Linking Natural Language to Action

Hadas Kress-Gazit

Jan M. Allbeck

Advisors: George Pappas and Norm Badler
SUBTLE MURI

Figure 1: Team Structure
LTL and PAR Integration

• Pragmatics-$\rightarrow$PAR-$\rightarrow$LTL-$\rightarrow$PAR
• Commands in the form of PARs will be instantiated from pragmatics.
• If the command requires planning (e.g. search), then LTL is called.
• LTL automatically and verifiably composes controllers that satisfy high level task specifications.
• PAR can then be used to fill in parameters of the actions and for simulation.
• Additionally, PAR provides LTL with precepts of the environment that produce state transitions in the LTL automaton.
Example Mission

- Murray starts in room 11.
- “Search rooms 1, 2, 3 and 4. If you see a dead body, abandon the search and go to room 11. If you see a bomb, pick it up and take it to room 13 and then resume the search.”
“Search rooms 1, 2, 3 and 4. If you see a dead body, abandon the search and go to room 11. If you see a bomb, pick it up and take it to room 13 and then resume the search.”
Known workspace
Dynamic environment

Sensor inputs
Actions
high level task

Automatic Correct by construction
Correct robot motion and action
Known workspace
Dynamic environment

Sensor inputs

Actions

Robot

High level task

LTL formula $\varphi$

Automaton

Hybrid Controller

Correct robot motion and action

Binary Propositions

Discrete Abstraction
Linear Temporal Logic (LTL)

Syntax:
\[ \varphi ::= \pi \mid \neg \varphi \mid \varphi \lor \varphi \mid \bigcirc \varphi \mid \square \varphi \mid \Diamond \varphi \mid \varphi U \varphi \]

Semantics: Truth is evaluated along infinite computation paths \( \sigma ((a,b),a,a,a... (a,b),(a,b),(a,c),(a,c),...) \)

- \( \sigma, i \models \pi \) iff \( \pi \in \sigma(i) \)
- \( \sigma, i \models \neg \varphi \) iff \( \sigma, i \not\models \varphi \)
- \( \sigma, i \models \varphi_1 \lor \varphi_2 \) iff \( \sigma, i \models \varphi_1 \) or \( \sigma, i \models \varphi_2 \)
- \( \sigma, i \models \bigcirc \varphi \) iff \( \sigma, i + 1 \models \varphi \)
- \( \sigma, i \models \square \varphi \) iff for all \( k \geq i \) \( \sigma, k \models \varphi \)
- \( \sigma, i \models \Diamond \varphi \) iff there exists \( k \geq i \) such that \( \sigma, k \models \varphi \)
- \( \sigma, i \models \varphi_1 U \varphi_2 \) iff there exists \( k \geq i \) such that \( \sigma, k \models \varphi_2 \), and for all \( i \leq j < k \) we have \( \sigma, j \models \varphi_1 \)
“Search rooms 1, 2, 3 and 4. If you see a dead body, abandon the search and go to room 11. If you see a bomb, pick it up and take it to room 13 and then resume the search.”

\[
\begin{align*}
\land (r_1 \rightarrow (r_1 \ r_5)) \\
\land (r_2 \rightarrow (r_2 \ r_6)) \\
\land (sawDead \rightarrow r_{11}) \\
\land ((haveBomb \land \neg sawDead) \rightarrow r_{13})
\end{align*}
\]
Automaton synthesis

• LTL formula converted to an automaton such that every execution is guaranteed to satisfy the formula (achieve the task) – if feasible
Sensor inputs → Actions → Dynamic environment

Known workspace

Robot

Sensor inputs

Actions

Discrete Abstraction

Binary Propositions

LTL formula $\varphi$

Automaton

Hybrid Controller

Correct robot motion and action

high level task
Hybrid Controller

Bisimilar low-level controllers: PAR or Feedback Control
Guarantee

• If the task is feasible, a controller will be created and the robot’s behavior will be correct, if the environment behaves well.
Simulation
Challenge

“If you see a bomb, pick it up and take it to room 13 and then resume the search”
Parameterized Action Representation

Ontology for simple and complex physical behaviors.

- Natural language and animation intermediary
- Applications: VET, ATOV
- Action and Object representations
- Stored in Hierarchies
- Uninstantiated and instantiated
Information in Effective Instructions

- Core action semantics (e.g. “remove”)
- Action/sub-action structure
- Participants (agent, objects)
- Path, manner, purpose information (“context”)
- Initiation conditions (applicability | preconditions)
- Termination conditions (success or failure cases)
PAR Actions

- **core semantics**: motion, force, state-change, paths
- **participants**: agent, objects
- **purpose**: state to achieve, action to generate, etc.
- **manner**: how to perform action (e.g. “carefully”)
- **type**: aleatoric, reactive, opportunistic
- **duration**: timing, iteration, or extent; e.g., “for 6 seconds”, “between 5 and 6 times”
- **sub-steps**: actions to perform to accomplish action (includes parallel constructs)
- **next-step**: next action to be performed
- **super-step**: parent action
- **conditions**: prior, post
Object Representation

\[
\text{type object representation = (name: STRING; is agent: BOOLEAN; properties: sequence property-specification; status: status-specification; posture: posture-specification; location: object representation; contents: sequence object representation; capabilities: sequence parameterized action; relative directions: sequence relative-direction-specification; special directions: sequence special-direction-specification; sites: sequence site-type-specification; bounding volume: bounding-volume-specification; coordinate system site; position: vector; velocity: vector; acceleration: vector; orientation: vector; data: ANY-TYPE).}
\]
NL: Murray, pickup bomb quickly

PAR: Agent: Murray  Action: PickUp
Object: Bomb  Manner: quickly

Animation:
Murray Interactive Demo
Action: Open the door
- TC: Is the door open?
- PS: Is the agent grasping the doorknob?
  - Exec: Turn the doorknob. Swing open the door.

Action: Grasp the doorknob
- TC: Grasping the doorknob?
- PS: Reach the doorknob?
  - Exec: Reach for the doorknob. Grasp the doorknob.

Action: Walk to the doorknob
- TC: At the doorknob?
- PS: Is the agent standing?
  - Exec: Walk to the doorknob

Action: Stand up
- TC: Is the agent standing?
- PS: TRUE
  - Exec: Stand up

END
PAR Summary

- Data driven
- Includes a world model
- Provides context
- Captures semantics
- Links to other software systems
- Levels of detail
- Reusable
- Composeable