-2: Likelihood of selecting dialogue option rank N or less.

6.1.4 Other Metrics

The results above show that, in the context-only condition, users aborted 34% of the time, out of dissatisfaction with the available dialogue options. This means that 66% of the time (two thirds), based on context alone, the system was able to provide an acceptable choice, though its rank may been further down in the list forcing the user to click through several pages of options. The ability of the system to suggest appropriate options based on context alone is only impressive if the number of possible alternative dialogue options is vast. Perplexity is a metric from the field of natural language processing for measuring the effective branching factor of a model based on entropy. Based on a trigram model learned from the 1,000 transcript training set (compiled into ACTIDs and U-SETIDs) driving interaction in this evaluation, the perplexity is computed to be 41.35. Essentially, it is as if each time the system selects the next action, it is selecting from about 41 possibilities, all equally likely. For comparison, a trigram model trained on 38 million words from the Wall Street Journal was found to have a perplexity of 109.

Figure 6-3, below, illustrates the effect of corpus size on coverage of recognized words and utterances. The graph was generated by extracting all utterances from the five Text+Context games, and counting how many of the words and utterances have an exact match (case- and punctuation-insensitive) in a corpus of size 1 ... 10,027. About 90% of the words are recognized within the first 500 games. The rate of recognition after that point is significantly slower, and reaches about 96% coverage by the time the entire corpus of 10,027 game logs has been processed. The six unrecognized words are mostly misspellings, with one colloquialism: blooowwwwwwssss, bullocks, dellicious, wadday, yodelehehoooooo, and zzzzzzzzz.
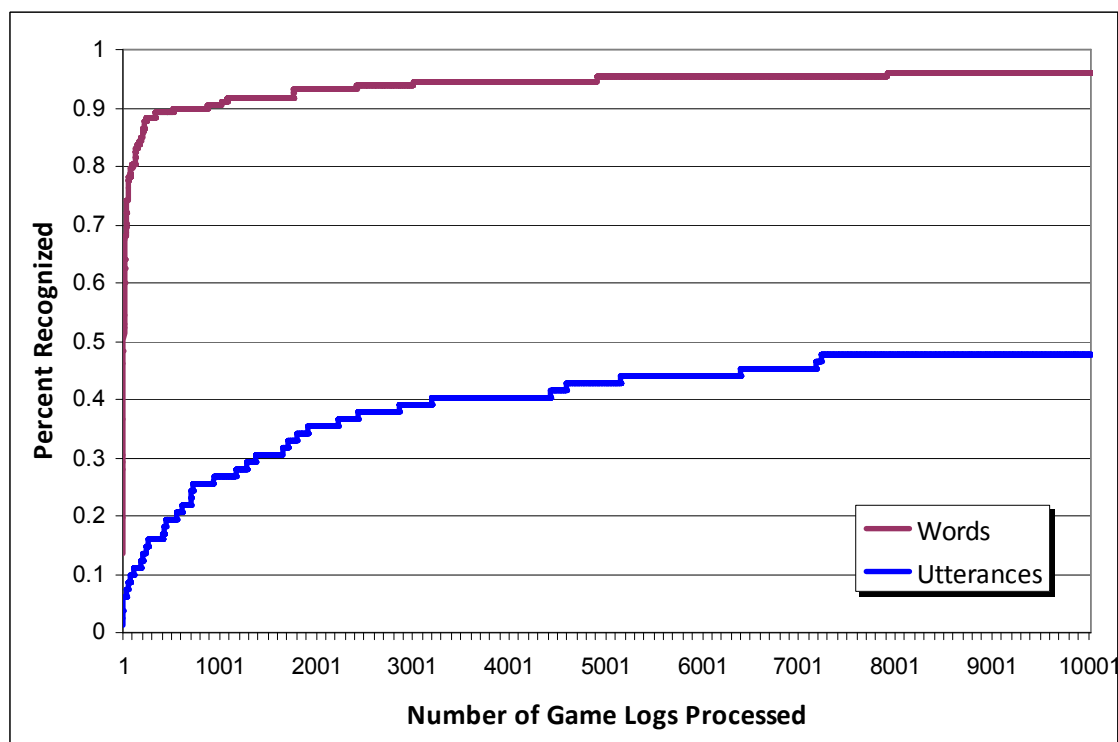
Figure 6-3: Percentage of words and utterances recognized, with increase in logs processed.

Utterances are recognized at a more gradual rate, and hit a plateau of about 48% coverage after processing 7,230 game logs. Table 6-12 contains the 43 utterances never observed in 10,027 game logs. Some of these utterances contain out of vocabulary words, but many are composed of recognized words in unfamiliar sequences. Some of the utterances are surprisingly ordinary (e.g. "I'd like a glass of the house red"), but just did not happen to come up in exactly the same form in the 10,027 recorded game logs. Note that the implemented system does have two mechanisms in place to recognize close, but not exact, matches (described in Chapter 5) -- clustering concepts, and retrieving U-SETs by keywords found within. Thus many of these utterances will be interpreted correctly. In many cases, the corpus does contain something nearly identical, for example, "sure, I'd like a glass of the house red wine." This exploration of utterance coverage is presented simply to highlight one of the challenges of understanding open-ended natural language input, and to quantify the effect of corpus size.

beer and the lobster	ill start with a salad please
bring me berry pie	its very tasty
bullocks	manhattan shaken not stirred
can i get some mutton pie now	may i have the nectarine tart please
can i have another glass of beer	maybe ill have a drink at the bar
can i have some pepper	really what does that taste like
check please this place blows	so bitch
decaf coffee please	tell the chef the steak sucks
dellicious	the chef is bad
do you have any beer	the chef needs work
do you have decaf	the salmon was excellent
get me a seat first	there was a hair in my steak
hey buddy your food sucks	this food is terrible
hi honey	this looks like white wine
how about red wine	this restruant blooowwwwwwssss
how about the spaghetti	wadday mean enough
i already had dessert	what is your favorite dessert
i would like a soup please	what kind ao special
id like a glass of the house red	what main course would you recommend
id like a glass of wine	yodelehehoooooo
ill have the check now please	zzzzzzzzz
ill have the pie a la mode	

Table 6-12: Utterances in five human-NPC games, never observed in 10,027 recorded games.

6.1.5   Comparison to N-Gram Models

Chapter 4 describes some of the drawbacks of automating NPCs with n-gram models (e.g. non-sequiturs and cycles). This section gives a concrete side-by-side comparison between a system driven by n-grams, and the system driven by human annotation (the system presented in Chapter 5 and evaluated in this

chapter), in terms of dynamically generating dialogue options. For the Text+Context and Speech+Context games evaluated in Section 6.1.2, Table 6-13 details how often the player's selected dialogue option was ranked in the top 5 or 10 percent by the annotation-driven system. These results are compared to percentages of how often the selected dialogue option would have been in the top 5 or 10, when dialogue options are generated with a trigram model trained on the 1,000 annotated game logs.

The trigram model is learned from logs that have been compiled into sequences of ACTIDs and U-SETIDs, seamlessly interleaving physical and linguistic actions. For each natural language input, the trigram model populates the list of dialogue options with the 10 highest probability U-SETIDs, based on the two most recent observations, and computes the rank of the U-SETID of the utterance actually selected. The results show that the trigram model dramatically underperforms, due to its reliance on only low-level patterns in recent observation history. In many cases (77% of Text+Context, and 70% of Speech+Context), the trigram model did not predict any utterance at all at the moment of player input, thus was unable to provide any dialogue options. This is a problem of sparse data, when only trained on 1,000 games, but the system driven by human-vetted annotations does not suffer from this problem when running from the same corpus. Training the trigram model on more data was not possible for this test, because compiling logs into ACTIDs and U-SETIDs relies on annotated logs, for human clustering of utterances. It is notable that there were a few cases (1 for Text+Context, and 3 for Speech+Context) where the trigram model proposed correct utterances where the annotation-driven system failed, indicated by the player aborting the interaction.

Text+Context	Annotation-Driven	Trigram-Driven
% in Top 5	71.3	5.3
% in Top 10	77.7	5.3

Speech+Context	Annotation-Driven	Trigram-Driven
% in Top 5	60.2	8.0
% in Top 10	68.2	8.0

Table 6-13: Comparison of dialogue option quality from annotation- and n-gram-driven systems.

6.2 Qualitative Evaluation

The previously described study demonstrates that the system is successful in supporting open-ended interaction driven by natural-language input, for a customer in a typical dining interaction, but has no means of measuring the impact this open-endedness has on the player's experience. The purpose of the qualitative study is to explore whether players notice that *The Restaurant Game* (TRG) is more open-ended than other games, whether they feel their natural language input has effect on the narrative, and to what degree this open-endedness enhances their experience. This section reports observations captured from focus groups in which *The Restaurant Game* is compared side-by-side with two other games: *Façade*, an experimental interactive drama from 2005; and *Skyrim* (Bethesda 2011), a AAA commercial role-playing game (RPG) from 2012.

This section describes the experimental setup, presents results, and concludes with some discussion of what this qualitative study revealed. Prior to the results, visualizations of gameplay in TRG and *Façade* are provided, to characterize interactivity in these games. Results are grouped into transcribed salient observations about different aspects of the experience, with some surrounding commentary. The discussion highlights successes of the system, priorities for future work, and some surprising observations.

6.2.1   Experimental Setup

Three focus groups were conducted, with six participants each. Participants were recruited via an email to the university EECS mailing list, and screened to ensure they had some prior game playing experience. None of the participants had played *Façade* or TRG previously. Each group had at least two people who played *Skyrim*. The groups of six were divided into two groups of three, where each member of the group played each game once for ten minutes, while the others observed. In total, 18 recordings were captured of each game. Note that *Façade* and TRG only take 10-15 minutes to complete, while *Skyrim* can take 30 hours, or much more. For the purposes of this study, focused on NPC interactions, players loaded a previously saved game placing them into a tavern in *Skyrim*, in which they could interact with NPCs for 10 minutes. While this short interaction does not give players the full gameplay experience with *Skyrim*, it is enough to remind participants of the current state-of-the-art for typical NPC interaction in commercial games.

TRG is positioned side-by-side with two complete, polished games -- the experimental game *Façade*, and the recent blockbuster AAA RPG *Skyrim*. Despite the word "game" in the title, TRG is not really a game at all, but rather a simulation of everyday restaurant behavior, without any goals for the player aside from having a virtual meal. TRG is really a demo of how data-driven technology could be incorporated into a full-fledged game in the future. The purpose of this study is compare different approaches to interacting socially with characters, in order to evaluate whether players notice that TRG allows more open-ended NPC interaction, and if so, whether this contributes to a more engaging experience. Everyone plays each game for 10 minutes, which means players are getting a severely limited experience with *Skyrim*, an epic-scale game that can take 50 hours or more to complete. Locking players in a tavern focuses observations on the NPC interactions, rather than combat or exploration, putting the experience on more equal footing with the short one-room experiences in *Façade* and TRG.

6.2.2   Visualizing Interactivity in *The Restaurant Game* and *Façade*

Façade was chosen as one of the games for comparison in the qualitative study due to its similarities to TRG in terms of face-to-face social interaction in an everyday setting, with a similar interface for unrestricted typed-text dialogue input. As discussed in Chapter 2, *Façade* has a different emphasis than TRG. *Façade* prioritizes delivering a dramatic experience, while TRG prioritizes supporting an open-ended player-directed experience, where language can be used effectively as action. The qualitative study is intended to explore whether players notice and appreciate the open-endedness of TRG. Because some readers may not have played *Façade*, and most readers will not have played TRG, this section uses data from the qualitative study to visualize how the gameplay experience differs in these two projects.

Figure 6-4 plots all of the action sequences observed in 18 play throughs of TRG (on left) and *Façade* (on right). Existing infrastructure was used to plot the graph of TRG. The methodology for graphing *Façade* was the following: First, each gameplay session was transcribed by a human. Next, functionally similar utterances were clustered, similar to the approach for TRG described in Chapter 5. Finally, all 18 transcripts were plotted on the same graph. As a post processing step for the graphs of both TRG and *Façade*, linear action sequences within the graphs in which there are no branches coming in or out were collapsed into a single node. This step focuses the visualization on decision points, to illustrate how the branching patterns of the two games differ. In both graphs, physical and dialogue actions are represented uniformly interleaved, and sequences are capped at 25 actions, to facilitate visualization.

At a minimum, one can recognize that the structure of these graphs looks visually different. The graph of TRG contains about 40% more edges (230 edges vs. 136 for *Façade*), and appears more chaotic than *Façade*. The graph of *Façade* could be described as a more neatly structured cascade of visible beats. Table 6-14 compares the lengths of games, and number of unique actions observed.

	The Restaurant Game	**Façade**
Min. Game Length	63	14
Max. Game Length	147	138
Mean Game Length	108	95
# unique actions observed	633	590

Table 6-14: Comparison of game lengths and variety of actions observed.
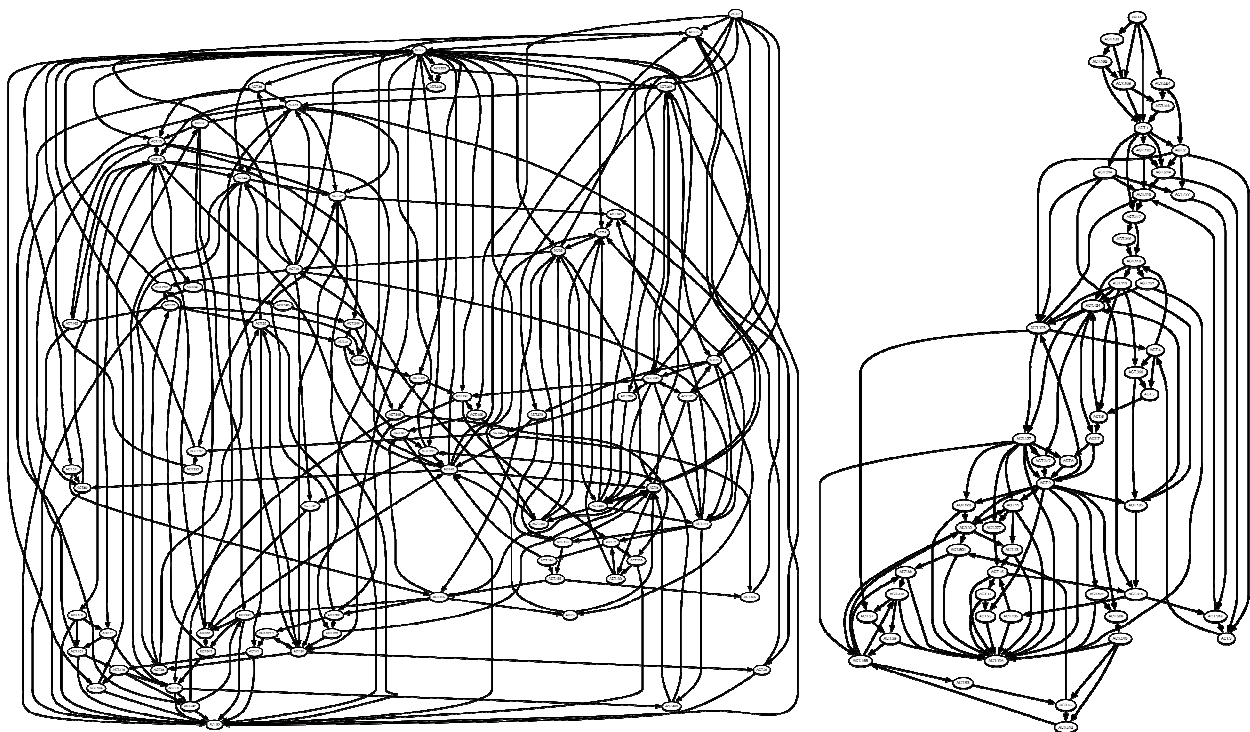
Figure 6-4: Branch points observed in 18 games of The Restaurant Game (left) and Façade (right).

The structure of the above graphs highlights the difference in variability per play through. What is not evident from these graphs is the percentage of actions taken by the player, versus taken by NPCs, and the rate of growth of the action lexicon. Figure 6-5 plots the total number of unique actions observed after playing N games (1 to 18). The graph on the left plots the count of unique action bigrams (two actions in sequence). The graph on the right plots a total count of unique actions taken by the player. The variety of unique action sequences begins to plateau after 18 games of *Façade*, while it continues to increase in TRG, and the variety of player actions grows at a much slower rate in *Façade*. Figure 6-6 illustrates the percentage of all actions taken by the player versus the NPCs, across all play throughs. In TRG, almost half of the actions are taken by the player, where they account for only 14% in *Façade*.
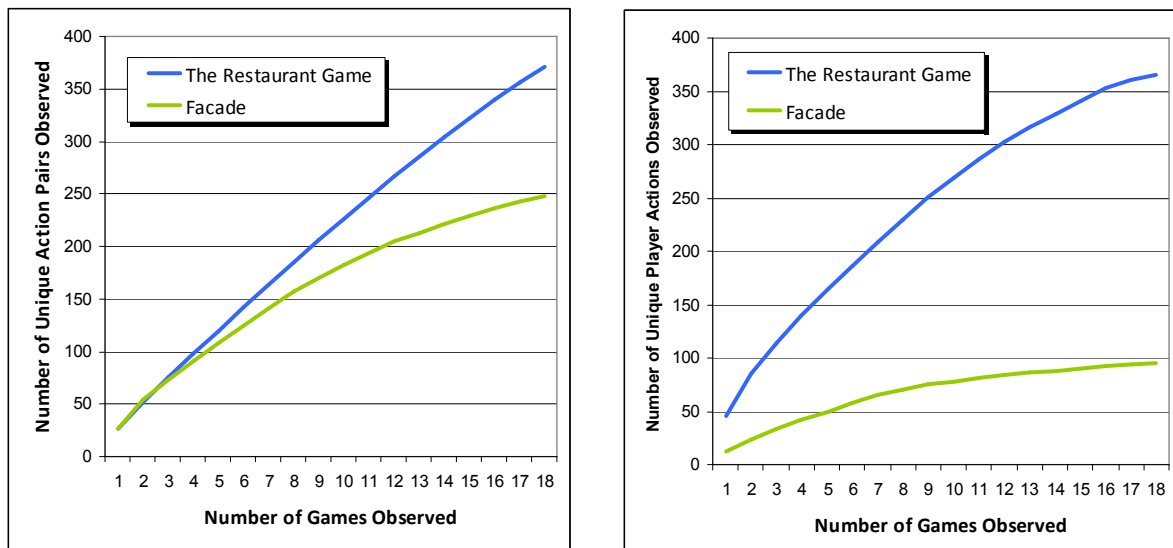

Figure 6-5: Number of unique observations after N games: action pairs (left), player actions (right).
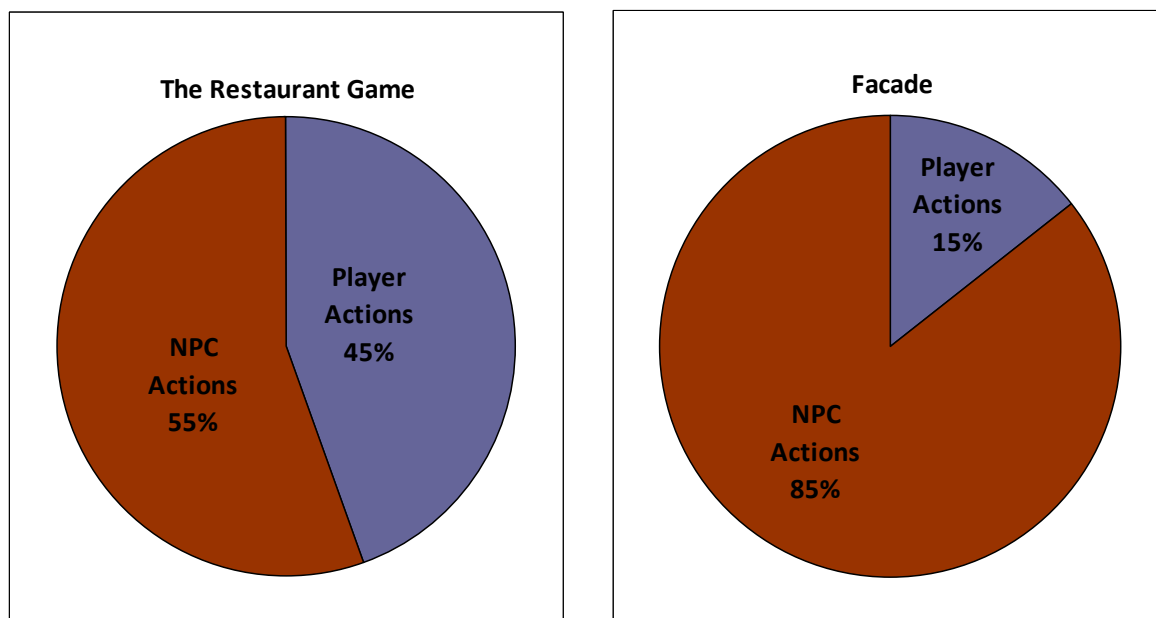

Figure 6-6: Percentage of player actions, compared to NPC actions.

6.2.3 Observations About Engagement and Replay Value

Whether a game is designed to entertain or educate, ultimately it can succeed at neither unless it engages the player. A game that continues to engage over repeated interactions maximizes its potential to entertain or educate. Observations from the play sessions reveal that there are numerous factors that impact engagement and replay value, including goals, challenge, variety of experiences and outcomes, depth of character and story, and how active or passive the interaction is. Some of these factors can be highly subjective. Below are various focus group observations identifying these factors (prefixed with the focus group ID: G1, G2, or G3).

Façade engaged players by confronting them with the goal of saving Grace and Tripp's relationship -- a goal that seems difficult, yet not impossible, to achieve.

> *(G1) [Façade] feels like it has the most replayability, because I want to see it work.*

> *(G2) [Façade] was a real interesting game, it like really pulled me in, I wanted to make it work.*

The fact that Grace and Tripp move the narrative forward, sometimes faster than the player can respond, adds to the challenge of mediating Grace and Tripp's argument. Some players found this frustrating, while others noted that the challenging timing added replayability.

> *(G3) It was very difficult to carry out the objective that I wanted to because of the response windows, like you know I would want to respond to something that was just said, but by the time I typed it out and hit enter, they think you know they've already said something else, and then they're like yelling at me because I said the wrong thing.*

> *(G3) [Façade] is very replayable because it's time-based, and because you don't know how the thing is going to parse it, you don't know if you're about to get kicked out, you don't know if they're just going to you know kiss and make up, I feel like it's harder to see how you're directing the events, so I think because of that it's harder to narrow down possibilities, therefore it will be more replayable.*

Other players felt like there were a finite number of end conditions to *Façade*, and they simply wanted them enumerated rather than experiencing it.

> *(G3) I don't want to keep playing it, but I think it would take a while to figure out all the endings. I'll probably go online just to find out what happens, read the wikipedia page, and see if it has like the answer, or like possible paths.*

In contrast, the open-endedness of TRG encouraged replay for exploration, while the prescribed nature of *Skyrim*'s NPC interactions eliminates replay value.

> *(G1) The thing with Skyrim is that since you're only given a small number of dialogue options you can't really influence the direction of the world.*

> *(G3) I played Skyrim second, and all the interactions had already been had, there was like no freedom, like I only watched one person play before me, and I did the exact same thing basically*

> *(G3) I think the most fun to replay would be the waitress just because so much of the game was dictated by things my typing would prompt, it seemed the most open-ended game, well sort of, had many paths to explore, which was kind of fun, and then the Façade seemed like no matter what you did you either got kicked out or yelled at, but that's still more entertaining than having the same dialogue over and over. if I was playing Skyrim again I would probably just get the sword and see how many people I can kill.*

For some players, open-endedness is not enough to create engagement without more challenging and explicit goals, or a supporting story. The basic goals involved in dining at a restaurant are taken for granted, so familiar that they are perceived as easy to achieve. Whereas *Façade* felt like decisions within the game might have consequences in a larger overarching narrative.

> *(G3) I think that out of all the three games, The Restaurant Game would be the game that I just wouldn't play again, because I mean when you read the initial screen the goal was to get dinner at the restaurant, but it's very easy to get dinner at the restaurant, you know if depending on what you say she either didn't serve you dinner or called the police or whatever it would be different but you know.*

> *(G3) It's like life. Like you walk in you don't really have an objective and like it's not particularly hard to go to a restaurant so it kind of reminded me of actually going to a restaurant.*

> *(G2) There's not much of a history and a world trace to like work off of, so you're just kind of telling yourself ok this is a restaurant, and what I do here doesn't matter anywhere else, whereas in Façade you have some kind of backstory right? Like you were friends with this person and they want to reconnect, and if this goes badly what will I do, how will their marriage end up?*

> *(G2) I invested more in [Façade] because I was worried about how it would turn out.*

The combination of the deeper back story with the time-based nature of *Façade* created palpable tension for players. Some players described it as movie-like, and were content to sit back and watch the narrative unfold for minutes at a time.

> *(G2) [Façade] just keeps going, there's like no way to stop it. No matter what you do, they just end up arguing more. There was a lot more tension in that game than the other ones*

> *(G1) Façade would move forward without you doing anything, and so I feel like that one had almost like a movie sense to it, where you could interact with it if you wanted, but you could also sit back for minutes at a time and just listen to what they were saying.*

It was interesting to observe that players sometime assume that all narrative variations are a consequence of the player's actions.

> *(G2) In Façade I think [my actions] did matter. For example I played last and both of them had, when they were playing Tripp was playing with the magic 8 ball, but when I played he didn't even touch it, he didn't talk about it at all.*

6.2.4 Observations About Variety and Coherence

Players commented that *Façade* and TRG varied content to play out differently each time, while *Skyrim* played the same each time.

> *(G2) With Façade you can always get new things.*

> *(G3) [The waitress's] reactions didn't seem deterministic either, because sometimes even the way she would greet you when you entered like, it's not like you did anything different but she'd say something different. ... There would probably be a lot of like hidden gems that even if you did the same thing every single time you'd probably get different reactions.*

> *(G2) [In Skyrim] you can talk to that same character again, and you'd have the same exact options and the same conversations.*

However, some players expressed that the variations in TRG sometimes felt incoherent.

> *(G1) like we talked to the waitress and asked her name several times and she gave a different answer each time.*

> *(G3) You know in the waitress thing I asked her before stealing the cash register do you want to go out with me, I got a cold shoulder, I stole the cash register, ran out the door and then said do you want to go out with me and she's like, "What time?"*

These observations are side-effects of generating NPC dialogue by replaying fragments of games recorded from thousands of different people. The waitress's inconsistent name could be corrected by adding names to the ontology as concepts, so that the system can track them, and a *Critic* could prevent her from using the wrong name once her name has been established. The attitude *Critic* described in Section 5.3.4 could enforce a consistent personality over time (e.g. aloof vs. outgoing, or flirtatious). This *Critic* was implemented after running the studies. Dynamically changing an NPCs attitude in response to player behavior requires implementing a system for tracking affinity between characters, which remains for future work. Players noted that *Façade* was actively tracking affinity of NPCs toward the player.

> *(G1) I felt like [Façade] didn't actually listen to what I was saying most of the time, it would just kind of decide who's side I took every time I said something.*

6.2.5 Observations About Main Characters and Agency

Façade was the first instance of a new variant of games called an Interactive Drama, designed around the philosophy that the minimum number of characters to produce drama is three, where the player is explicitly not the main character (Mateas 2003). This unique design provides a practical way to work around the difficult problems of natural language understanding, masking understanding failures with two self-absorbed main characters, Grace and Tripp, who start arguing whenever they do not understand the human input -- a practical solution, which feels plausible, even natural, given the scenario dramatized in *Façade*. Players recognized this relationship with the NPCs in *Façade*.

> *(G1) Grace and Tripp are the main characters in Façade. The player is just there for the story to progress around.*

(G2) I feel like both of them were, their situation, the structure of the relationship was the main character.

(G3) I think in Façade you're not really the main character, because the main story line doesn't really have anything to do with you really I feel like you're more of an observer.

In the observations above, the main characters are determined by their roles in the story. Players also noted a connection between the influence one has on the story and who is the main character.

(G3) [Façade] seemed very game-led which was weird because it was a free-response game, and then sometimes you know I'd be trying to respond to the question they just asked me, and they'd be like oh look at our window, and it's just completely unrelated and they don't even respond to the last thing I said, so like I said it seemed to not matter at times what you said, but the game would just automatically move forward. I think if you don't have a lot of ability to exercise free will you're not really a character.

Façade is well executed, and the previous section illustrates that ignoring the player at times does not negatively impact engagement, and even increases tension, however this is not a general solution applicable to all games. In contrast, the majority of games orbit around the decisions of the player, who controls the main character. Players commented that TRG and *Skyrim* were player-driven, while also noting that the waitress does take the initiative at times in TRG.

(G1) Façade led you, The Restaurant Game lets you lead it.

(G3) In Skyrim, and The Restaurant Game a lot of the times, I was driving the situation. In The Restaurant Game there was like a couple, like once or twice when the waitress would initiate conversation and would ask me what I wanted to do, but I still had to respond in the proper manner, whereas in Façade regardless of how I responded the situation would progress.

(G3) In The Restaurant Game sometimes she would initiate conversation if you just sat there, she'd be like oh do you want a table, do you want to sit down, do you want a menu, stuff like that, if there was no user input for a while. In Skyrim you know like AI doesn't just walk up to you and ask you questions, it's always me walking up and saying, "oh yes, hi play this song for me. I really like it."

Given the freedom the player was given in TRG, players were surprised by the range of interactions the waitress could respond to. Players commented that the waitress does try to support the direction the player takes the narrative.

(G2) I tried to click on the chair, and I ended up sitting on the table by accident, and she just came over and said "you're sitting on the table", I'm like "yeah sorry about that." I was a little bit surprised that she didn't just come back with a stock conversation like "What would you like to order?" She like noticed that I was sitting on the table, and asked what I was doing.

(G1) It felt like The Restaurant Game was trying to play along with the player. It just kind of roles with it.

Some players drew a distinction between being the main character and being player-driven, based on the existence or absence of plot.

(G1) It's hard to make the player the main character if you choose what to say, and you have no back story. If you need plot for a main character, then the waitress isn't, because there is no plot.

147

(G2) I feel like in The Restaurant Game there really isn't a main character because there wasn't really a story.

The first-person perspective in all three games also factored into some player's definition of the main character.

(G3) I feel like I was the main character in all of them. I mean in Façade it opens with, the conversation's about you, and the phone call that happened you're, and it opens with you standing alone, it's your first-person, like you're taking on the act of facilitator.

6.2.6 Observations About Natural Language Understanding

Skyrim presents the player with predetermined dialogue options for conversing with each NPC. In *Façade*, the player types open-ended text input. Predetermined dialogue options ensure the system will understand the player's choices, while open-ended natural language input gives the player freedom to express him/herself. TRG's interface is designed to provide the best of both worlds, giving the player freedom to type anything, and then dynamically generating dialogue options intended to be as semantically similar as possible, and at a minimum contextually relevant.

Opinions on the best approach to dialogue interaction varied greatly. Some players were impressed with the ways *Façade* masked understanding failures.

(G1) It responded really well to things it didn't understand. There's like "what?" and then they would go back to what they were saying. You just kind of have them react in a normal way, and just kind of brush it off, and go back to what they were doing so you don't get hung up on it.

(G3) I think if you said something sometimes that didn't fit into the script, like they didn't understand, they would try to work around it. Cause like the first time we went to the painting and he was trying to get us to say what it was. I said "love" and he said, "great that's right," and then the second time we went to the painting he said something like "wasteful" and he's like "no romantic." Later I said "castle," and he said "no!" and she said "honey he could be right." That to me just seemed like a clever way of saying I don't know what he said. I think stuff like that I think is a little better for the immersion, you know realistic.

While other players found *Façade*'s approach frustrating, and would have preferred more transparent feedback.

(G2) The situation in Façade where you type something and the game doesn't understand, I felt like often they would just brush you off and try doing something different, so if you say something complicated they're like, "eh whatever, I don't understand it so I'm just going to change the topic," so it felt like a device to try and make you talk in simple terms, so I think something like the dialogue options where they definitely understand could be useful.

(G3) There was a worry that like when you type something in, it wasn't always clear if it would register, like I was saying in Façade like sometimes they'll just kind of look at you, and they'll have like a weird reaction, like pseudo reaction, and I can't tell if they just didn't understand what I said or if like what I said was awkward or like something like that, so it was like kind of difficult to discern.

(G3) I would've much preferred like if Façade just told me it didn't understand, because then I would know what I did made no sense. I would have preferred being able to see a dialogue box. In general I prefer dialogue boxes because it gives me the I guess the feeling that I can't break the game.

Players commented that *Façade* seemed to be looking for keywords, and noted where this caused problems for them.

(G1) The impression I got when I was playing it was that there were a few big like chunks of content that you could unlock by saying certain things, like I mentioned therapy and divorce to them, and it suddenly picked up on that and started going on that, so I think there a few specific like keywords that they'll pick up on that can influence the direction of conversation but that otherwise the general things that you say aren't so crucial.

(G1) Certain colloquialisms don't work very well. For instance, in my run they were talking about bartending, and so you know snark, snark, snark about bartending, and he's embarrassed about it, and so my reaction was "bartending's not a bad job," except that there's two negatives in there, and so it just automatically took it as a negative rather than negating the negative.

(G1) Yeah I noticed a couple times when we were playing too that it would like take the opposite interpretation of what we meant.

TRG's approach of dynamically generating dialogue options based on human input provides transparency, which players appreciated for supporting more nuanced input, and giving more control over how utterances would be interpreted by NPCs.

(G1) What I noticed about The Restaurant is that it was trying to do more than Façade in the sort of AI actual interpretation of colloquialisms.

(G1) I thought that the idea where it would like take what you type in and then say which of these did you mean was a really a good idea because it's like that way the program can pick a statement that it understand and then you can confirm that is in fact what you meant. And I think that would also let you avoid the problem you had with Façade where for example the double negative thing. So I think that's a really good system for like having the computer check to make sure that it knows what you mean while at the same time giving you the freedom like do whatever you want basically.

However, other players noted that TRG's interface, which pauses the game while the player selects a dialogue option, would detract from the tension of a real-time interaction. Players also commented that it felt strange when the options presented did not match the input (a side-effect of retrieval by context).

(G2) The problem with that is if you ended up having like type and it would pause their conversation to figure out what you were saying it would take away from the whole building tension in the game, because it's like stopping the conversation.

(G2) It was kind of awkward in The Restaurant Game to have to type something and then choose from a menu because sometimes what comes up isn't the question that you asked.

While others felt that seeing unintended dialogue options actually improved the experience, by suggesting things to say that might not have occurred to the player.

(G2) Mostly there was an option that was close to it, at least there was usually one, and there were some weird ones which was just fun. It's always fun to just type some random thing, see what you get.

149

In general, players expressed that they found open-ended natural language input to be more engaging. However, they also commented that some games are better with a more limited set of static options. In particular, players said that the actual full-scale *Skyrim* experience (as opposed to the limited tavern experience) would be overwhelming with open-ended input, and that deep NPC interaction is not really the focus of the intended experience.

> *(G1) Comparing the hard-coded options to the more open-ended you can say what you want, it definitely made me feel like I had more options that I wanted to explore when I could type whatever I wanted, like I wasn't sure how it would react to different things, so I would be inclined to spend a lot more time in a short segment of the game trying a whole bunch of different options and seeing what the outcomes would be, in The Restaurant or in Façade, where in Skyrim you're presented with the options and they're the expected you just go through the list.*

> *(G2) In Skyrim, you can't voice your own opinions, so you're not as invested.*

> *(G1) I like the big world of Skyrim. Skyrim is about exploration. Depth of character is not what the audience wants. It might be excessive, ...but does make games more interesting.*

Finally, some players simply prefer the reassurance of a preset list of options -- that no content has been missed.

> *(G1) The fact that characters just wait for you in Skyrim breaks immersion, but it's reassuring that you're not going to miss anything. Façade is like a movie. It doesn't get boring though, it's fluid, felt the most natural, but you wouldn't have time for so much conversation in a bigger environment.*

> *(G3) Actually I like the choice interface. When I play a video game I'm like a completionist. I like to, like, do everything and make sure you know, maybe I save it and I try one route and see if that likes comes out with the goal, and then I like start over or something, but, yeah it's almost overwhelming having too many paths. Like if you can imagine every conversation being like Façade with like every character you meet in the game, that would just be ridiculous, you know.*

6.2.7   Discussion

Based on the observations from the qualitative study in the previous sections, this discussion section highlights positive results and areas to focus on in future work, and concludes with comments about some of the more surprising observations.

The focus groups revealed anecdotal evidence that players did find TRG to be more open-ended than the other games, with respect to NPC interaction, frequently referring to it as a *sandbox*, unprompted, and there was general agreement that the player was able to drive the experience with natural language input more so than in other games. Several players noted that the TRG interface better supports more complex natural language input, including colloquialisms and nuance.

Despite incredible production values, *Skyrim* was the least engaging experience for the players -- with the significant concession that it was not being played as intended, with players confined to one tavern. Players found that NPC interactions in the tavern played out exactly the same way each time, and offered minimal player freedom or influence over the narrative. Each gameplay session invariably ended in combat, as players ran out of anything else to do. These observations about *Skyrim* validate the value of more open-ended interaction with NPCs, to produce more engagement when replayed.

This thesis began with a discussion of the two bottlenecks that introduce tension between player freedom and system comprehension: the authorial effort, and limited human imagination, which lead to limited coverage of possible inputs (actions and utterances). The contrast between the focus group responses to TRG and *Skyrim* illustrates how CAI addresses these bottlenecks to mitigate the tension between freedom and comprehension. In the case of TRG, one researcher (plus some outsourced labor) was able to create an experience so open-ended that players spontaneously labeled it a sandbox. When players refer to TRG as a sandbox, they are recognizing both the unusual freedom given to players, and the fact that the system comprehends and responds to much of what players choose to do. Players were pleased to find an NPC in TRG who responds to sitting on tables, stealing cash registers, and getting hitting on for a date, in addition to responding to typical restaurant inputs expressed with a wide variety of natural language utterances (as validated in Section 6.1). The freedom and comprehension in TRG are made possible by drawing from an enormous pool of content, mined for interconnections. *Skyrim*, on the other hand, has no issues with system comprehension, but greatly restricts player freedom (outside of combat). While this might be a deliberate design choice, one can speculate on the practical considerations which motivated this design decision. TRG and *Skyrim* inhabit similar spaces (a restaurant and a tavern). Using current tools of the trade in commercial game development, authoring responses to all possible physical and linguistic interactions in the tavern by hand would be prohibitively costly, in terms of development effort, and would likely not adequately cover all possibilities. The developers of *Façade* did choose to embrace the challenge of open-ended interaction, despite authoring content by hand, and as a result *Façade* provides much more player freedom than *Skyrim*. Design choices in *Façade,* which favor dramatic tension over agency, distract the player's attention from system comprehension failures. Section 6.2.2 illustrates the impact of *Façade*'s design philosophy on how player-directed the experience is, leading to less of a sandbox experience.

Focus group feedback indicating that player freedom in TRG provided a unique experience, adding entertainment value and encouraging replay, is a positive result given the unfinished nature of TRG compared to the other games. However, player freedom is not the only factor to ensure an engaging experience. With a noticeable lack of explicit goals or strong narrative, players of TRG were not as engaged with the game as they were with *Façade*. *Façade* overwhelmingly dominated the discussion in all three focus groups. While players did express awareness that *Façade* often ignored their dialogue input, they were impressed that it unfolded differently each time, and were driven to try to intervene in the awkward, intense, uncomfortable situation in front of them. Players were also impressed with the depth of character and backstory in *Façade*, and the expressiveness of their faces.

There were many comments about goals in TRG, both the lack of prescribed goals, and the ability to set your own goals. As a result of being asked to play each game three times, groups of players were observed, unprompted, setting new goals for themselves on subsequent play sessions of TRG (e.g. ordering as many items as possible, stealing things, getting a date with the waitress), and generally found that TRG supported them in these pursuits. As players experimented to find the boundaries of the simulation, they were often pleasantly surprised that the waitress could respond to interactions with many objects in the environment, as well as questions about herself and requests for dates. However, with limited numbers of examples in the 1,000 games, sparse data sometimes led to insufficient numbers of dialogue options for social interactions outside the scope of basic restaurant interaction. While some players enjoyed the freedom to explore this sandbox, others commented that it felt too much like the real world, without being given explicit gameplay goals, thus decreasing engagement. In retrospect, having humans play the waitress with NPC customers may have better supported comparisons with other games, and led to a more engaging experience with natural goals of trying to

earn the most money, either by serving faster, upselling more items, or through social interaction encouraging higher tips.

The majority of negative comments about TRG were related to issues with coherence of personality, which players interpreted as memory failures. As discussed in Chapter 2, one of the challenges of automating an NPC from thousands of recorded performances is portraying a consistent personality, and avoiding schizophrenia. The waitress's behavior was *structurally* coherent with regards to completing the expected events involved in ordinary restaurant interaction (e.g. seating customers, taking orders, bringing bills), and the waitress did remember to bring the correct items ordered, and even add them to the bill. However, competency for basic restaurant interactions is expected in a restaurant simulator, and is so familiar that it is taken for granted. Incoherence in other respects distracted from the coherence of the core restaurant behaviors. Players commented that the waitress's behavior sometimes appeared "random," as her attitude fluctuated, and her behavior did not seem to take into account how she was treated in the past. Players tended to ascribe mental models to NPCs, and associated coherence with an NPC's memory. The perceived quality of memory appeared to have a big impact on how engaged players were with a game. Players were impressed that NPCs in *Façade* sometimes referred to things that had been said in the past, and were disturbed when past behavior seemed to have no effect on the waitress's later behavior in TRG. Based on these observations, the highest priority for future work on CAI should focus on increasing coherence by implementing a system to modulate affinity toward other characters based on observations. The attitude *Critic* is part of the solution, but was implemented after this study concluded, and does not yet address changing attitude dynamically over time.

There were a few surprises related to observations of player responses to *Façade*, which could be informative for future game designs. Related to the above discussion of goals, it was interesting to see the prominence of accomplishing goals in players' response to a social simulation. *Façade* is a game about social interaction, yet players comments often characterized it as a puzzle-solving or adventure game, which they were driven to figure out. Many players commented on the importance of response time in *Façade*, and the frustration of trying to type a response within the allowed window of time, before the NPCs move on to another topic. The drive to accomplish the goal of saving Grace and Tripp's relationship factored into their response to this decreased player agency (due to timing-dependent inputs) in an unexpected way. Multiple players explained that this actually increased the drive to replay the game, to type input within the allowed time in the next play session. In addition, it was interesting to observe that some players assumed that any variation in content was caused by player actions. On subsequent replays, some players believed that differences in Grace and Tripp's behavior was due to player actions, when in fact many of these differences were due to random selection of variations of dramatic beats.

One somewhat surprising insight came out of discussions with players who were familiar with *Skyrim* -- the game as a whole outside of this study. As commercial games increasingly strive for photorealism in graphics, physics, and animation, one might assume that more lifelike social interaction and dialogue would be desirable as well, to complete the realization of immersive simulation. In fact, some of these players commented that they would not want such spontaneous, open-ended NPC interaction in *Skyrim*, as it would interfere with the rest of gameplay, which focused on collecting items, increasing character stats, and killing enemies. Rather than quickly restocking in town, and possibly obtaining new quests through brief conversations in the tavern, players would be slowed down with time consuming dialogue. The design of today's games is a response to years of working around the limitations of artificial intelligence, and as a result gameplay is typically less focused on NPC interactions outside of combat.

Games keep players engaged for many hours by stretching the story across enormous worlds, not expecting players to spend too much time in a small area (like a tavern). Yet there is a tension between the enjoyment players had interacting with the NPCs in *Façade* and TRG, and the disconnect in applying new NPC technologies to the designs of currently successful games like *Skyrim*. This observation suggests that there is an opportunity to rethink game designs, and how stories are told in games. Rather than trying to enhance existing games with CAI, it may be more productive to explore previously impossible designs, which depend on rich, open-ended social interaction.

7 Conclusion

This thesis has described an end-to-end system for recording thousands of people playing roles online, discovering patterns in this data through a human-machine collaboration, and automating data-driven NPCs who can interact in the 3D environment and converse with humans via typed or speech input. The unconventional development process associated with this system introduces a new division of labor in interactive media production, where the majority of work is performed by non-programmers, democratizing content creation by recording performances from anyone online, and providing tools for non-experts to structure and filter the data. Combining an accessible, collaborative authoring process with a data-driven runtime planning system, that functions like a search engine, leads to the realization of NPCs who can say and do thousands of things in response to player actions and utterances.

By crowdsourcing content creation, and mining this data for inter-connections, CAI is able to make significant strides toward overcoming the bottlenecks related to authorial effort and the limits of human imagination, and deliver an experience that simultaneously supports improved increased player freedom and increased system comprehension. A quantitative evaluation has shown that the implemented Collective AI (CAI) system handles a wide range of natural language input, and can often leverage content to compensate for understanding failures. Focus groups demonstrated that players do find *The Restaurant Game* (TRG) to be unusually open-ended and player-directed, as compared to other existing games. However, these focus groups also revealed some weaknesses with the current implementation, and identified additional factors that contribute to engagement, which should be considered when extending CAI in the future.

Ultimately, the work described in this thesis brings us a step closer to realizing the full potential of an interactive storytelling medium -- where the player's actions shape the story being told; actions both physical and linguistic, open-ended and seamlessly interleaved; interacting with data-driven digital actors, capable of improvising and adapting to the wills of the player, while paying heed to an intended narrative. This chapter concludes the thesis summarizing contributions, responses to criticism, future work, and final thoughts.

7.1   Contributions

While much work remains for the future, the work completed thus far has made several contributions.

Collective Artificial Intelligence:
The primary contribution of this thesis is an end-to-end process called Collective Artificial Intelligence (CAI) for compositing recorded performances and automating NPCs who can play roles in interactive stories. The narrative structure associated with this process organizes each recorded performance into an action sequence, overlayed with an event hierarchy, supplemented with additional meta-data capturing dependencies and attitudes. The combination of this CAI process and narrative structure produce a new data-driven medium for simulated role-playing. A case-based planning system exploits this medium to bring NPCs to life, who can converse and cooperate with human players in open-ended scenarios. NPCs can interact with humans via typed text or speech input. This data-driven approach is a significant divergence from current hand-crafted approaches to authoring interactive experiences,

overcoming previous limitations of human imagination and the authoring bottleneck. Evaluations of an implementation of CAI applied to TRG have demonstrated that this approach supports open-ended, player-directed interaction using language as action.

Quantitative & Qualitative Evidence of the Value of Data-Driven Interaction:
Judging the success of a system that simulates human behavior and language can be difficult. The default metric is often a variant of the Turing test, where humans guess if an interaction partner is a human or machine. The goal of this work is not to deceive players into thinking the NPCs are human, but rather to support player autonomy and agency with improvisational NPCs, capable of playing along with the player through a fluid mixture of words and physical actions. To that end, the quantitative evaluation measures how well the CAI system can leverage corpus-driven vocabulary coverage and inferred context (from the *Plan Recognizer*), to provide relevant dialogue options, based on unconstrained spoken input. This study finds that the implemented system is able to provide a satisfactory dialogue option 28% more often than a system that relies solely on speech input, and cannot exploit context. These strong results are made possible by hundreds of hours of human labor tagging game logs, through a successful online data annotation effort. Focus group participants reported that they found TRG to be more open-ended and player-directed than other games.

A Practical Transparent Interface for Natural Language Interaction:
A data-driven interactive medium enables rethinking the interface for natural language interaction. By scaling up to provide adequate coverage of possible inputs, and exploiting context to compensate for understanding failures, a data-driven interface can robustly support using language as action in a player-directed experience. A quantitative evaluation has demonstrated that the interface implemented for TRG was able to provide a dialogue option adequately semantically similar to human input 90% of the time (and 88% with speech input). Qualitative focus group discussions confirmed that the transparent interface for natural language interaction supported more nuanced input than an opaque interface. In their own words:

> *It felt like The Restaurant Game was trying to play along with the player. It just kind of roles with it.*

> *I think the most fun to replay would be the waitress just because so much of the game was dictated by things my typing would prompt, it seemed the most open-ended game.*

> *In Skyrim, you can't voice your own opinions, so you're not as invested.*

> *What I noticed about The Restaurant is that it was trying to do more than Façade in the sort of AI actual interpretation of colloquialisms.*

Data Sets from *The Restaurant Game* and *Improviso*:
There is much potential to learn from the TRG and *Improviso* data sets beyond this thesis. TRG offers thousands of examples of everyday interaction in a restaurant using actions and words, while *Improviso* captures dramatic performances in a science fiction scenario. It is likely that other researchers will glean new insights from these data sets related to common sense, linguistics, action planning, data mining, and machine learning.

7.2   Responding to the Critics

While there is increasing interest in applying data-driven approaches to development of interactive media, there is also a healthy dose of skepticism. Over the years of working on this research, I have sought out feedback from a variety of experts from academia and the game industry. This section provides responses to a number of valid questions and concerns that have come from these interactions.

How can CAI be applied to larger scale games like *Skyrim* and *Mass Effect*?
Commercial Role-Playing Games (RPGs) are used in this work as examples of the current state-of-the-art in dialogue interaction with NPCs, which often leads to questions about whether CAI could be applied to games of enormous scale, consisting of many locations and characters, and hundreds of different dialogues. While it may be possible to apply CAI at this scale, to ask this question is somewhat missing the point. *Skyrim* and *Mass Effect* already exist, and a large audience of gamers enjoy them in their current form. This research is intended to enable new experiences, more focused on open-ended, replayable, player-directed social interaction with NPCs.

The game industry is going through an incredible period of disruption and change right now, due to new platforms and business models made possible by mobile devices, the cloud, and improved connectivity. The audience for games is growing exponentially, but not all players have the same tastes. In particular, there are trends toward shorter, highly replayable experiences for $0.99, rather than 30+ hour AAA experiences for $60. From a business standpoint, games earn more money the longer they retain players' attention. While the old model required enormous game worlds to keep players entertained for weeks, games today retain players by offering replayable experiences (e.g. play hundreds of times to earn all of the achievements), or by supporting user-generated content, continually evolved by an engaged community. CAI has the potential to enhance replayability with collaborative role-playing characters, and democratizes content creation through a process that could be managed by communities of player-designers.

What if it is not possible to collect the data?
Due to the data-driven nature, CAI requires recording human performances to begin the authoring process. This introduces a different production process than that currently followed in the game industry. Where there is proven value, the industry has been known to adapt new processes, some of which are data-driven. For example, motion captured animation, and even capturing 3D models from peoples' faces, have become accepted parts of production. In general, beyond game development, in all fields, it is becoming increasingly easy to capture, store, and process massive amounts of data, and there is no reason to expect this trend to change anytime soon. Data-driven approaches are becoming the norm in all aspects of life, and it is just a matter of time before this is the case for authoring interactive content for games.

How does crowdsourcing content help an industry that is focused on crafted experiences?
There is a trend in AAA game development toward crafted experiences that are approaching Hollywood level production values. For instance, the lighting is perfect, camera angles are carefully planned, and characters are voiced by A-list actors. Outsourcing content creation to the crowd is inconsistent with this trend. CAI is a disruptive technology that might be considered two steps forward, one step back. While

developers give up some control, the result provides new levels of interactivity, intentionally moving away from the linear medium of Hollywood films.

The history of graphics is informative here. While resolution was steadily increasing to the point that hand-illustrated 2D games looked as good as Disney films, 3D graphics engines emerged. Initial 3D renderers offered blocky, pixelated visuals that could not compare to hand-drawn 2D, yet interactivity literally gained a new dimension. Ultimately, 3D graphics hardware and software steadily improved, and today's games look nearly as good as Pixar films while rendered in real-time from the player's camera. One can imagine a similar trend for data-driven interaction with NPCs. Initially designers may not have as precise control over every detail of the interaction, and may have to make sacrifices in exchange for increased interactivity, but as approaches like CAI improve, the experience of interacting with data-driven NPCs will far exceed anything that designers would be able to craft by hand.

Furthermore, there is an argument for data-driven behavior and dialogue that is consistent with the industry's reliance on motion capture. While development studios have skilled animators, and sophisticated animation tools, hand animation of human motion fails to capture the nuance of actual human motion. When humans observe an animated character, they can tell whether it was hand-animated or motion-captured for reasons that are difficult to articulate. Something deep in our subconscious mind is wired to recognize human motion. The same is true of patterns of interaction and dialogue. Hand-crafted behavior and dialogue somehow fails to capture the nuance that separates actual human interaction from hand-crafted content.

Voice recording thousands of lines of dialogue is impractical.
The current TRG prototype includes 18,000 lines of dialogue, used by the waitress to both understand and generate utterances. Professional game developers frequently respond to this fact with concern over production time and costs of recording all of these lines. This concern is valid. However, in the short term this issue can be addressed by only recording a portion of the lines, since many are variations that have the same semantic function. The real benefit of supporting 18,000 lines of dialogue is coverage for understanding human input, thus generation of all of these lines is not as crucial. Long term, it is possible that speech synthesis technology will improve to the point where generating thousands of lines is possible without actually recording each line. This thesis research is thinking about content at a different scale than what developers assume are the limits of possibility today. Regardless of the challenges of supporting thousands of lines of dialogue, if game players demand content at this scale, the industry will find a way to meet the demand.

How can coherence be ensured when content is crowdsourced from thousands of different people?
Coherence of personality has been addressed previously in this document. The attitude *Critic* enforces consistency of personality, based on a specified attitude, and future work remains to implement a system to modulate affinity between characters, and dynamically adjust attitude accordingly. Beyond these technical solutions, coherence can also be addressed by considering who is the in crowd that generates the content. As an academic research project, TRG collected data from the general public. A commercial application of this approach might collect data from a cast of employees in house, or might hire a collection of writers or actors. A crowd of trusted collaborators can be given more in depth direction and backstory for how to play roles. Higher quality participation might require fewer recorded performances, as less data will consist of misbehavior. However, even with a trusted crowd, it might be useful to collect additional data from the public to better cover the space of how people will try to interact with the system.

7.3 Next Steps

The focus groups were enlightening in terms of identifying factors that to lead to engaging gameplay experiences, and recognizing where TRG excels, and where to focus in the future. Next steps will focus on addressing shortcomings in TRG, as well as improving the development and debugging process, generalizing to new scenarios, scaling up to more content, and integrating into larger game worlds. This section focuses on near term next steps, while the following section explores broader scoped future work.

7.3.1 Capturing and Generating Strong Narratives

The original human players of TRG, during the data collection phase, were simply told to either earn money as a waitress, or have dinner as a customer. As a result, the "game" generated from this data is simply a simulation of dining in a restaurant, with no overarching narrative and few gameplay goals. The focus groups' response to *Façade* demonstrates the powerful effect of a strong narrative on player engagement, even at the cost of reduced agency -- players were drawn into the game despite reporting that their input was often ignored. Some players enjoyed the open-endedness of TRG, creating their own goals (e.g. getting the waitress to go on a date), but others lost interest, complaining about the absence of a motivating story. This shortcoming is more related to the data collected for TRG than the technology driving the experience, and future work will focus on capturing and generating scenarios with stronger guiding narratives and player objectives. In fact, data sets already exist for this purpose -- Mars Escape is a goal-directed puzzle, and *Improviso* has a strong backstory, and three act narrative structure. However, additional types of annotation and *Critics* may be required to produce a satisfying experience. For example, *Façade*'s drama manager ensures that the dramatic beats unfold such that the tension of the experience adheres to a dramatic arc. Meta-data and *Critics* have not yet been implemented to modulate tension in CAI.

7.3.2 Improving Personality Coherence

Players complained that the behavior of the waitress in TRG was at times incoherent. Her name could change within the same gameplay session, and her personality could oscillate from polite to rude seemingly without reason. While the CAI system does ensure structural coherence of basic restaurant interaction (e.g. the waitress serves what the customer ordered, knows to bring appetizers first and dessert after the main course, and follows established social conventions of getting seated before ordering, and bringing the bill after the customer is finished), these everyday behaviors are so familiar that players take them for granted, as they fixate on inconsistencies mentioned previously. Currently, the ontology includes concepts for items involved in restaurant interaction, such as food, the bill, and the menu, but does not cover concepts related to the waitress as an individual (e.g. her name, age, where she is from, etc.). Extending the ontology would provide the means for correcting inconsistencies.

The attitude *Critic* described in Section 5.3.4 was implemented after the focus groups, and addresses the oscillating personality, though this *Critic* enforces a static personality. Ideally the attitude would change in response to observed player behavior (e.g. respond rudely after observing offensive player

utterances). Focus group participants noted that the NPCs in *Façade* seemed to associate each player input with an affinity toward one NPC or the other, and adjusted their behavior accordingly. There were comments that this is a more complex version of the way commercial RPGs, like *Skyrim*, often associate dialogue options with the effect each option will have on an NPC's affinity toward the player. A mechanism for modeling affinity has not yet been implemented for CAI. This feature would require tagging of actions and utterances that modulate affinity, and a data structure that associates (previously tagged) attitudes with ranges of affinity. As mentioned in Chapter 2, the implementation of *social physics* in *Prom Week* might provide insights for the way forward. It would be interesting to learn such social physics from data by discovering causal chains in recorded performances, either automatically or through human annotations.

7.3.3   Scaling Up

The current implementation of CAI has successfully scaled to running from a corpus of 1,000 game logs, composed of over 18,000 utterances and 1,414 unique actions. While one of the strengths of the system is that it can utilize examples of behavior observed only once, adding more data will further improve coverage of human input, variety of NPC behavior and dialogue, and ability to respond to less common interactions where data is sparse (e.g. complaining about the food or bill). Future work will work toward annotating and importing all 10,027 game logs, and eventually scaling beyond the current corpus. This will introduce new challenges in searching for responses in real-time. The current implementation relies on linear searches through the corpus (leveraging the ELI for efficiency), but the architecture is well suited for parallelization, due to the fact that each proposed game log is validated independently. Scaling up the corpus by orders of magnitude will require a more distributed, parallelized approach to searching for proposed plans, evolving even closer to the architecture of a search engine.

Increasing the corpus size is an example of scaling one dimension of the system. There are other dimensions that can scale as well, including the number of characters, and the scope of the interaction. *Improviso* collected data from a game where each scene can include up to five characters. Future work remains to generate a game from this data, but as long as each role is distinct, the system theoretically scales to more than two characters. The scope of all of the data collection games is limited to face-to-face interaction in small environments, within a relatively short game (about 10 to 20 minutes). An interesting future direction would be to explore applying CAI to a storytelling layer that sits above the face-to-face interactions, and sequences plot points. While the current source of data is recorded demonstrations, the system itself is agnostic about the source, provided that the data is represented as a corpus of linear sequences, which can be abstracted into events. For this storytelling layer, the data might come from linear sequences of plot points authored by writers, which are then tagged and clustered by annotators. The approach to annotation might draw insights from story generation systems like *Mexica*.

A game with a broader scope may introduce additional challenges, such as dynamic role bindings. For example, when a player exits a restaurant, he is no longer a customer. From the perspective of other NPCs in the world, the former customer may now be a mailman, a suspect, or a father. A related issue is the generalization of content. Many of the actions and utterances in the TRG corpus are specific to restaurant interactions, but some are generally applicable to any social interaction, while others are applicable to any scenario involving a transaction between a buyer and seller of food (e.g. a grocery store). As future work, it should be possible to tag instances of events with labels indicating their scope,

with regards to generalization to other possible scenarios, resulting in large cross-domain corpora of content for social interaction in various contexts.

7.3.4   Development, Tuning, and Debugging

The CAI system, as described in this thesis, is a first implementation, and much has been learned in the process. There are numerous opportunities to improve the authoring and debugging processes, as well as tuning the runtime experience. The authoring process could be made more inclusive and distributed in two ways: transforming annotation from an outsourced task to a truly crowdsourced task, and formalizing the encoding of domain knowledge to eliminate, or at least reduce, the amount of programmer involvement. Hiring annotators through oDesk.com demonstrated that it is easy to hire people anywhere in the world to tag data, and this work requires minimal training. The experience with oDesk can be considered a pilot test, refining the process in preparation for true crowdsourcing in the future (e.g. through Amazon.com's *Mechanical Turk*). Currently, annotators tag entire game logs with the full set of event tags. This task might be broken down to per-event tagging tasks, with each game log distributed to multiple crowdsourced workers at once. Another promising future direction is automating or semi-automating repetitive aspects of the annotation process, possibly leveraging techniques described in Chapter 4.

Encoding domain knowledge is currently an ad hoc process where validation functions are programmed by hand in Java. Reviewing the domain knowledge implemented for all of the events in TRG, various design patterns and repetition of validation functions have emerged. Future work will focus on centralizing different types of validation, so that domain knowledge can be represented with configuration files, or a domain-specific language, accessible to non-programmers.

Debugging an open-ended, data-driven system that delivers unscripted NPC behavior is a time consuming process. After one month of full-time debugging for TRG, the game is not entirely bug free. Section 5.5 describes a number of mundane suggestions to facilitate debugging. All of these suggestions can be further refined to deliver significant decreases in debugging time. Suggestions include leveraging statistical techniques to preemptively catch tagging inconsistencies, generating logs of NPC-NPC games faster than real-time, and analyzing these logs for suspected bugs, as well as submitting logs for inspection by humans. For the thesis, the quality assurance process was run at a small scale, with four outsourced testers playing TRG, while I fixed bugs myself. Embracing game development as an iterative process, where games are living systems evolving online, and deploying games to many players online sooner rather than later, may provide opportunities to engage the community in the QA process, and take advantage of the economy of scale. With more players online, it may be easier to identify, and prioritize which issues to fix first, given limited resources for debugging. Recognition of recurring problems might even be possible to automate.

Finally, future work remains to improve one of the key components supporting human interaction, the system that dynamically generates dialogue options. Currently, options are sorted through a simplistic approach. If the set of keywords in the input utterance maps to keywords in a U-SET, then the validated dialogue options are sorted to maximize the number of matching input words. Otherwise, if proposed options are retrieved by context, options are sorted arbitrarily, shuffling them to ensure that each page of options offers a variety of utterances representing different events. Other factors that could be incorporated in the future include the likelihood of different utterance types, the redundancy of the

utterance (with consideration for similar utterances that reference different concepts), whether an utterance could respond to an unanswered question, and possibly other factors.

7.4   Future Work

This thesis has demonstrated how CAI can amplify the content creation productivity of one graduate student, to produce an experience with thousands of possible actions and lines of dialogue. However, this first exploration only scratches the surface of what may be possible, given more time and resources. Following are several thoughts on broader potential applications of CAI.

7.4.1   New Data Sources

Online games are one obvious source of data, but are not the only possibility. For instance, one can imagine recording humans interacting physically through the Kinect, while recording spoken dialogue. The pattern discovery process would need to be expanded to recognize actions from continuous skeletal animation, and segment utterances from (automatic or human) transcribed speech, but it is not inconceivable that this may be possible someday. Our world is increasingly being covered with sensors, and pervasive mobile devices are capturing tremendous amounts of data about our lives -- photos, emails, phone calls, GPS, etc. In sense, life is becoming a game. One can imagine aggregating this data to automate role-playing characters in the future from real-world data.

7.4.2   Decision Support in the Real World

Current research has focused on automating role-playing NPCs from crowdsourced data, but the medium produced through the CAI process could be used in other ways. The natural language interface implemented for TRG presents contextually relevant dialogue options based on human input, and players avatar executes the selected option in the game world. A similar system could act as a support system for people making decisions in the real world. Perhaps options could be displayed through a head-mounted display, or simply displayed on a hand-held mobile device. This system could be configured to suggest utterances at any time without requiring input, and could suggest physical actions as well. Making decisions in the real-world relies on being able to perceive actions and utterances, but as the previous section explores, there are an increasing variety of options for sensing interactions in the world. Translating sensor input into discreet observations, as input to the decision making system, may require human assistance, or may be possible to semi- or fully automate in the future. While decision support systems may have applications in business or the military, they could also be powerful tools for people struggling with neurological disorders affecting social interaction, like autism, or for travelers negotiating life in a foreign language.

7.4.3   Simulating Real-World Role-Playing Scenarios

Over the years of pursuing this research, a variety of people from different fields have proposed applications of simulated role-playing for solving real world problems. This section reports briefly on a selection of these potential applications. Medical students today receive instruction on how to communicate difficult information to patients and their families (e.g. cancer diagnoses) through interaction with specially trained, live, improvisational role-playing actors. Simulating such an interaction could offer several advantages over live role-players. In addition to being more cost-effective, simulation would allow students to spend more time replaying the material, ensure consistent lessons between students, and capture data about student performances. Perhaps simulation could complement live instruction, making time with human instructors more valuable. A similar argument exists for large corporations, with high turnover and training needs. Simulated role-playing could replace or supplement live role-playing or less interactive training videos, teaching subjects like interviewing skills, salary negotiations, and conducting performance reviews. Simulated role-playing has also been suggested as a way for corporations to communicate with and educate consumers. For example, someone who has never applied for a mortgage before might feel intimidated by the process, and could interact with a simulation before meeting with the bank to better understand what to expect. Practicing social interaction in a safe simulated setting before a real interaction has also been suggested as part of therapy for those diagnosed with autism or Asperger's. There is a counter argument, that autistics do not generalize well from games to the real world, however the spontaneous, replayable, highly varied nature of a data-driven interaction may be able to overcome this concern. As another example of social interaction practice, there has been interest from law enforcement agencies in using face-to-face simulated role-playing to practice non-violent conflict resolution. Finally, as robots become common in the home and workplace, their ability to cooperate as a peer in everyday activities becomes increasingly important. The Mars Escape project has already taken a step toward demonstrating the value of data-driven approaches in automating robots who can serve as better teammates.

7.4.4   Improvisational Interactions between Heterogeneous NPCs

The CAI process democratizes the content authoring process, producing improvisational NPCs compositing performances from thousands of human performances. The research thus far has focused on NPCs interacting with humans, primarily using NPC-NPC interactions as a means of testing the behavior before spending human testing cycles. In addition, current research explores automating NPCs in the same scenario in which the data was recorded. There are two promising directions for interactive storytelling, yet to be explored. One of these directions is interactive storytelling that focuses on interactions between heterogeneous NPCs, powered by data collected from different populations, and different environments. Is it possible to deliver a coherent story in a world populated with NPCs who have essentially never met before, but all draw upon large corpora of human performances in relevant scenarios? The second promising direction focuses on NPC-NPC interactions in persistent worlds. Videogames typically spawn NPCs (friends or foes) near the player at opportune moments, for practical reasons of limited processing resources. If the higher-level thinking of NPCs moves onto CAI servers on the cloud, there is the potential to allow these characters to continue going about their lives, interacting and conversing with one another. Data-driven NPCs dynamically conversing, drawing from thousands of possible lines of dialogue, could discuss recent observations and future plans. One can imagine new experiences exploiting this persistent simulation, for example a player in a stealth game might be able to

hack into enemy radio chatter, eavesdropping as they discover evidence of the player's previous actions, to learn of and foil their future plans.

7.5   Final Thoughts

Compared to other mediums, such as books and film, interactive mediums for storytelling are young, and continuing to rapidly evolve. No one can predict what interactive storytelling will look like three years from now, let alone 10 or 20. Given the trajectory seen in other fields, it seems inevitable that data-driven approaches will transform the way interactive stories are created, and the experiences they deliver. This thesis has shown how an online, data-driven, distributed development process can amplify the productivity of one individual developer to author an enormous amount of content, supporting open-ended simulated social role-play, in a small environment with two characters. What would be possible if the CAI process was scaled up by multiple orders of magnitude? It is not inconceivable that one day, we will reach a point where traces exist on the cloud of every game played, forming dense coverage of possible behavior and language. This thesis began with a discussion of the limitations of human imagination. Interestingly, this same limitation prevents any one person from fully imagining the future of interactive storytelling. In the world of videogames, there is a trend toward thinking of games as living systems, which communities evolve over time, rather than static immutable releases. It is likely that the future of storytelling will not come from an individual designer, researcher, or studio, but instead will emerge through a collective pursuit of new experiences. The interactive storytelling medium produced through the CAI process was conceived with this collective pursuit in mind. The future of this medium is in the crowd.

Appendix A: Event Tagging Tutorial

Event Tagging Tutorial (page 1 / 9)

You will be tagging events in transcripts from a pairs of players in a restaurant video game.

Players type chat text and interact with objects through a point-and-click interface (pictured below). Behavior of players ranges from typical restaurant interactions to bizarre misbehavior. Misbehavior may include off-topic dialogue, eating flowers, theft, etc. A game lasts about 10-15 minutes.
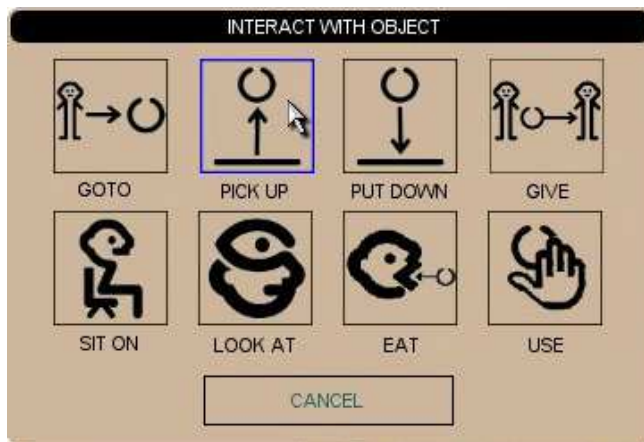
Below is a video of a typical experience in the game:

Event Tagging Tutorial (page 2 / 9)

The interface will display each transcript as a timeline of actions (in yellow boxes). You can scroll the timeline using the scroll bars, arrow keys, or by dragging the canvas while holding down your mouse button.

CUSTOMER ENTERS Restaurant	WAITRESS: " hello sir	CUSTOMER: " hello	WAITRESS: " table for one?	CUSTOMER: " yes, please	WAITRESS: " follow me and i will have u seated
[0]	[1] "	[2] "	[3] "	[4] "	[5] "

Your task will be to tag events by grouping action sequences in colored boxes. To draw a box, hold CTL while clicking the first action, and while continuing to hold CTRL click the last action in the sequence. While holding CTRL, you can scroll the canvas to access off-screen actions. Boxes can be moved, deleted, or overwritten as needed.

C_ARRIVES

CUSTOMER ENTERS Restaurant	WAITRESS: " hello sir	CUSTOMER: " hello
[0]	[1] "	[2] "

C_GETS_SEATED

WAITRESS: " table for one?	CUSTOMER: " yes, please	WAITRESS: " follow me and i will have u seated
[3] "	[4] "	[5] "

166
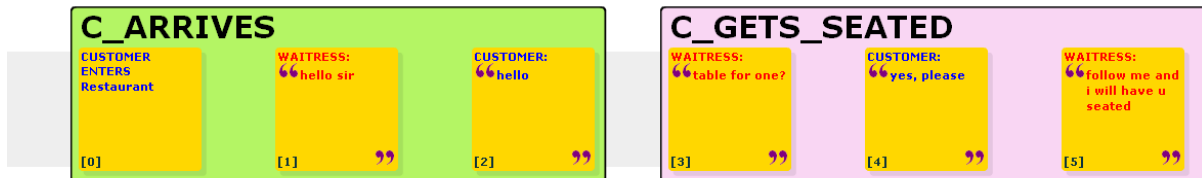
Event Tagging Tutorial (page 3 / 9)

To set the label on a box:

> 1) Click the box to select it (it will highlight with a red border).
>
> 2) Hold SHIFT and press the right and left arrow keys to toggle the label.
>
> - OR -
>
> Hold SHIFT or CTRL, and click a label in the list on the right-hand side of the screen.

You will be provided with a list of definitions and examples for all event labels.

EVENTS ▲
C_ARRIVES
C_COMPLAINS_BILL
C_COMPLAINS_FOOD
C_COMPLAINS_ORDER
C_COMPLIMENTS_FOOD
C_DECLINES_ORDER
C_DEPARTS
C_DRINKS
C_EATS_FOOD
C_GETS_BILL
C_GETS_MENU
C_GETS_SEATED
C_GETS_UP
C_ORDERS
C_PAYS_BILL
DELAY_ORDER
DISCUSS_AGE
DISCUSS_GEOGRAPHY
DISCUSS_MENU
DISCUSS_NAMES
OTHER
SOCIALIZE
W_CLEANS_TABLE

Event Tagging Tutorial (page 4 / 9)

Sometimes multiple events overlap. The interface allows you to move actions up and down to separate the events. Actions can only move vertically. Below is the same timeline before and after moving actions.

BEFORE:

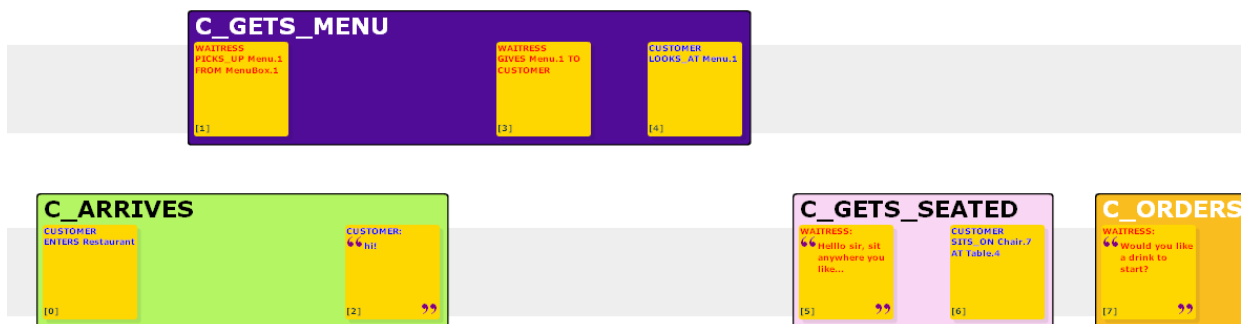| CUSTOMER ENTERS Restaurant	WAITRESS PICKS_UP Menu.1 FROM MenuBox.1	CUSTOMER: 66 hi!	WAITRESS GIVES Menu.1 TO CUSTOMER	CUSTOMER LOOKS_AT Menu.1	WAITRESS: 66 Hello sir, sit anywhere you like...	CUSTOMER SITS_ON Chair.7 AT Table.4	WAITRESS: 66 Would you like a drink to start?
[0]	[1]	[2] 99	[3]	[4]	[5] 99	[6]	[7] 99

AFTER:

C_GETS_MENU

WAITRESS PICKS_UP Menu.1 FROM MenuBox.1	WAITRESS GIVES Menu.1 TO CUSTOMER	CUSTOMER LOOKS_AT Menu.1
[1]	[3]	[4]

C_ARRIVES

CUSTOMER ENTERS Restaurant	CUSTOMER: 66 hi!
[0]	[2] 99

C_GETS_SEATED

WAITRESS: 66 Hello sir, sit anywhere you like...	CUSTOMER SITS_ON Chair.7 AT Table.4
[5] 99	[6]

C_ORDERS

WAITRESS: 66 Would you like a drink to start?
[7] 99

Event Tagging Tutorial (page 5 / 9)

It does not matter where boxes are placed vertically. Below is the same timeline with two different vertical event placements. These two examples are equivalent.

EXAMPLE 1:

C_GETS_MENU

WAITRESS PICKS_UP Menu.1 FROM MenuBox.1 [1]

WAITRESS GIVES Menu.1 TO CUSTOMER [3]

CUSTOMER LOOKS_AT Menu.1 [4]

C_ARRIVES

CUSTOMER ENTERS Restaurant [0]

CUSTOMER: "hi! [2] "

C_GETS_SEATED

WAITRESS: "Hello sir, sit anywhere you like... [5] "

CUSTOMER SITS_ON Chair.7 AT Table.4 [6]

C_ORDERS

WAITRESS: "Would you like a drink to start? [7] "

EXAMPLE 2:

C_ARRIVES

CUSTOMER ENTERS Restaurant [0]

CUSTOMER: "hi! [2] "

C_GETS_SEATED

WAITRESS: "Hello sir, sit anywhere you like... [5] "

CUSTOMER SITS_ON Chair.7 AT Table.4 [6]

C_GETS_MENU

WAITRESS PICKS_UP Menu.1 FROM MenuBox.1 [1]

WAITRESS GIVES Menu.1 TO CUSTOMER [3]

CUSTOMER LOOKS_AT Menu.1 [4]

C_ORDERS

WAITRESS: "Would you like a drink to start? [7] "
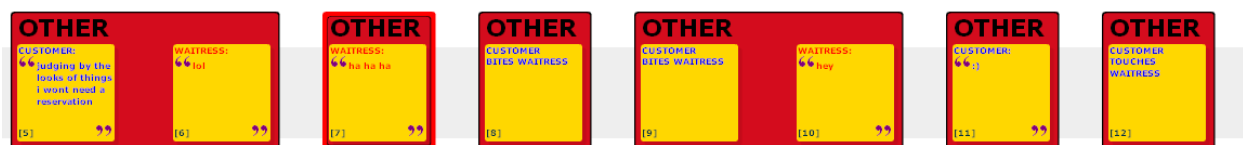
Event Tagging Tutorial (page 6 / 9)

Any action or sequence of actions that does not fit any of the event labels should be labeled OTHER. This includes nonsense behavior, incomprehensible dialogue (e.g. gibberish, foreign languages), as well as dialogue and actions that are not representative of typical restaurant behavior. For example, customers eating flowers is nonsense, so it is marked OTHER. Waitresses eating (rather than customers) is also atypical, and should be marked OTHER. When in doubt, use OTHER!

When marking actions as OTHER, it does not matter if they are grouped, or marked individually. Everything marked OTHER will be ignored by the systems that will use these annotations. Below is the same timeline with two different markings of OTHER. These two examples are equivalent.
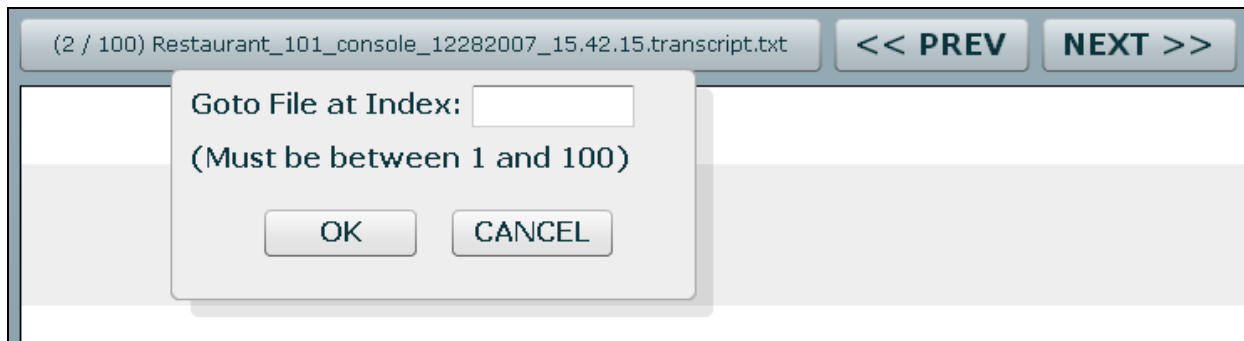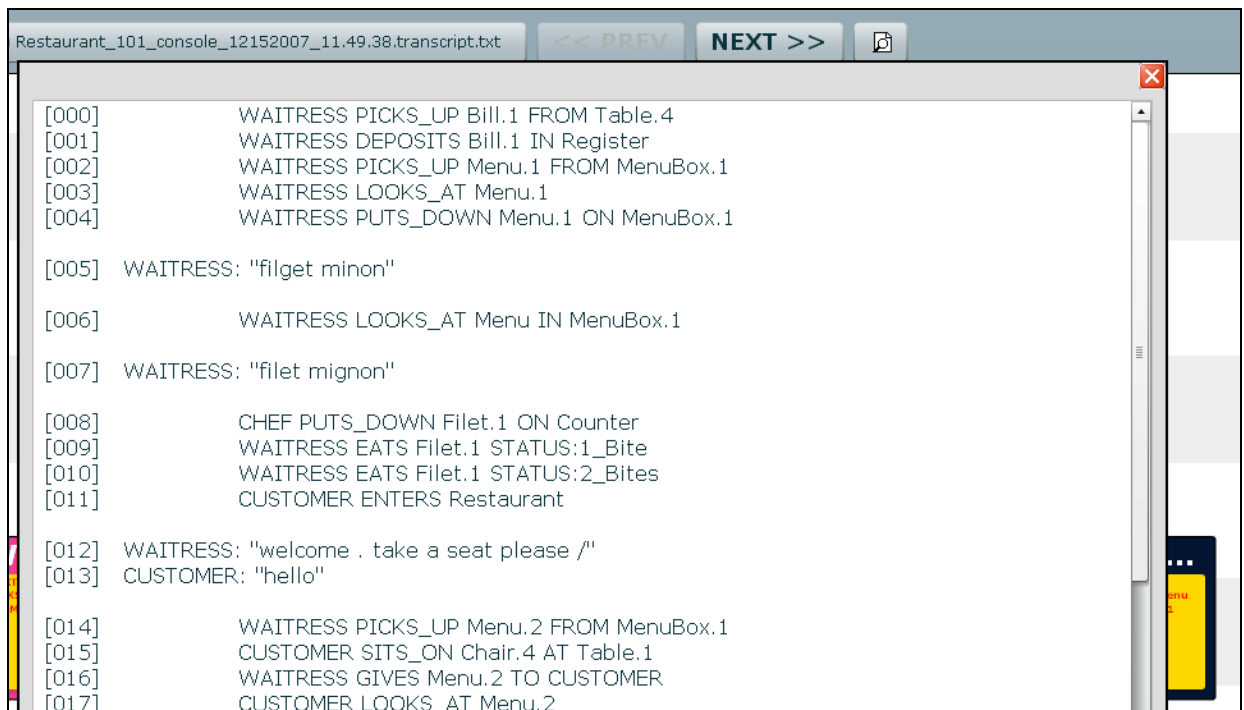
EXAMPLE 1:

EXAMPLE 2:

Event Tagging Tutorial (page 7 / 9)

You will be given a set of files to tag. You can advance through the files with the PREV and NEXT buttons. Clicking on the filename allows you to skip to a specific file.

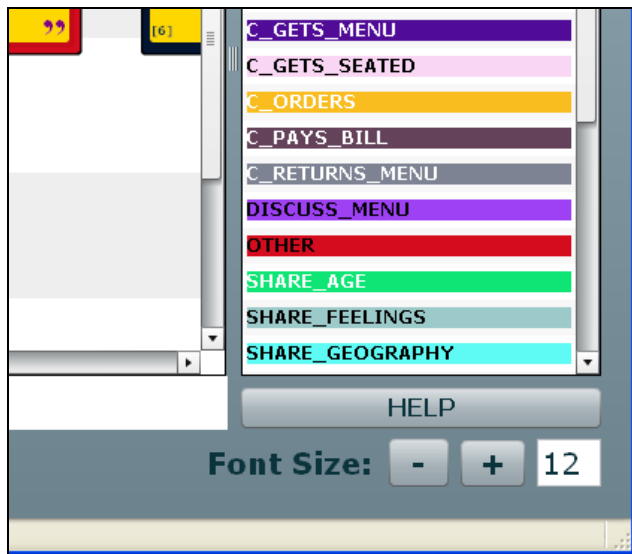The preview button allows you to view the entire transcript as a text file.
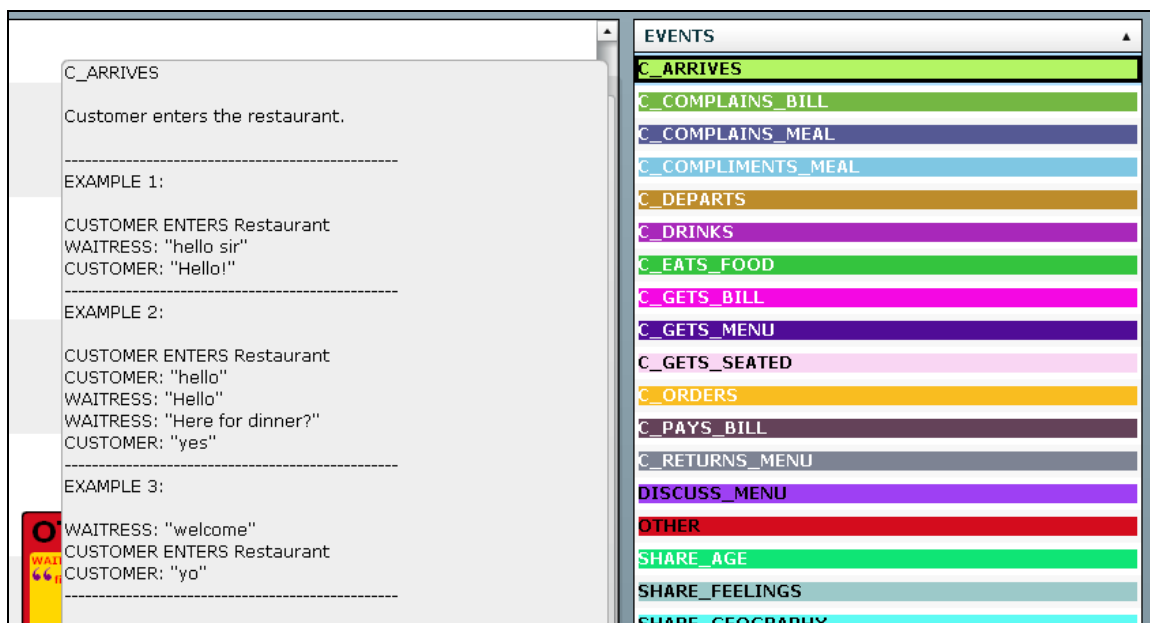
Event Tagging Tutorial (page 8 / 9)

Interface help and event definitions/examples are available anytime by clicking

To see the help page now, **click here**. (Opens in a separate window).

You can also get context-sensitive help by clicking the tilda key.
Move the mouse cursor over an event label, and hit tilde to get the
definition/examples in a pop-up window.

Event Tagging Tutorial (page 9 / 9)

You will be given a set of 10 files that have already been tagged, followed by file(s) to tag yourself. Examine the 10 files that are already tagged, and then begin tagging the next file. You do not have to complete tagging in one sitting. You can log into the system and continue at any time.

Save your work often by clicking the **SAVE** button, or hitting CTRL + S.

If you encounter any interface bugs (e.g. a box won't move, or a label won't change), the bugs should correct themselves if you just save your work, move to a different file, and return to the original file.

Read over the **event definitions and examples** before beginning.

When you are ready to begin, log in at:
http://krypton.media.mit.edu/annotate.swf

If you do not have a user name, contact **jorkin@media.mit.edu**
If you have any questions, email **jorkin@media.mit.edu**

Appendix B: Script to Generate Event Log Index

set TOOLS_DIR=..\..\torque_win\tools

// Compile every physical action in every game log into an ACTID.
java -classpath %TOOLS_DIR%\GenActIDs;%TOOLS_DIR%\BassShared GenActIDs .\ini_files\GenActIDs.ini

// Convert every utterance in every game log into a U-SETID.
java -classpath %TOOLS_DIR%\GenSignedDlg;%TOOLS_DIR%\BassShared GenSignedDlg
.\ini_files\GenSignedDlg.ini

// Identify who speaker is talking to, for every utterance in every log file.
java -classpath %TOOLS_DIR%\GenListeners2;%TOOLS_DIR%\BassShared GenListeners2
.\ini_files\GenListeners2.ini

// Merge time-coded ACTIDs, U-SETIDs, and listeners into single file per game log.
java -classpath %TOOLS_DIR%\DlgMergeListeners;%TOOLS_DIR%\BassShared DlgMergeListeners
.\ini_files\DlgMergeListeners.ini

// Merge human annotations with compiled ACTIDs and U-SETIDs.
java -classpath %TOOLS_DIR%\EventMergeAnnotation;%TOOLS_DIR%\BassShared EventMergeAnnotation
.\ini_files\EventMergeAnnotation.ini

// Generate Event Dictionary from annotated log files.
java -classpath %TOOLS_DIR%\EventDictionaryGen;%TOOLS_DIR%\BassShared EventDictionaryGen
.\ini_files\EventDictionaryGen.ini

// Identify who is executing every action and utterance in every log file.
java -classpath %TOOLS_DIR%\RoleTagger;%TOOLS_DIR%\BassShared RoleTagger .\ini_files\RoleTagger.ini

// Extract semantic frames from each utterance (e.g. concepts for food items).
java -classpath %TOOLS_DIR%\GenSemFrames;%TOOLS_DIR%\BassShared GenSemFrames
.\ini_files\GenSemFrames.ini

// Extract human readable action descriptions and utterances from compiled log files.
java -classpath %TOOLS_DIR%\GenDetails;%TOOLS_DIR%\BassShared GenDetails .\ini_files\GenDetails.ini

// Extract list of time-codes per log file.
java -classpath %TOOLS_DIR%\GenTimecodes;%TOOLS_DIR%\BassShared GenTimecodes
.\ini_files\GenTimecodes.ini

// Merge compiled logs, semantic frames, time-codes, action descriptions, and utterances into index.
java -classpath %TOOLS_DIR%\GenLogEventIndex;%TOOLS_DIR%\BassShared GenLogEventIndex
.\ini_files\GenLogEventIndex.ini

Appendix C: Configuration for Goals and Critics

// Max pressure before complete planning failure.

max_pressure = 3

// Critics, sorted by priority.

Critic0 = "SynthCriticRequiredRole, -1, -1"
Critic1 = "SynthCriticNoReruns, -1, -1"
Critic2 = "SynthCriticInvalidatedAction, -1, -1"
Critic3 = "SynthCriticDegenerateEvent, -1, -1"
Critic4 = "SynthCriticCensor, -1, -1"
Critic5 = "SynthCriticAttitude, -1, -1, BIAS"
Critic6 = "SynthCriticPersonalityFallback, -1, -1"
Critic7 = "SynthCriticStructure, -1, -1"
Critic8 = "SynthCriticStaleDialogue, -1, -1"
Critic9 = "SynthCriticCausalChain, -1, -1"
Critic10 = "SynthCriticResourceConflict, -1, -1"
Critic11 = "SynthCriticDomain, -1, -1"
Critic12 = "SynthCriticReference, -1, -1"
Critic13 = "SynthCriticReservedLog, -1, -1, BIAS"
Critic14 = "SynthCriticHumanInput, -1, -1"

// Goals, sorted by priority.

Goal0 = "SynthGoalRespondToSequence, -1, -1"
Goal1 = "SynthGoalExtendCompletedEvent, -1, -1"
Goal2 = "SynthGoalRespondToCompletedEvent, -1, -1"
Goal3 = "SynthGoalCompleteEvent, -1, -1"
Goal4 = "SynthGoalCompleteCausalChain, -1, -1"
Goal5 = "SynthGoalExtendScenario, -1, -1"
Goal6 = "SynthGoalExtendStructure, 2, -1"

Bibliography

Abbott, H.P.2002. *The Cambridge Introduction to Narrative*. Cambridge University Press.

Ambertation. 2004. *SimPE: The Sims 2 Package Editor*. http://sims.ambertation.de.

Argall, B.D., Chernova, S., Veloso, M., and Browning, B. 2009. A survey of robot learning from demonstration. *Robotics and Autonomous Systems,* 57(5): 469-483.

Austin, J.L. 1962. *How to Do Things with Words: The William James Lectures delivered at Harvard University in 1955*. (ed. J. O. Urmson) Oxford: Clarendon Press.

Baldwin, D.A., Baird, J.A., Saylor, M.M., and Clark, M.A. 2001. Infants parse dynamic action. *Child Development*, 72(3): 708-717.

Barzilay, R., and Lapata, M. 2005. Modeling local coherence: an entity-based approach. In *Proceedings of the Fourty Third Annual Meeting on Association for Computational Linguistics*.

Bates, E. 1979. *The Emergence of Symbols: Cognition and Communication in Infancy*. Academic Press.

Bates, J. 1992. Virtual reality, art, and entertainment. *Presence: The Journal of Teleoperators and Virtual Environments,* 1(1): 133-138.

Baum, L. E.; Petrie, T. 1966. Statistical inference for probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics*, 37(6): 1554-1563.

Bethesda Game Studios. 2011. *The Elder Scrolls V: Skyrim*. Bethesda Softworks.

BioWare. 2012. *Mass Effect 3*. Electronic Arts.

Brown, P.F., Della Pietra, V.J., Della Pietra, S.A., and Mercer, R.L. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics,* 19(2): 263-311.

Bruner, J. 1977. Early social interaction and language acquisition. In H.R. Schaffer (Ed.), *Studies in Mother-Infant Interaction*. New York: Academic.

Bungie. 2001. *Halo: Combat Evolved*. Microsoft Game Studios.

Cavazza, M., Charles, F., and Mead, S.J. 2002. Sex, lies and videogames: An interactive storytelling prototype. In *Proceedings of the AAAI 2002 Symposium on Artificial Intelligence and Interactive Entertainment*.

Chernova, S., Orkin, J., and Breazeal, C. 2010. Crowdsourcing HRI through online multi-player games. In *Proceedings of the AAAI Fall Symposium on Dialog with Robots*.

Chernova, S., DePalma, N., Morant, E., and Breazeal, C. 2011. Crowdsourcing human-robot interaction: Application from virtual to physical worlds. In *Proceedings of the Twentieth IEEE Symposium on Robot and Human Interactive Communication*.

Chumbalum Soft. 1996. *Milkshape 3D*. http://chumbalum.swissquake.ch.

Cortes, C., and Vapnik, V.N. 1995. Support-Vector Networks. *Machine Learning*, 20(3): 273-297.

Cover, T.M., and Thomas, J.A. 1991. *Elements of Information Theory*. John Wiley & Sons, Inc.

CryTek Frankfurt. 2007. *Crysis*. Electronic Arts.

Earley, J. 1970. An efficient context-free parsing algorithm. *Communications of the ACM,* 6(8):451–455.

Erol, K., Hendler, J., and Nau, D. 1994. HTN planning: Complexity and expressivity. In *Proceedings of the Twelth National Conference on Artificial Intelligence* (NCAI 1994).

Ezeife, C. I., Lu, Y., and Liu, Y. 2005. PLWAP sequential mining: open source code. In *Proceedings of the First International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations*. ACM.

Fasciano, M.J. 1996. *Mayor*: Plan Use in an Everyday World. PhD Thesis, University of Chicago.

Fikes, R.E. and Nilsson, N.J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3-4): 189-208.

Fleischman, M. and Hovy, E. 2006. Taking advantage of the situation: Non-linguistic context for natural language interfaces to interactive virtual environments. In *Proceedings of the International Conference on Intelligent User Interfaces* (IUI 2006).

Fredkin, E. 1960. Trie memory. *Communications of the ACM* 3(9): 490–499.

Frey, B.J. and Dueck, D. 2007. Clustering by passing messages between data points. *Science*, 315(5814): 972–976.

Furnas, G.W., Landauer, T.K., Gomez, L.M., and Dumais, S.T. 1987. The vocabulary problem in human-system communications. *Communications of the ACM,* 30(11): 964-971.

Garage Games. 2006. *Torque Game Engine v1.5*. http://www.garagegames.com.

George, J. and Langley, P. 1995. Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*.

Gordon, A., van Lent, M., van Velsen, M., Carpenter, P., and Jhala, A. 2004. Branching storylines in virtual reality environments for leadership development. In *Proceedings of the Sixteenth Innovative Applications of Artificial Intelligence Conference on Artificial Intelligence*. (IAAI04).

Gorin, A., Riccardi, G., Wright, J. 1997. How may I help you? *Speech Communication,* 23(1-2): 113-127.

Gorniak, P. and Roy, D. 2005. Probabilistic grounding of situated speech using plan recognition and reference resolution. In *Proceedings of the Seventh International Conference on Multimodal Interfaces* (ICMI 2005).

Hammond, K.F. 1990. Case based planning: A framework for planning from experience. *Cognitive Science*, 14(3): 385–443.

Hanson, P. and Rich, C. 2010. A non-modal approach to integrating dialogue and action. In *Proceedings of the of Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (AIIDE 2010).

Hewlett, D., Hoversten, S., Kerr, W., Cohen, P., and Chang, Y.H. 2007. Wubble world. In *Proceedings of the Third AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (AIIDE 2007).

Insomniac Games. 2006. *Resistance: Fall of Man*. Sony Computer Entertainment.

Joachims, T. 2008. *SVMhmm: Sequence Tagging with Structural Support Vector Machines*. http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html.

Jones, K.S. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation,* 28 (1): 11–21.

Jurafsky, D. and Martin, J. 2000. *Speech and Language Processing*. Prentice Hall.

Kautz, H. and Allen, J. 1986. Generalized plan recognition. In *Proceedings of the National Conference on Artificial Intelligence* (NCAI 1986).

Kolodner, J., 1983. Reconstructive memory: A computer model. *Cognitive Science,* 7: 281-328.

Lenat, D. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM,* 38(11): 33-38.

Li, B., Lee-Urban, S., Appling, D.S., and Riedl, M.O. 2012. Crowdsourcing narrative intelligence. *Advances in Cognitive Systems,* 2: 25-42.

Lionhead Studios. 2001. *Black & White*. Eletronic Arts.

Lionhead Studios. 2005. *Black & White 2*. Electronic Arts.

Liu, H. and Singh, P. 2004 ConceptNet: A practical commonsense reasoning toolkit. *BT Technology Journal*, 22(4): 211-226.

Lochbaum, K.E. 1998. A collaborative planning model of intentional structure. *Computational Linguistics,* 24(4): 525–572.

Magerko, B., Dohogne, P., and DeLeon, C. 2011. Employing fuzzy concepts for digital improvisational theatre. In *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (AIIDE 2011).

Manning, C.D. and Schutze, H. 2000. *Foundations of Statistical Natural Language Processing*. MIT Press.

Mateas, M. 2002. *Interactive Drama, Art and Artificial Intelligence*. PhD Thesis, Carnegie Mellon University.

Mateas, M., and Stern, A. 2005. Procedural authorship: A case study of the interactive drama Façade. In *Proceedings of Digital Arts and Culture* (DAC 2005).

Maxis. 1989. *SimCity*. Brøderbund.

Maxis. 2000. *The Sims*. Electronic Arts.

Maxis. 2004. *The Sims 2*. Eletronic Arts.

McCoy, J., Treanor, M., Samuel, B., Wardrip-Fruin, N., and Mateas, M., Comme il Faut: A system for authoring playable social models. In *Proceedings of the Seventh Artificial Intelligence and Interactive Digital Entertainment* (AIIDE 2011).

Meehan, J. 1976. *The Metanovel: Writing Stories by Computer*. PhD Thesis, Yale University.

Minsky, M. 1974. *A Framework for Representing Knowledge*. MIT-AI Laboratory Memo 306.

Ontañón, S., Mishra, K., Sugandh, N., Ram, A. 2007. Case-based Planning and Execution for Real-time Strategy Games. In *Proceedings of the Seventh International Conference on Case-Based Reasoning* (ICCBR 2007).

Ontañón, S., and Zhu, J. 2010 Story and text generation through computational analogy in the Riu system. In *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (AIIDE 2010).

Orkin, J. 2007. *Learning Plan Networks in Conversational Video Games*. MS Thesis, Massachusetts Institute of Technology.

Orkin, J. and Roy, D. 2009. Automatic learning and generation of social behavior from collective human gameplay. In *Proceedings of the Eigth International Conference on Autonomous Agents and Multiagent Systems* (AAMAS 2009).

Orkin, J. and Roy, D. 2010. Semi-Automated Dialogue Act Classification for Situated Social Agents in Games. In *Proceedings of the Second AAMAS Agents for Games and Simulations Workshop* (AGS 2010).

Orkin, J., Smith, T., Reckman, H., and Roy, D. 2010. Semi-automatic task recognition for interactive narratives with EAT & RUN. In *Proceedings of the Third Intelligent Narrative Technologies Workshop* (INT3).

Orkin, J. and Roy, D. 2012. Understanding speech in interactive narratives with crowdsourced data. In *Proceedings of the Eigth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (AIIDE 2012).

Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Fourtieth Annual Meeting of the Association for Computational Linguistics* (ACL 2002).

Pérez y Pérez, R., and Sharples, M. 2001. MEXICA: a computer model of a cognitive account of creative writing. *Journal of Experimental and Theoretical Artificial Intelligence*, 13(2): 119-139.

Prince, G. 1987. *A Dictionary of Narratology*. University of Nebraska Press.

QuArK. 2001. *Quake Army Knife*. http://quark.planetquake.gamespy.com.

Reckman, H., Orkin, J., and Roy, D. 2010. Learning meanings of words and constructions, grounded in a virtual game. In *Proceedings of the Tenth Conference on Natural Language Processing* (KONVENS 2010).

Ryan, M. 1991. *Possible Worlds, Artificial Intelligence, and Narrative Theory*. Indiana University Press.

Ryan, M. 2006. *Avatars of Story*. University of Minnesota Press.

Schank, R., and Abelson, R. 1977. *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*. Lawrence Erlbaum Associates.

Schank, R. 1983. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. Cambridge University Press.

Schank, R. 1995. *Tell Me a Story: Narrative and Intelligence (Rethinking Theory)*. Northwestern University Press.

Schank, R. 1999. *Dynamic Memory Revisited*. Cambridge University Press.

Searle, J.R. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

Sengers, P. 1999. Designing comprehensible agents. In *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI 1999).

Singapore-MIT GAMBIT Game Lab. 2011. *Improviso*. http://gambit.mit.edu/loadgame/improviso.php

Singh, P., Lin, T., Mueller, E.T., Lim, G., Perkins, T., and Zhu, W.L. 2002. Open Mind Common Sense: Knowledge acquisition from the general public. In *Proceedings of the International Conference on Ontologies, Databases and Applications of Semantics* (ODBASE 2002).

Singh, P., and Williams, W. 2003. LifeNet: A propositional model of ordinary human activity. In *Proceedings of the Workshop on Distributed and Collaborative Knowledge Capture*.

Singh, P. 2005. *EM-ONE: An Architecture for Reflective Commonsense Thinking*. PhD Thesis, Massachusetts Institute of Technology.

Smith, T. 2011. *Unsupervised Discovery of Human Behavior and Dialogue Patterns in Data from an Online Game*. M.Eng EECS Thesis, Massachusetts Institute of Technology.

Surendran, D. and Levow, G. 2006. Dialog act tagging with support vector machines and hidden markov models, In *Proceedings of Interspeech*.

Swanson, R., and Gordon, A. 2010. A data-driven case-based reasoning approach to interactive storytelling. In *Proceedings of the International Conference on Interactive Digital Storytelling* (ICIDS 2010).

Swanson, R. and Gordon, A. 2012. Say Anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Transactions on Intelligent Interactive Systems,* 2(3).

Turner, S. R. 1994. *The Creative Process: A Computer Model of Storytelling and Creativity*. Lawrence Erlbaum Associates.

von Ahn, L., and Dabbish, L. 2004. Labeling images with a computer game. In *Proceedings of ACM Conference on Human Factors in Computing Systems* (CHI 2004).

von Ahn, L. 2006. Games with a purpose. *Computer* 39(6): 92–94.

Vygotsky, L. 1978. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press.

Wardrip-Fruin, N. 2009. *Expressive Processing: Digital Fictions, Computer Games, and Software Studies*. MIT Press.

Wittgenstein, L. 1965. *The Blue and Brown Books*. Harper Torchbooks.

Young, R.M., Riedl, M.O., Branly, M., Jhala, A., Martin, R.J., and Saretto, C.J. An architecture for integrating plan-based behavior generation with interactive game environments. *Journal of Game Development*, 1: 51–70.