Chapter 7
Planning with applications to quests and story

Yun-Gyung Cheong, Mark O. Riedl, Byung-Chull Bae, and Mark J. Nelson

Abstract Most games include some form of narrative. Like other aspects of game content, stories can be generated. In this chapter, we discuss methods for generating stories, mostly using planning algorithms. Algorithms that search in plan space and those that search in state space can both be useful here. We also present a method for generating stories and corresponding game worlds together.

7.1 Stories in games

Games often have storylines. In some games, they are short backstories, serving to set up the action. The first-person shooter game Doom’s storyline, about a military science experiment that accidentally opens a portal to hell, is perhaps the canonical example of this kind of story: its main purpose is to set the mood and general theme of the game, and motivate why the player is navigating levels and shooting demons. The level progression and game mechanics have very little to do with the storyline after the game starts. In other games, the storyline structures the progression of the game more pervasively, providing a narrative arc within which the gameplay takes place. The Final Fantasy games are a prominent representative of this style of game storyline.

Since the theme of this book is to procedurally generate anything that goes into a game, it will not surprise the reader that we will now look at procedurally generating game storylines. As with procedural generation of game rules, discussed in the previous chapter, procedural generation of storylines is somewhat different from generation of other kinds of procedural content, because storylines are an unusual kind of content. They often intertwine pervasively with gameplay, and their role in a game can depend heavily on a game’s genre and mechanics.

A common way of integrating a game’s storyline with its gameplay, especially in adventure games and role-playing games, is the quest [23, 1]. In a quest, a player is given something to do in the game world, which usually is both motivated by the
current state of the storyline, and upon completion will advance it in some way. For example, the player may be tasked with retrieving an item, helping an NPC, defeating a monster, or transporting some goods to another town. Some games (especially RPGs) may be structured as one large quest, broken down into smaller sub-quests that interleave gameplay and story progression.

There are several reasons a game designer might want to procedurally generate game stories, beyond the general arguments for procedural content generation discussed in Chapter 1. One reason is that procedurally generated game worlds can lack meaning or motivation to the player, unless they are tied into the game story by procedurally generating relevant parts of the story along with the worlds. As Ashmore and Nitsche [2] argue, “without context and goals, the generated behaviours, graphics, and game spaces run the danger of becoming insubstantial and tedious.” A second reason is that proceduralizing quests can make them truly *playable*. Sullivan et al. [21] note that computer RPGs often have a particularly degenerate form of quest, “generally structured as a list of tasks or milestones,” rather than open-ended goals the player can creatively satisfy. Table-top RPGs have more complex and open-ended quests, since in those games, quests can be dynamically generated and adapted during gameplay by the human game-master, rather than being prewritten. Procedural quest generation gives a way to bring that flexibility back into videogame quests.

### 7.2 Procedural story generation via planning

One way to think about procedurally generating stories is to consider them to be a *planning* problem. In artificial intelligence, planning algorithms search for sequences of actions that satisfy a goal. A robot, for example, plans out the series of actuator movements necessary to pick up an object and carry it somewhere.

What are the sequences of actions for a story, and what is the goal? There are a number of ways to answer those questions, and researchers on procedural story generation started looking at them in the 1970s—at the time, generating purely text-based short stories, not game stories.

We could answer that a story is a sequence of events in a story world (in our case, a game world)—a sequence that eventually leads, through the chain of events, to the story’s ending. Therefore we generate stories by simulating a fictional work: to tell a story, we first simulate what happens as characters move around and take actions in the story world, and then the story consists of simply recounting the events that happened. One of the first influential story-generation systems, *Tale-Spin* [14], takes this approach.

Generating stories by simulating a story world does have some shortcomings. It does not take into account what makes a *story*—particularly an interesting story—different from simply a log of events. Stories are carefully crafted by authors to have a certain pace, dramatic tension, foreshadowing, a narrative arc, etc., whereas a simulation of a day in the life of a virtual character does not necessarily have any