The AIIDE-17 Workshop on
Intelligent Narrative Technologies
WS-17-20

Bardic: Generating Multimedia Narrative Reports for Game Logs

Camille Barot,[1] Michael Branon,[4] Rogelio E. Cardona-Rivera,[3] Markus Eger,[1] Michelle Glatz,[2] Nancy Green,[4] James Mattice,[4] Colin M. Potts,[1] Justus Robertson,[1] Makiko Shukonobe,[2] Laura Tateosian,[2] Brandon R. Thorne,[1] R. Michael Young[3]

[1] Department of Computer Science, North Carolina State University, Raleigh, NC, USA
[2] Center for Spacial Analytics, North Carolina State University, Raleigh, NC, USA
[3] School of Computing and the Entertainment Arts and Engineering Program, Salt Lake City, UT, USA
[4] Department of Computer Science, UNC Greensboro, Greensboro, NC, USA
camillebarot@gmail.com, {meger | mlglatz | cmpotts | jjrobert | mshukun | lgtateos | brthorne}@ncsu.edu,
{mabranon | jrmattic | nlgreen}@uncg.edu, {rogelio | young}@eae.utah.edu

Abstract

In this paper we present the system called *Bardic*, which was developed over three years as the core technology in the Narrative for Sensemaking project, an effort to automatically generate narrative from low-level event data as an aid for sense making. At its core, Bardic is a narrative report generator that uses a logic-based language to represent a story based on event/activity logs. Bardic generates different types of narrative discourse designed to convey different aspects of the underlying data. The system consists of a Unity application that allows users to explore generated stories and select parts of the stories to analyze in detail. In addition to its narrative generation capabilities, the application provides visualization and query capabilities. In this paper we provide screen shots of the application in use, show examples of narrative output produced by Bardic, detail how Bardic generates its output, and discuss the current system's capabilities and limitations.

Introduction

Narrative is one central way that people make sense of the world (Bruner 1991). In the Narrative for Sensemaking project, we are concerned with producing summary reports or *narrativizations* describing activity in complex domains, leveraging people's common cognitive orientation around narrative to make the reports more accessible to non-experts. To do so, we use a tripartite model of narratives as described by Barot et al. (2015), consisting of a story, a discourse, and a narration. In this paper we describe the tool Bardic and its associated web services and how they can be used to generate multimedia narratives characterizing game logs of the online multi-player game *Defense of the Ancients 2* (or DOTA 2 (Valve Corporation 2013)). The current design of the Bardic tool targets users seeking to make sense of DOTA 2 's low-level event logs (e.g., the large number of DOTA 2 fans that review competitive DOTA 2 logs). However, the tool and its capabilities are intentded to serve as proof-of-concept for a broader application, specifically the automated narrativization of large corpora of low-level event data, ultimately to increase users' understanding of that data.

DOTA 2 is a multiplayer online battle arena (MOBA) game in which two teams of five players compete in fantasy-style combat to destroy the other team's base. The game is played in a virtual outdoor arena roughly the size of four soccer fields. DOTA 2 presents a number of challenges for systems working to automatically generate narrative summaries of complex event sequences. In DOTA 2 , players often form, revise, and drop plans over time as they obtain new information. There are multiple factions and teamwork within each faction, but a team is not in the same location at all times, so the resulting narratives will feature a variable number of actors. Games typically last around 45 minutes, are self-contained, and follow a natural progression of phases. The early part of game is dedicated to building up resources followed by a more objective-oriented mid-game and a team fight-focused late-game. Log data from gameplay is at a very fine-grained, high frequency level focused on game state snapshots compared to the abstract narrative-oriented communicative style that people naturally use to communicate about game dynamics.

System Architecture

Bardic is a software application built with the Unity3D game engine[1] that generates multimedia narrative reports for DOTA 2 game logs. Our tool, Bardic, allows users to analyze a particular game, select subsets by time and characters of interest, and generate narrative reports about these subsets in the form of text, maps, and machinima (cinematic sequences generated using a 3D game engine). Figure 1 shows an overview of the system architecture. Bardic's pre-processing phase consists of three sub-processes: parsing the log files produced by DOTA 2 using the third-party libraries Skadi[2] and Tarrasque[3], translating the extracted in-

[1] https://unity3d.com/
[2] https://github.com/skadistats/skadi
[3] https://github.com/skadistats/Tarrasque

formation into a logical language, and automatically annotating the resulting knowledge base with information relating the actions of the characters within the game world to their respective intentions. The resulting knowledge base is then used by Bardic to provide query capabilities and visualizations as well as to drive the generation of the narrative. The process of integrating the event data into coherent *stories* and determining how to communicate these stories across media in *narrative discourse* is done by an external web service running on a centralized server.

Pre-processing

The pre-processing step takes a game log from DOTA 2 and translates it into Impulse, a logic-based language for the encoding of stories (Eger, Barot, and Young 2015). A story encoded in Impulse consists of three main parts: Actions, actants (i.e. actors and objects), and facts that hold in the world. In logical terms, the facts are logical sentences, the actions describe how facts change over time, and the actors and objects are what the facts are about. Every fact is indexed by the temporal interval over which it holds. In addition to the standard logic primitives, Impulse also has modalities to describe characters' beliefs, desires, and intentions during different time intervals (although the representation of beliefs and desires for DOTA 2 characters is not currently used in Bardic).

Game Log Parsing DOTA 2 automatically creates game log files for every game session. During gameplay, the game records snapshots of the game state, fully specifying all attributes of each character (e.g., location, health, money, items) 30 times per second and stores them in an associated game log file. Additionally, the game log also contains a list of every action that occurs in the game (e.g., attacks, player deaths, destroyed buildings). The game logs are encoded in a proprietary, closed binary format, requiring the use of one of several publicly available libraries to extract their contents.

The translation from game log data to Impulse is straightforward. The player and non-player characters referenced in the logs are translated to actors in the story; items are translated to objects; attacks and abilities used by characters are translated to story actions. Movement actions, which are not explicitly marked in the log file, are reconstructed from the location attribute of the characters by fitting line segments to the sequence of location data when a character moves more than a small distance away from their location. As long as they move within a small tolerance of a straight line at constant speed, they are considered to be performing the same movement action. Once they change direction or stop, a `move` action is added to the impulse file. Other static and time-dependent player properties like player location, team affiliation and character levels are also translated into logical sentences, an example of which can be seen in Listing 1. However, because DOTA 2 is a proprietary game not all information present in the replay file can be extracted by the third-party libraries. We therefore do not have certain information, like which trees characters cut down or all abilities of all available characters. However, we fully support over 25 characters with their abilities, and many of the other character's abilities in a generic way.

In addition to translating the actions present in the game log, the parsing process also infers causal relationships between actions. This is done by referring to action models that characterize DOTA 2 actions using a STRIPS-style (Fikes and Nilsson 1971) action representation, and then matching every action's preconditions with consistent effects asserted by the most recent prior actions. The action models come from hand-authored STRIPS operators specific to a given domain, in our case, DOTA 2. The process for determining causal connectivity is a greedy one, essentially tagging the temporally closest potential source for each causal relation as its source. For example, an ATTACK action has the precondition of being proximal to its target, while a MOVE action has the effect of positioning its moving character at a given location. To determine the causal source of an ATTACK action's proximity precondition, the system selects the MOVE action that precedes the ATTACK, that places the moving character in range of the ATTACK's target and that moves the same character involved in the ATTACK.

```
<predicate name=''alive''>
    <constant value=''creep_1520526''/>
    <constant value=''16798-19514''/>
</predicate>
```

Listing 1: A sentence expressing when a particular computer controlled character was alive.

Intention Recognition After parsing the contents of the game log into Impulse elements, the pre-processor adds annotations linking each action to the intention(s) that it helps achieve for the character performing the action. As a simple example, the action of player A attacking player B is performed as a means to achieve A's goal of having player B dead. Our pre-processing step groups temporally close actions that happen in pursuit of the same goal into an *intention frame*, following terminology introduced by Ware and Young (2011). The possible intentions for each action type and their maximum temporal extents, which determines how actions are grouped were defined by a domain expert. The intentions are added as logical sentences to the Impulse file that describe which actions are members of which frames, and what the intended goal of each frame is. Other aspects of narrative modeling (e.g., tracking character belief (Thorne and Young 2017) and the interplay between belief and intention revision (Young 2017) could also be added in future versions of the system; such additions would potentially increase the expressivity of the system by providing a representation for narrative situations where, for instance, characters come to hold mistaken beliefs that lead them to take up doomed courses of action.

Unity Application

The Bardic application first loads the Impulse files produced by the pre-processing steps. The user can then select a subset of the game indexed by time and characters, request that the system generate a corresponding narrative report (in any or all of the three media types) and, after inspecting the resulting report, refine or revise the system focus and request new

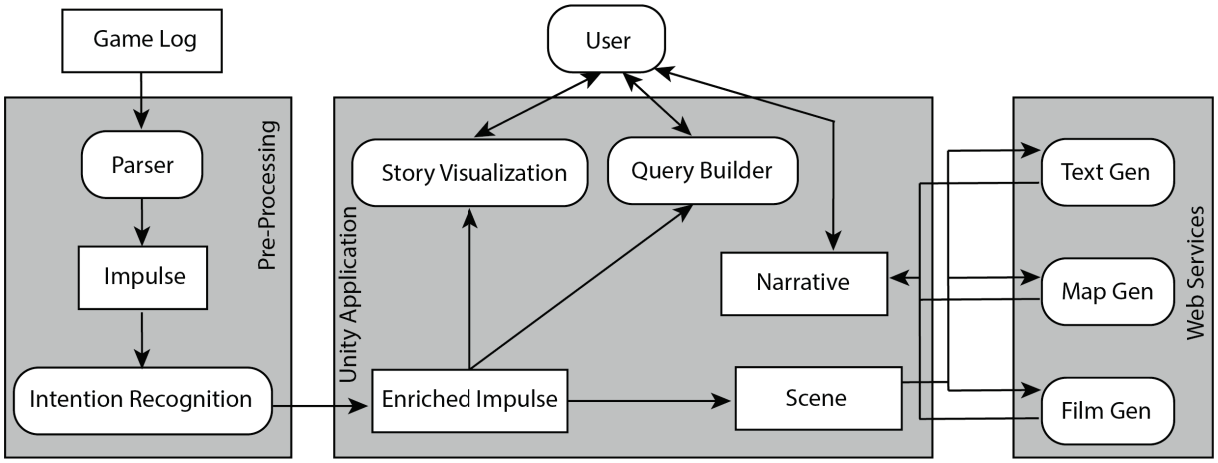

Figure 1: The system architecture for the Bardic system. In this figure, rectangles indicate data stores and ovals indicate processes. Data flow is indicated by the directed arcs.

Figure 2: Bardic's subset selection screen is used to constrain the game log content for which narrative reports are generated. The tool allows users to scrub through game activity over time, to view graphical characterization of game state variables and to restrict narrative generation to sub-stories by time and player participants.

Figure 3: Bardic's visualization of a generated story structure based on DOTA 2 log data. Rectangles in the diagram indicate actions, lines between actions indicate causal or temporal constraints and color is used to indicate sub-plans performed by distinct characters.

narrative reports. We have developed a desktop version of this interface as well as a multi-screen version that utilizes two table-top touch displays for user interaction and renders output on a 21'-wide wall-mounted display (the latter configuration is shown in Figure 7).

The first screen presented to a user after loading an Impulse file is the subset selection screen as seen in Figure 2. This screen shows how the game progresses over time. It contains static graphs that visualize key statistics over the course of the game as well as an animated map that displays the location of every character. The user can play back, pause, and scrub through time to see where the characters are, and use this information to select an interval of interest. The user can also restrict the characters that should be included in the selected subset by using the checkboxes. Once a user has selected a time interval and a set of characters, they can confirm the selection and all subsequent actions will only use the selected sub-story.

Story Visualization Viewing the underlying structure of actions, particularly how action types are distributed temporally and across actors, can yield insights into what is happening in the game. Our application therefore provides the capability to display that structure in a visual way. It shows each action as a box, with edges between them representing their causal relationships. The nodes are grouped by actor and are in temporal order from left to right with vertical offsets as necessary to prevent occlusions between actions that happen concurrently. Figure 3 shows an example of the visualization screen. Note that the user can include or exclude any actors in order to focus on interactions of interest.

Query Builder Since the representation we use for the story is based on a restricted temporal logic, it is possible to use an inference process defined over that language to answer logical queries. The query builder screen provides an interactive query builder for such queries. Our query builder works by allowing the user to choose which kind of query

they want to perform and specify the necessary parameters for each question type. Available question types are about who performed an action, where a character was located, which property an executed action had, and why an action was performed. Depending on the query's type, the other fields are then used to fill in particular action types, actors, or targets of the action in question. It is also possible to leave any of the fields blank to query any applicable action instance. To answer a query, the system will unify the query with entries in its knowledge base until a match is found, and respond with bindings to the variables in the query. Once presented with the response, the user may request that the system finds another answer, until the system tells the user that they have exhausted all possible answers. For example, one query that could be asked is "When Mirana healed anyone, what was the amount?". When the query is performed, the answer might be "37", together with the target of the healing action, e.g. "Skywrath Mage". The user may then request another answer, and get another entry of "44" with a target of "Crystal Maiden", and so on, until the system responds with "No more answers".

Data-Driven Narrative Generation and Narrative Web Services

The main functionality of our system is the generation of multimedia narratives for a subset of the game as selected in the UI. The generation approach follows a pipeline which takes as input a story, in our case the subset of the game, and some additional information as provided by the user in the UI, then produces a discourse plan that describes what should be communicated and then uses this plan to realize output in some medium. The general outline of this approach is well-established in the literature. The COMET system (Feiner and McKeown 1991), for example, generated instructions for maintenance and repair of military radio equipment using such a pipeline architecture. WIP is another early system with a similar architecture that was used to generate text and pictorial documents (Andre et al. 1993; Wahlster et al. 1993; André and Rist 1995). More recently, Molina and Flores (2012) describe a system that generates multimedia text, 2D graphics using sprites, and 3D animation summaries about the behavior of dynamic systems using a hierarchical planner, similar to how Bardic generates narratives. We build on these ideas, and add the capability to generate spatial maps.

The input to our narrative generation process consists of the selected subset of the game, the specification of the character or characters to focus on for the narrative, the constraints on the media types to generate and restrictions on which narrative trope to use to organize the narrative content.

At present, our system supports four narrative tropes to guide the generation of the narrative in different scenarios. In Bardic, tropes are hand-authored story schemas or patterns defined by system designers. These trope patterns match human-recognizable action sequences to the discourse conventions used to convey them. Future work will expand the set of tropes considered to address a broader range of narrative patterns. The currently supported tropes and how they guide the narrative generation are:

- **Chase Fight:** This trope is applicable in which a character flees from another character while being attacked.
- **Alpha Strike:** This trope refers to situations in which a group of characters attacks one or more enemies simultaneously.
- **Hard Work Montage:** This trope utilizes the encoded intentions to showcase situations in which a character must perform many actions to achieve a goal, for example repeatedly attacking and finally killing an enemy player.
- **Failure Montage:** In contrast to the hard work montage, this trope applies to cases in which a character tries to achieve some goal but ultimately fails, for example failing to kill an enemy player despite attacking them multiple times.

The narrative is then generated using planning to determine what to show to the audience and in which order, similarly to Jhala and Young (2010). The planning problem is set up as follows: the facts about the world and the game actions that occurred in the subset are encoded in the initial state, and the goal state consists of what to communicate. The operators in the planning domain are communicative primitives which depend on the medium to be generated. Once the planning process is complete, the resulting communicative primitives are used to request the actual media output from the particular medium realizer, as discussed below. The goal of the discourse planning process usually consists of communicating all actions by the focus character and other actions that affect them that happened in the selected subset, but selecting a trope will change this. When a trope is selected only the actions necessary for that trope will be communicated. However, since each trope is only applicable if the actions in the selected subset follow the pattern of the trope, this process may fail. In that case our system will fall back to generating the default narrative (e.g., a story that simply enumerates all actions in a given sequence).

Text generation The communicative primitives for text generation contain references to properties of characters and actions in the story. The text realizer translates these primitives into requests for sentences according to defined patterns. It then hands these over to a microplanner that performs basic text transformations like aggregation, pronominalization, adding connectives, and selection of the correct tense and conjugation. The web service uses SimpleNLG (Gatt and Reiter 2009) for final text realization.

Map generation Maps are generated using ArcGIS (ESRI 2016) with a custom map for the DOTA 2 game map,[4] and custom glyphs to represent the characters in the game and their actions. To fully utilize the capabilities of maps as a communicative medium, the communicative primitives for the maps also include an option to convey the positions of actors in addition to showing their actions. The rendering of the map places the characters and their actions at the appropriate locations, automatically determines the zoom level in

[4] DOTA 2 uses only a single map in which all games are played.

the map, and adds a legend and an overview map for orientation.

Film generation To generate cinematics, the discourse planner determines which actions should be filmed and what shot type to use for each of those actions. For example, certain team fights are best shown using a crane shot that shows an overview of what is happening, while character or group movement is typically filmed by dollying the camera along with the moving character(s). To achieve these shots, our system translates discourse plan steps into a camera shot specification language, Oshmirto, used by our cinematic rendering engine, FireBolt (Thorne et al. 2017). FireBolt also takes as input the set of story actions used by Bardic for the given domain and a cinematic model which maps logical actors and actions to renderable 3D models and associated animations. FireBolt then iterates over the requested shots and determines how to film them. For each shot, FireBolt must determine where to place the camera in accordance with the request. It chooses an ideal location for a given shot based on requested shot attributes such as subject, f-stop, and lens focal length. FireBolt then performs a raycast to determine whether view of the subject is obstructed. If an obstruction is detected, the engine searches locally around the ideal camera location for a new camera placement with the subject in view.

To create cinematics that more effectively convey the actions from the underlying storyworld domain, FireBolt makes use of virtual actors, objects and a virtual set that recreates its domain as closely as possible. In our DOTA 2 example, FireBolt uses a virtual set that recreates the DOTA 2 map within the Unity game engine. Character models from DOTA 2 and their animations are also used by FireBolt.

Generated Artifacts and Discussion

The main characterization of our system comes from observing the output and informally gauging its aesthetic qualities. In this section we will show several output examples and describe how they exhibit desirable qualities. Figure 4 shows a map output for a simple scene that focuses on one character. Here, the moves that are shown are on a relatively high granularity and the map describes how the character moved from the western jungle region of the map to the center. The overview map in the bottom right helps viewers to place the scene on the larger map, and the legend provides additional information for those not familiar with the game.

Table 1 shows the first few sentences of a generated text narrative of a scene involving multiple characters. As can be seen, the structure of the text follows a setup ("Wisp was a hero") and then lists the character's actions using the appropriate temporal or causal connectives ("At the same time, Wisp killed the Radiant bottom ancient tower," "Next Wisp killed an unknown creep because Wisp damaged the Radiant bottom ancient tower for 25 damage."). Use of these connectives is driven by the Impulse model's explicit representation of the corresponding temporal, causal and intentional relationships between actions. In some cases, using tropes can focus the narrative by restricting its content to focus on the actions relevant to the trope's structure. Table 2 shows a sample of a text narrative that uses the *Failure Montage* trope to convey a character attempting to achieve some goal and ultimately failing.

Wisp was a hero. Skywrath Mage was a hero. Dazzle was a hero. Skywrath Mage cast a frostbite spell on Pudge for 234 damage. Meanwhile, Dazzle healed himself for 25 damage. Then Skywrath Mage damaged a Radiant bottom ancient tower for 30 damage. Next Dazzle healed himself for 25 damage. Next Wisp killed an unknown creep because Wisp damaged the Radiant bottom ancient tower for 25 damage. At the same time, Wisp killed the Radiant bottom ancient tower. Then Dazzle healed himself for 25 damage. Next he moved from the Middle Outer Radiant Tower to the Middle Inner Radiant Tower.

Table 1: The start of a text narrative. In the DOTA 2 domain, the term "unknown creep" refers to a type of computer-controlled helper character. The term "Radiant" is the name of one of the two teams of players in the conflict.

Dazzle was a main-actor. He damaged Crystal Maiden for 22 damage with a poison touch. Then he moved from the Radiant Base to the Bottom Inner Radiant Tower. Next he damaged Crystal Maiden for 22 damage with the poison touch. Then he killed her because he damaged her for 22 damage with the poison touch. Next he healed Wisp for 25 damage. Then he died.

Table 2: Text narrative generated using the *Failure Montage* trope. This could be considered a success if the goal was killing Crystal Maiden, but from the goal of staying alive it is a failure.

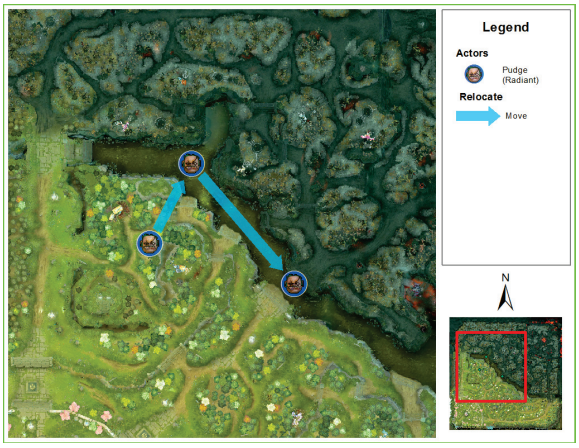

Figure 4: An example showing an automatically generated map. In this map, the movement of the character Pudge is shown along two distinct paths.

As for cinematic shot selection, Figure 5 shows the character Mirana on her tiger in a shot over her shoulder as she fires an arrow at an enemy character. This shot type was chosen by the system to show the attack action. In contrast, to show several characters moving at once, the system

Figure 5: An automatically generated over-the-shoulder shot of Mirana.

Figure 6: An automatically generated crane shot of three characters moving along a row of trees.

produced the crane shot shown in Figure 6. As mentioned above, this is not the only possible shot for moves. For instance when a move involves only one character, a shot in which the camera moves along with that character is used.

Finally, to illustrate our demo setup, Figure 7 shows the cinematic rendered on a 21' display. In the foreground, a table-top touch display used as the system controller is shown. This setup is particularly conducive to use by a team or larger group of users interested in reviewing the narrative output collectively.

Discussion

The Bardic system generates text, map and video narratives characterizing stories based on video game logs. However, these capabilities did not come without obstacles or limitations. First, while our data source has many desirable properties, the binary format has not been fully reverse engineered, so there are some actions that we can not extract. Second, the amount of log entries in a typical DOTA 2 log makes processing data from a full game log complex. A typical game has around 60000 actions that we can extract in our Impulse file, which translates to about 150MB of XML. While this is not large for static analysis, using the full file as input, e.g. to the discourse planner, would be computationally infeasible.

Our subset selection UI allows the user to select an interesting part of the game in order both to show only relevant information and to keep computation times reasonable. In its present stage, the selection UI can be used in real time, with narrative generation taking between $2-10$ minutes for subsets of the game that correspond to about 2 minutes of game time, depending on the complexity of the selected subset.

Figure 7: Bardic running on a 21' display using multi-screen table-top touch screen controllers. The image shows a generated cinematic for the DOTA 2 domain.

In terms of output, as the examples above show, there are still areas for improvement. In some corner-cases the text-output can contain sentences like "then a was". These are manifestations of actions that do not have sufficient information to be realized as a complete clause or sentence, such as actions with no actor. The text currently generated by Bardic is limited in terms of features like aggregation, effective reference to the underlying Impulse temporal model, and discourse-level structure that would facilitate abstraction or ellipsis in the description of action.

Map output like the one shown in Figure 4 provides a good overview of character actions. Unfortunately, situations like the one depicted in which a character moves from one part of the map to another are not the norm. In busy fight scenarios, the maps get visually cluttered very easily. Our current approach therefore uses the maps as an overview only.

The biggest challenge for the video generation is camera placement, because the planner is unaware of the scene geometry and may request shot types that cannot be fulfilled in a satisfactory manner. For example, if the requested shot is a medium shot and the character is surrounded by trees, there may not be any place to put the camera to fulfill this request. Our approach tries a number of possible positions, but in such cases the view may be (partially) obstructed by the trees. Another issue is in determining when an action is shown multiple times. If we show one character attacking another character, this may also show other actions in the background. Determining when this exposition constitutes actual communication to the audience and when the background action should be filmed again separately is not feasible in the general case. Our current approach is to assume that none of the background action is communicated and film every action that is to be shown.

Conclusion and Future Work

Bardic and its web services generate multimedia narratives from game logs created by the game *Defense of the Ancients 2*. The system works by translating the game data into a logic-based format called Impulse, which is then used by intelligent planning systems and other modules to generate narrative-focused visualizations, to support query capabilities, and to aid users in identifying a scene of interest in a given game. The main capability of our system is the generation of a narrative report in either text, map or video form about the actions in that scene. The sample output discussed above is illustrative of the kinds of reports users can produce. Overall, the system represents a significant integration of disparate intelligent tools to provide support for sensemaking from large amounts of low-level event data. We hope to see both direct applications (e.g., automatically generating commentary or highlight reels for DOTA 2 or other games, perhaps in an esports context) and wider-ranging contexts (e.g., interactive tools for sensemaking in network analysis or national security activities).

We are currently working to extend Bardic to support another activity domain based on e-mail data from the ENRON corpus (2004). This effort requires the development of automated processes to extract event data from the email corpus and encode that data into the Impulse format. Further, it requires the specification of templates for appropriate media realizations. We selected the ENRON corpus as a second domain primarily because it characterizes a significantly different type of activity from the DOTA 2 logs. Through this effort, we hope to increase the generality of Bardic by extending it to disparate domains.

In our future work, we hope to connect Bardic's methods more directly with the growing body of work on narrativization – the generation of narrative from low-level data as a way of conveying the data's structure. For example, work on generating narratives from sports data (Bouayad-Agha et al. 2012; Lareau, Dras, and Dale 2011; Allen et al. 2010) and from gameplay (Antoun et al. 2015) logs (Antoun et al. 2015; Gervás 2014b; 2014a; Mateas, Vanouse, and Domike 2000; Cheong et al. 2008a) focuses on challenges ranging from the representation of event types, to identification of interesting events/event sequences to the lexical realization of the narrative summary, all challenges for extensions to Bardic.

Another open research area is a modification of the multimedia narrative generation to extend the set of tropes we use as well as to better integrate the different media types with one another. Work by Lehnert (Lehnert 1981) previously defined a set of plot units used in narrative generation, and exploring the adaptation of some or all of those concepts into new tropes may prove promsing. Futher, we plan to modify the narrative generation process to integrate communicative content across media simultaneously, according to which action is being communicated.

Additionally, in our current implementation, it is possible for the narrative generator to request communicative actions which the media realizer is unable to fulfill properly (e.g., camera shots for which there is no occlusion-free camera position). We hope to address this by developing bidirectional communication between the discourse planner and the media realizer, allowing the planner to adapt a discourse plan accordingly (Ronfard and Szilas 2014).

Acknowledgements

The authors wish to thank Nathan Brown, Scott Carpenter, Mark DeMaria, Alyssa Knoll, Jeff Ligon, Max Litvinov, Trey Overman, Aaron Quidley, Nishant Rodrigues, John Slankas, David Winer, and Donnie Wrights for their work on Bardic.

Acknowledgements

This material is based upon work supported in whole or in part with funding from the Laboratory for Analytic Sciences (LAS). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the LAS and/or any agency or entity of the United States Government.

References

Allen, N.; Templon, J.; Summerhays McNally, R.; Birnbaum, L.; and Hammond, K. 2010. Statsmonkey: A data-driven sports narrative writer. In *AAAI Fall Symposium: Computational Models of Narrative*.

André, E., and Rist, T. 1995. Generating coherent presentations employing textual and visual material. *Artificial Intelligence Review* 9(2):147–165.

Andre, E.; Finkler, W.; Graf, W.; Rist, T.; Schauder, A.; and Wahlster, W. 1993. WIP: The automatic synthesis of multimodal presentations. In Maybury, M., ed., *Intelligent Multimedia Interfaces*, 75–93. AAAI Press.

Antoun, C.; Antoun, M.; Ryan, J.; Samuel, B.; Swanson, R.; and Walker, M. 2015. Generating natural language retellings from prom week play traces. In *Workshop on Procedural Content Generation in Games*.

Barot, C.; Potts, C. M.; and Young, R. M. 2015. A tripartite plan-based model of narrative for narrative discourse generation. In *Proceedings of the Joint Workshop on Intelligent Narrative Technologies and Social Believability in Games at the 11th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2–8.

Bouayad-Agha, N.; Casamayor, G.; Mille, S.; and Wanner, L. 2012. Perspective-oriented generation of football match summaries: Old tasks, new challenges. *ACM Transactions on Speech and Language Processing (TSLP)* 9(2):163–182.

Bruner, J. 1991. The narrative construction of reality. *Critical Inquiry* 1(18):1–21.

Cheong, Y.; Jhala, A.; Bae, B.; and Young, R. 2008a. Automatically generating summaries from game logs. In *Proceedings of Conference on AI and Interactive Digital Entertainment*, 167–172.

Cheong, Y.-G.; Jhala, A.; Bae, B.-C.; and Young, R. M. 2008b. Automatically generating summary visualizations from game logs. In *Proceedings of the Conference on AI and Interactive Digital Entertainment*, 167–172.

Eger, M.; Barot, C.; and Young, R. M. 2015. Impulse: A formal characterization of story. In *Proceedings of the 6th Workshop on Computational Models of Narrative*, 45–53.

ESRI. 2016. ArcGIS. Software available at http://www.arcgis.com/.

Feiner, S. K., and McKeown, K. R. 1991. Automating the generation of coordinated multimedia explanations. *Computer* 24(10):33–41.

Fikes, R. E., and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 5:189–208.

Gatt, A., and Reiter, E. 2009. SimpleNLG: A realisation engine for practical applications. In Maybury, M., ed., *Proceedings of the 12th European Workshop on Natural Language Generation*, 9093. AAAI Press.

Gervás, P. 2014a. Composing narrative discourse for stories of many characters: A case study over a chess games. *Literary and Linguistic Computing*.

Gervás, P. 2014b. Metrics for desired structural features for narrative renderings of game logs. *Entertainment Computing* 5(4):245–250.

Jhala, A., and Young, R. M. 2010. Cinematic visual discourse: Representation, generation, and evaluation. *IEEE Transactions on Computational Intelligence and Artificial Intelligence in Games* 2:69–81.

Klimt, B., and Yang, Y. 2004. The enron corpus: A new dataset for email classification research. In *Proceedings of the 15th European Conference on Machine Learning*, 217226.

Lareau, F.; Dras, M.; and Dale, R. 2011. Detecting interesting event sequences for sports reporting. In *Proceedings of the 13th European Workshop on Natural Language Generation*, 200–205.

Lehnert, W. G. 1981. Plot units and narrative summarization. *Cognitive Science* 5(4):293–331.

Mateas, M.; Vanouse, P.; and Domike, S. 2000. Generation of ideologically-biased historical documentaries. In *National Conference of the American Association for Artificial Intelligence*, 236–242.

Molina, M., and Flores, V. 2012. Generating multimedia presentations that summarize the behavior of dynamic systems using a model-based approach. *Expert Systems with Applications* 39(3):2759–2770.

Ronfard, R., and Szilas, N. 2014. Where story and media meet: computer generation of narrative discourse. In *Proceedings of the 5th Workshop on Computational Models of Narrative*, 164176.

Thomas, J., and Young, R. M. 2010. Annie: Automated generation of adaptive learner guidance for fun serious games. *IEEE Transactions on Learning Technologies* 3(4):329–343.

Thorne, B., and Young, R. M. 2017. Generating stories that include failed actions by modeling false character beliefs. In *Working Notes of the AIIDE Workshop on Intelligent Narrative Technologies*.

Thorne, B.; Winer, D. R.; Barot, C.; and Young, R. M. 2017. Firebolt: A system for automated low-level cinematic narrative realization. Technical report, Entertainment Arts and Engineering Program, University of Utah, Salt Lake City, UT. EAE Technical Report 2017-004.

Valve Corporation. 2013. Defense of the ancients 2. Video Game, PC.

Wahlster, W.; Andr, E.; Finkler, W.; Profitlich, H.-J.; and Rist, T. 1993. Plan-based integration of natural language and graphics generation. *Artificial Intelligence* 63(1–2):387–427.

Ware, S. G., and Young, R. M. 2011. CPOCL: A narrative planner supporting conflict. In *In Proceedings of the 7th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 97102.

Young, R. M. 2017. Sketching a generative model of intention management for characters in stories: Adding intention management to a belief-driven story planning algorithms. In *Working Notes of the AIIDE Workshop on Intelligent Narrative Technologies*.