# Natural Language Game Interaction From a Planning Perspective

The Laboratory for Quantitative Experience Design

Jennifer Nelson, School of Computing
Supervised by Prof. Rogelio Cardona-Rivera, School of Computing

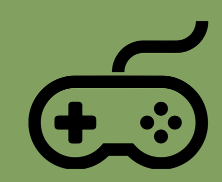
This work was supported by funding from the Undergraduate Research Opportunities Program at the University of Utah awarded to Jennifer Nelson.

Office of UNDERGRADUATE RESEARCH THE UNIVERSITY OF UTAH

## Interactive Narrative

Interactive narrative is cooperative storytelling, in that it allows both the author and the audience to shape a story as it is told. This research looks at interactive narrative between humans and computers.

In research, human-computer interactive narrative usually involves narrative planning.

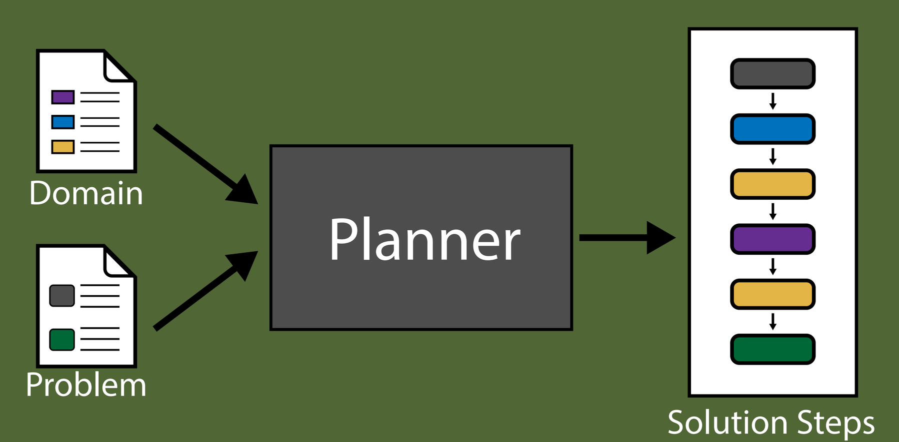


Video games are the most common form of interactive narratives between humans and computers.

Tabletop role-playing games are interactive storytelling between humans. This research aims to enable human-computer role-playing games.

Students learn better when content is presented in narrative form, and when the student interacts with the content. Education is a prime area for human-computer interactive narratives to make a big difference.

### Planning:




Planner: Takes a domain and problem, and produces a sequence of actions solving the problem.
Domain: predicates and actions
Actions: transitions between states
Problem: An initial state and goal state

**Narrative Planning**

Narrative planners use story-oriented domains and problems, and often alter planning algorithms to produce more interesting stories

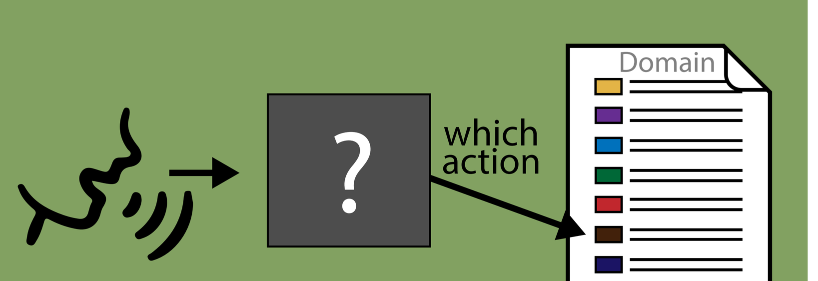
**Interactive Narrative Planning:**

Interactive narrative planners alternate story control between players and system, with the system re-planning a good story after every trade-off.

### "Interactive" ⟹ Interaction

**Interaction Requires Natural Language**

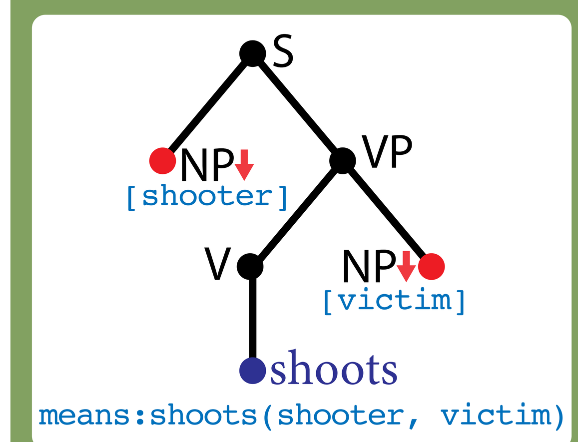
For humans to create collaborative stories with computers, the computer must speak our language. The golden goal here is for humans to speak naturally with the computer, as though speaking with another human.
The shorter-term goal is for the computer to correctly interpret a human's desired in-story action.

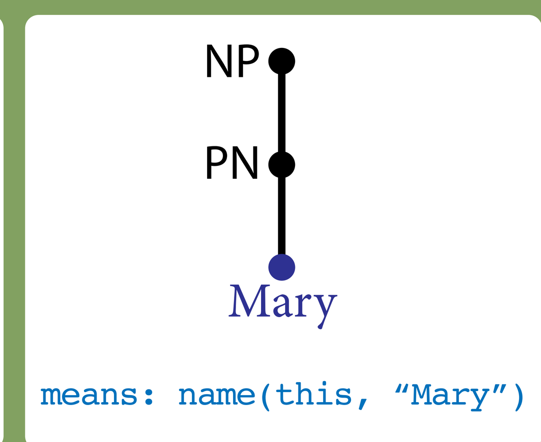


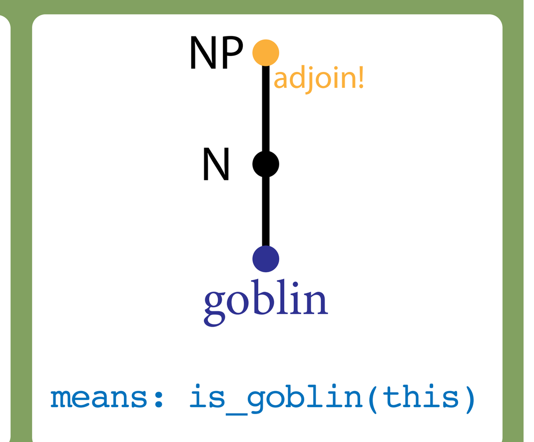

**Mystery Box Requirement**

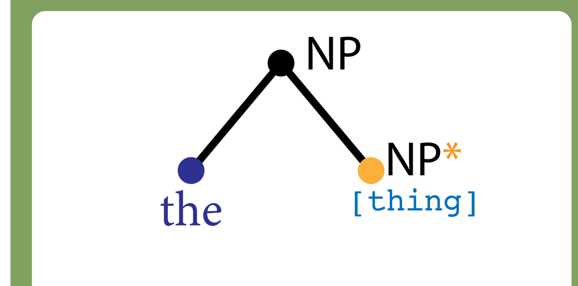
Given a simple sentence, determine which in-world action is meant, and its corresponding parameters.

## Speech Act Planning



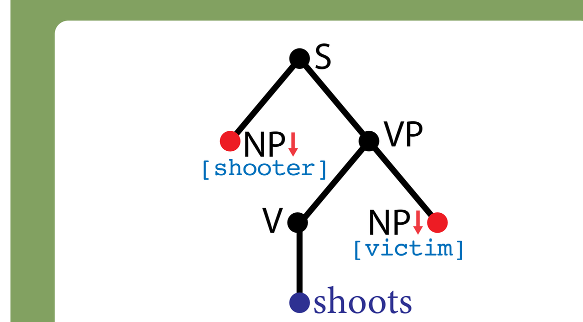
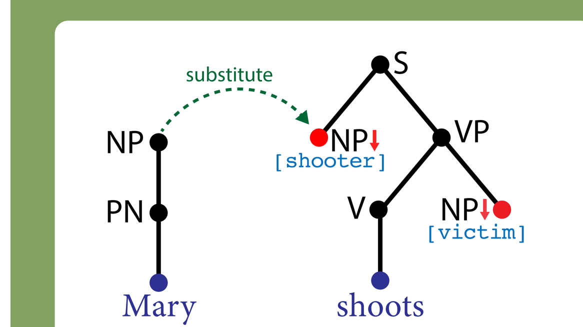
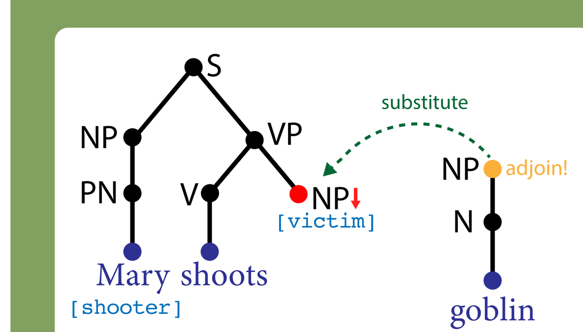
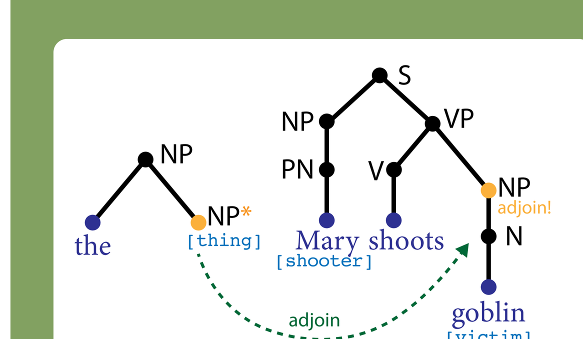
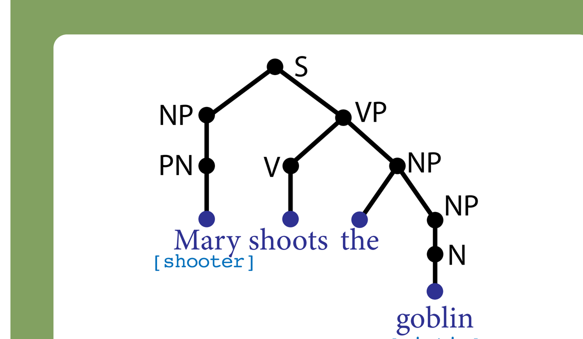




```
is_goblin(g1)
is_person(c1)
is_person(c2)
name(c1, "Mary")
name(c2, "John")
```

**GOAL:**
`express shoots(ch1, g1)`
(Tell listener that Mary shot the goblin.)

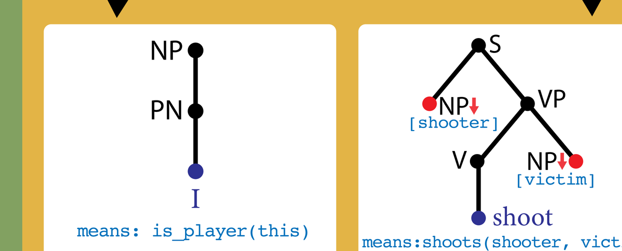
| Sentence Tree: | Action | State |
|---|---|---|
|  | `S-shoots(root):` `effect:` `!subst(S, root)` `subst(NP, shooter)` `subst(NP, victim)` `shoots(shooter,victim))` ? shoots ? | The listener now knows someone shoots something. Two NP nodes need substitution to specify who someone and something are. |
|  | `NP-Mary(shooter):` `effect:` `!subst(NP, shooter)` `name(shooter, "Mary")` Mary shoots ? | The listener now knows Mary shoots something. The other NP node still needs substitution to specify what something is. |
|  | `NP-goblin(victim):` `effect:` `!subst(NP, victim)` `adjoin_me(NP, victim)` `is_goblin(victim)` Mary shoots ? goblin | The listener now knows Mary shoots some goblin. The NP nodes are substituted, but we've added a new node that needs adjoining. The adjoin specifies *which* goblin, and makes the sentence grammatically correct. |
|  | `NP-the(victim):` `preconditions:` `hearer-old(victim)` `effect:` `!adjoin_me(NP,victim)` Mary shoots the goblin. | All gramatical requirements are satisfied. The listener knows Mary shoots the goblin. The goal is accomplished. |
|  | `shoots(shooter, victim)` `name(shooter, "Mary")` `is_goblin(victim)` Mary shoots the goblin. | We've built a full sentence with the desired message. ---------------------------------- Simplified syntax. Drawn from: Koller, Alexander, and Matthew Stone. "Sentence generation as a planning problem." (2007). |

## Plan Recognition

We use plan recognition algorithms to fill in the gaps of speech, and understand references.

We also utilize extra knowledge of the speaker's context, probable goals, and history.


**Observations:**




Game history:
- goblin1 attacked player1
- Mary shot goblin1
- player1 has cooperated with Mary
- player1 has never cooperated with goblins

**Plan Recognition Algorithms**

**Results:**

```
1 S-shoot(root)
2 NP-I(shooter)
3 NP-goblin(victim)
4 NP-the(victim)
```

"I shoot the goblin."

| | |
|---|---|
| **80%** | **shoot(p1,goblin1)** |
| 10% | attack(p1,goblin1) |
| ⋮ | ⋮ |
| .05% | shoot(p1,Mary) |
| .04% | eat(p1,sandwich2) |

**shoot(p1,goblin1)**

1. Match words with corresponding speech action(s)
2. Speech actions and in-game actions become observations for plan recognition algorithms.
3. Plan recognition algorithm reports the likeliest interpretations.

## Why Recognition

This approach is motivated by the use of planning for both language generation and language understanding. By using plan recognition, we hope to be able to incorporate contextual info from the narrative planner itself.

**Cons:**
- Planning may be too slow for real-time interaction
- Requires handwritten language domain
  - Language generation already required this. Reuse.
  - Doesn't require training data. Just assign meaning to words.

**Pros:**
- Treats speaker as an intelligent agent.
- Incorporates context from the narrative planner itself:
  - Game history
  - Models of player intention and knowledge
- Uses the same dictionary as language generation.
- Plan-based language generation is good at reference resolution, and we expect plan recognition to follow similarly.
- Utilizes off-the-shelf algorithms, and will improve as they do.