Learning High Level Planning From Text

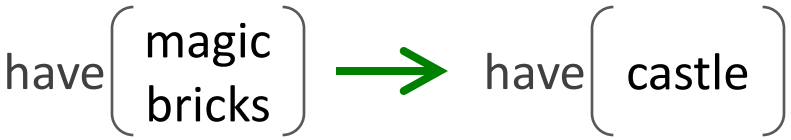
Nate Kushman

S.R.K. Branavan, Tao Lei, Regina Barzilay

Precondition/Effects Relationships

Castles are built with magic bricks

Classical Planning:



NLP: *Linguistic Relation*

Goal: Show that planning can be improved by utilizing precondition information in text

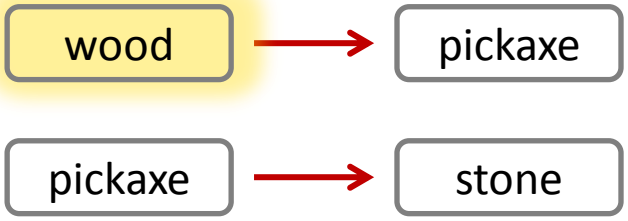
How Text Can Help Planning

Minecraft : Virtual world allowing tool creation and complex construction.

Text

A **pickaxe**, which is used to harvest **stone**, can be made from **wood**.

Preconditions



Plan

- Move to location: <3,3>
- Harvest: wood
- Retrieve: harvested wood

wood

- Setup crafting table
- Place on crafting table: wood
- Craft: pickaxe
- Retrieve: pickaxe

pickaxe

- Move to location: <1,2>
- Pickup tool: pickaxe
- Harvest: stone with: pickaxe
- Retrieve: stone

stone

Challenge: Preconditions from text cannot map directly to planning action preconditions

Opportunity

Classical Planning's Problem:

Exponential heuristic search

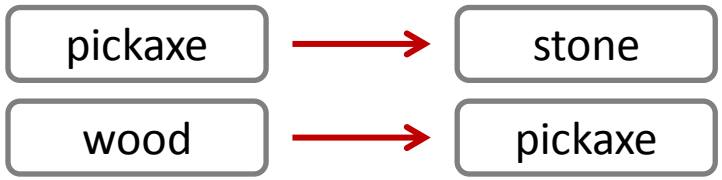
Traditional Solution:

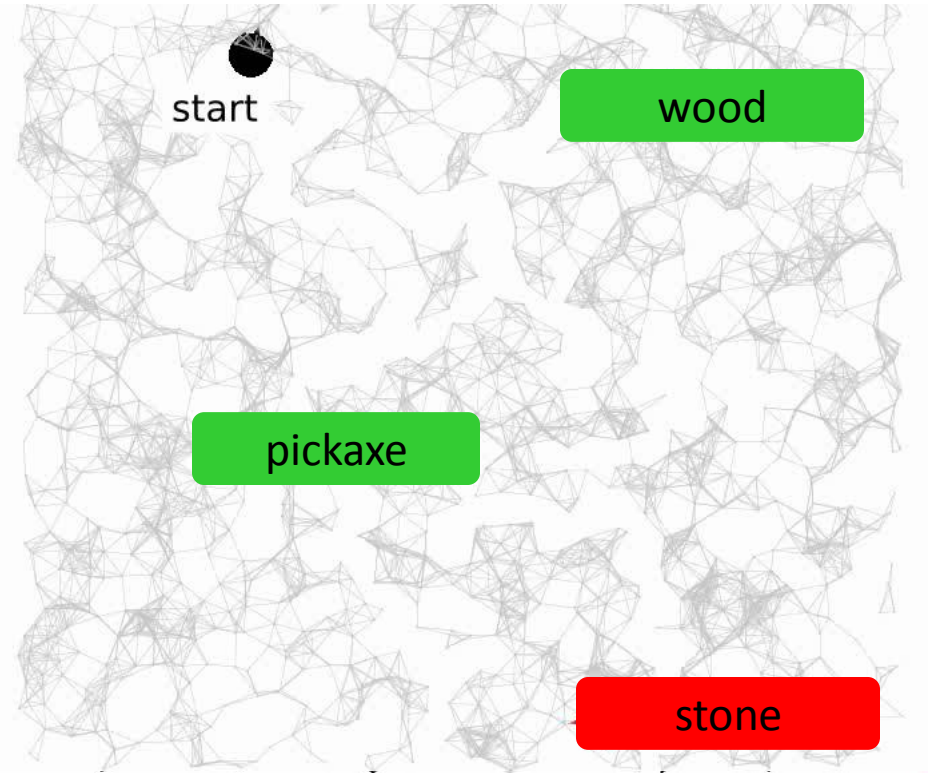
Analyze domain to induce subgoals

Text:

A **pickaxe**, which is used to harvest **stone**, can be made from **wood**.

Precondition Relations:





Key Idea: Map text precondition information to subgoals

Key Departures

Utilize domain specific information in text to induce subgoals

Jonsson and Barto, 2005; Wolfe and Barto, 2005; Mehta et al.,2008; Barry et al., 2011

looked only at domain, did not utilize text

Learn from only environment feedback

Girju and Moldovan, 2002; Chang and Choi, 2006; Blanco et al., 2008; Beamer and Girju, 2009; Do et al., 2011; Kwiakowski, 2012

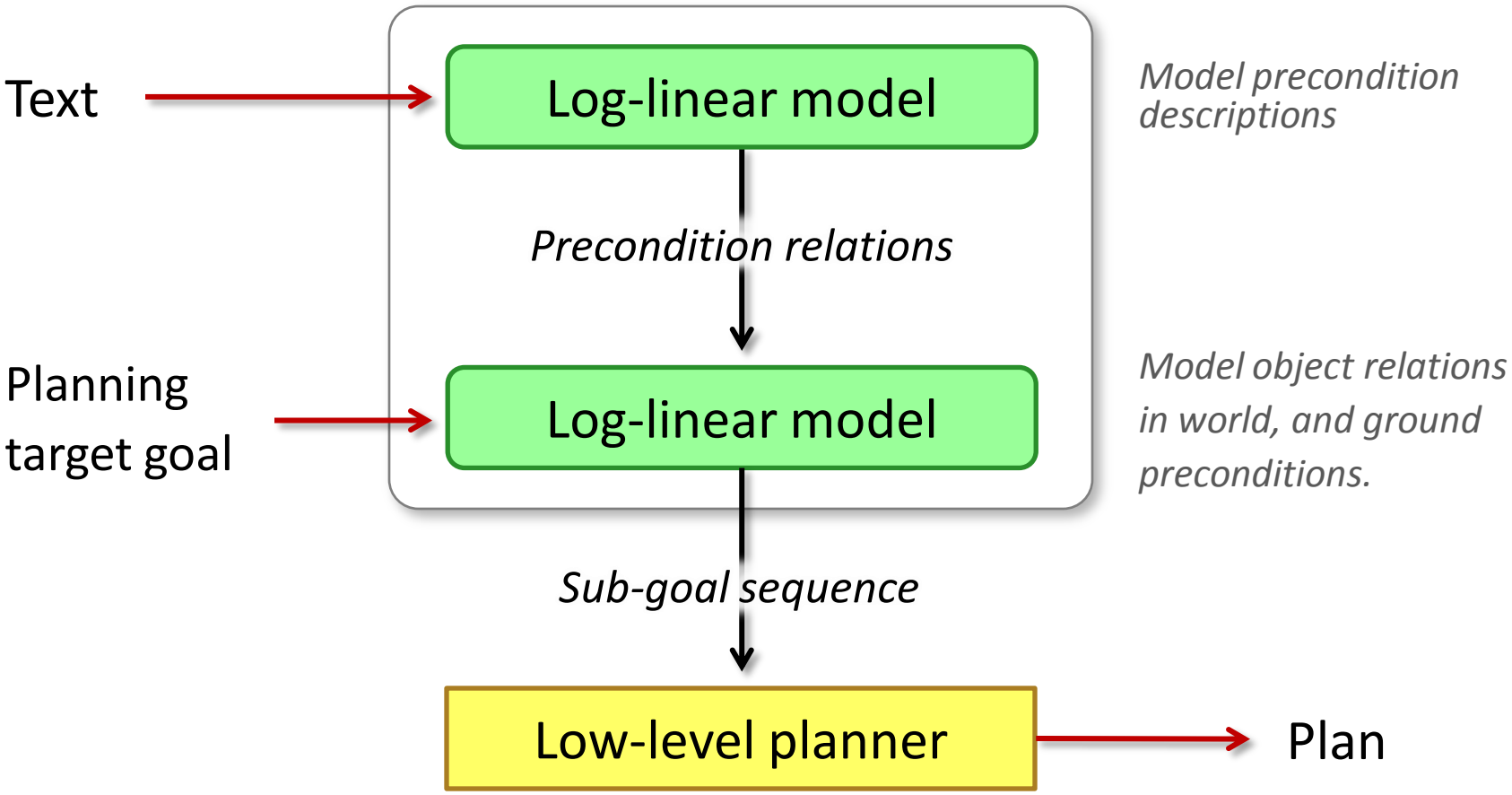
Learns from supervised data, does not utilized environment feedback

Utilize text providing abstract domain relationships (not goal specific)

Oates, 2001; Siskind, 2001; Yu and Ballard, 2004; Fleischman and Roy, 2005; Mooney, 2008; Branavan et al., 2009; Liang et al. , 2009; Vogel and Jurafsky, 2010; Branavan et al., 2009; Branavan et al. 2010; Vogel and Jurafsky, 2010; Branavan et al., 2011

Focused on grounding words to objects, does not ground relations

Hybrid Model



Learn model parameters from planning feedback

Modeling the World

- State is represented by a set of predicates

`current_location(1,2) = TRUE` `current_tool(pickaxe) = TRUE`

- Actions represented by preconditions and effects

Action: `chop_tree(1,2)`

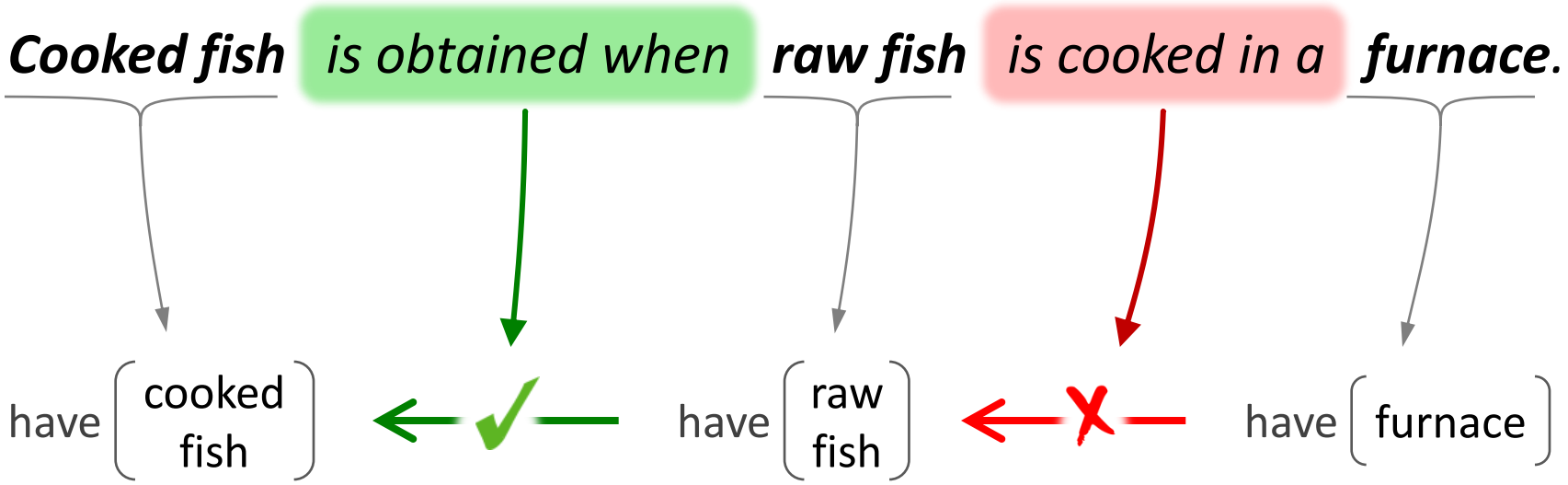
Preconditions: `tree_at(1,2) = TRUE` `current_location(1,2) = TRUE`

Effect: `tree_at(1,2) → FALSE` `have(wood) → TRUE`

Goals and subgoals are represented as predicates

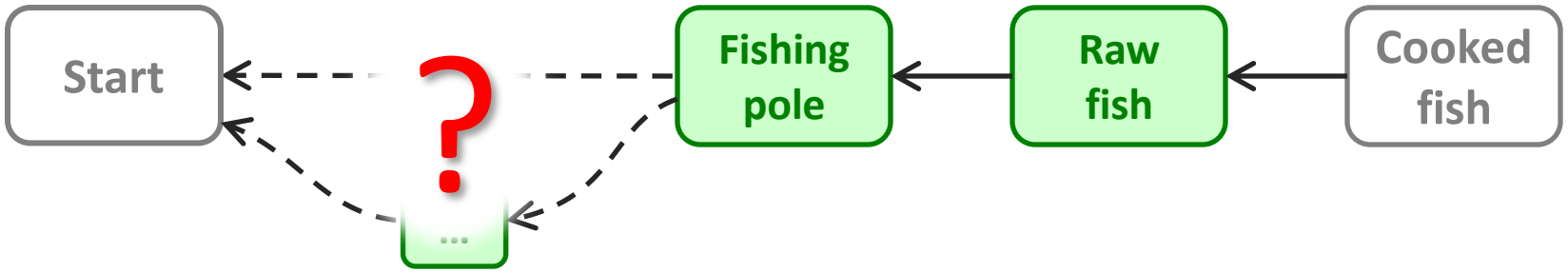
Model Part 1: Predict Precondition Relations

Goal Independent



Model Part 2: Predict Subgoal Sequence

Given Goal State



- Model as a Markov process
- Explicitly model preconditions observed via planner

Policy Functions

Model Part 1: Predict Precondition Relations from text

$$p(x_i \rightarrow x_j \mid \vec{w}_k, q_k; \theta_c) \propto e^{\theta_c \cdot \phi_c(x_i, x_j, \vec{w}_k, q_k)}$$

Prediction per pair

Manual groundings, x

Sentence, w, dependency parse, q

Model Part 2: Predict Subgoal Sequence

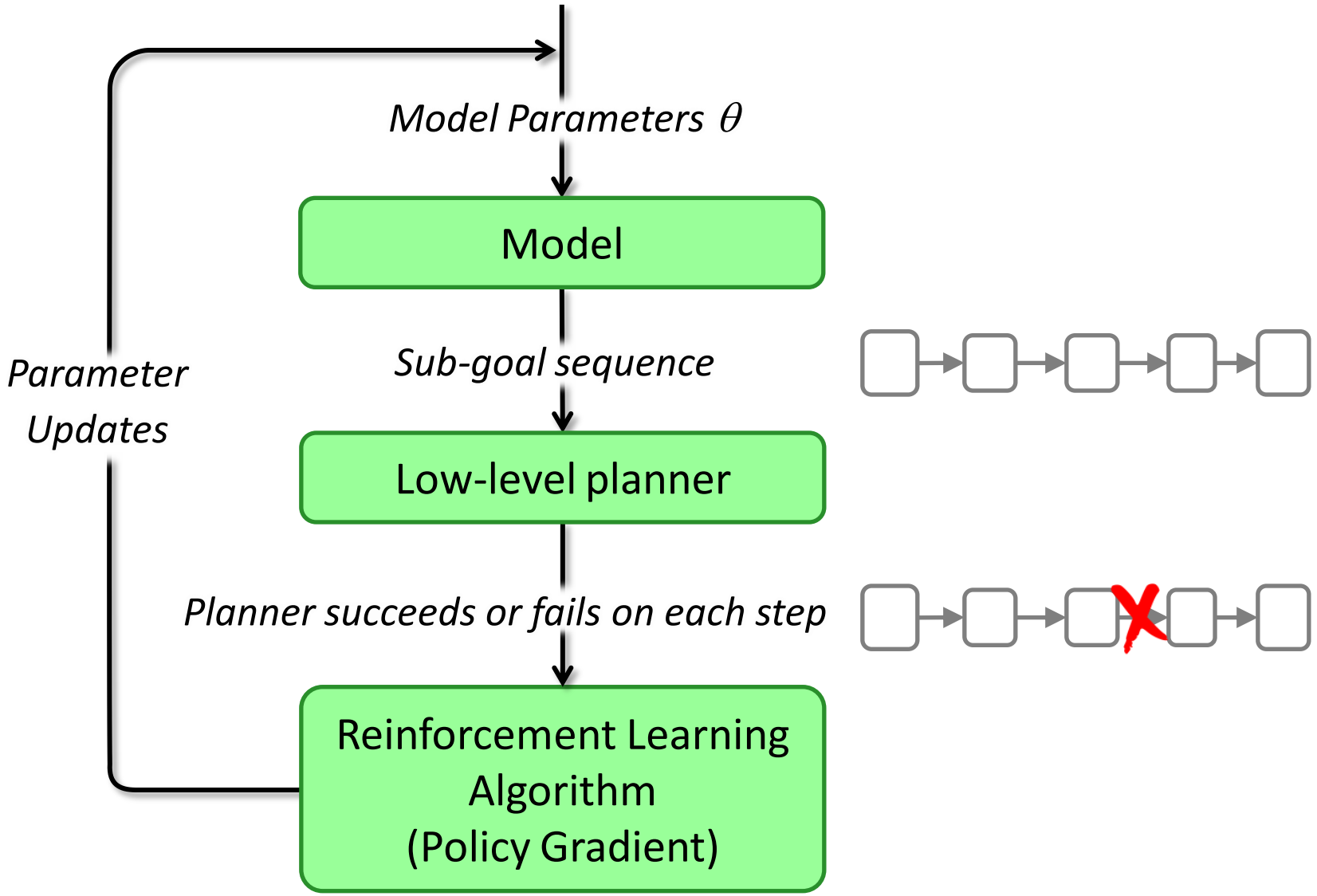
$$p(x_t \mid x_{t-1}, s_0^g, s_f^g, C; \theta_x) \propto e^{\theta_x \cdot \phi_x(C, x_t, x_{t-1}, s_0^g, s_f^g)}$$

Markov Assumption

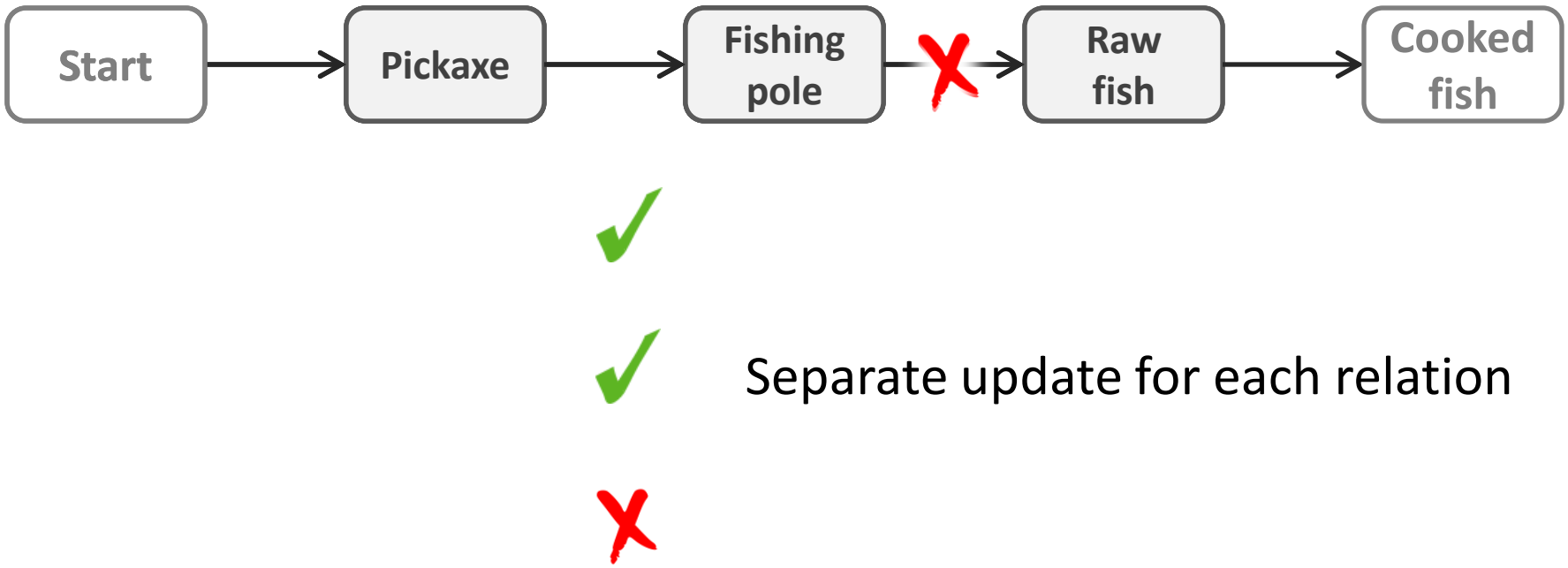
Relations from text, C

Relations between predicates, x

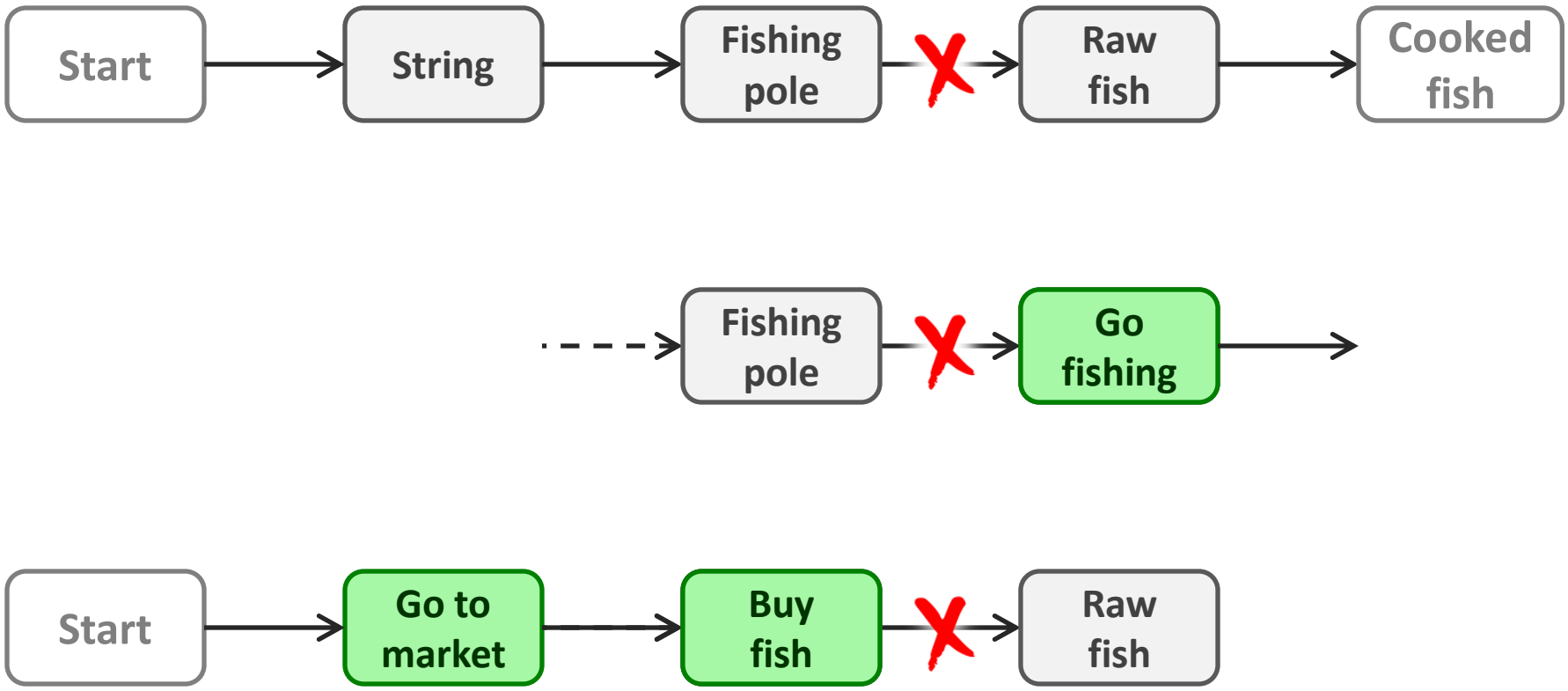
Learn Parameters Using Feedback from the Planner



Parameter Updates: Relation Prediction



Parameter Updates: Subgoal Sequence Prediction



One update for the whole sequence

Updates

Model Part 1: Precondition Relation Prediction

$$\Delta\theta_c \leftarrow \alpha_c r \left[\phi_c(\cdot) - \mathbb{E}\left[\phi_c(\cdot)\right] \right]$$

Success/failure of one subgoal pair

standard log-linear gradient

Model Part 2: Subgoal Sequence Prediction

$$\Delta\theta_x \leftarrow \alpha_x r \sum_t \left[\phi_x(\cdot) - \mathbb{E}\left[\phi_x(\cdot)\right] \right]$$

standard log-linear gradient

Success or failure of entire sequence

Sum over all subgoal pairs

Experimental Domain

World:

Minecraft virtual world

Documents:

User authored wiki articles

Text Statistics:

Sentences:	*242*
Vocabulary:	*979*

Planning task Statistics:

Tasks:	*98*
Avg. plan length:	*35*
Min. Branching Factor:	*8*

Models compared

Unmodified Low-level Planner

Fast-Forward – standard baseline in classical planning

No induced subgoals

No Text

Second half of model given no relations from text

All Text

Generate all connections with grounded phrase in same sentence

Second-half of model with this set of connections

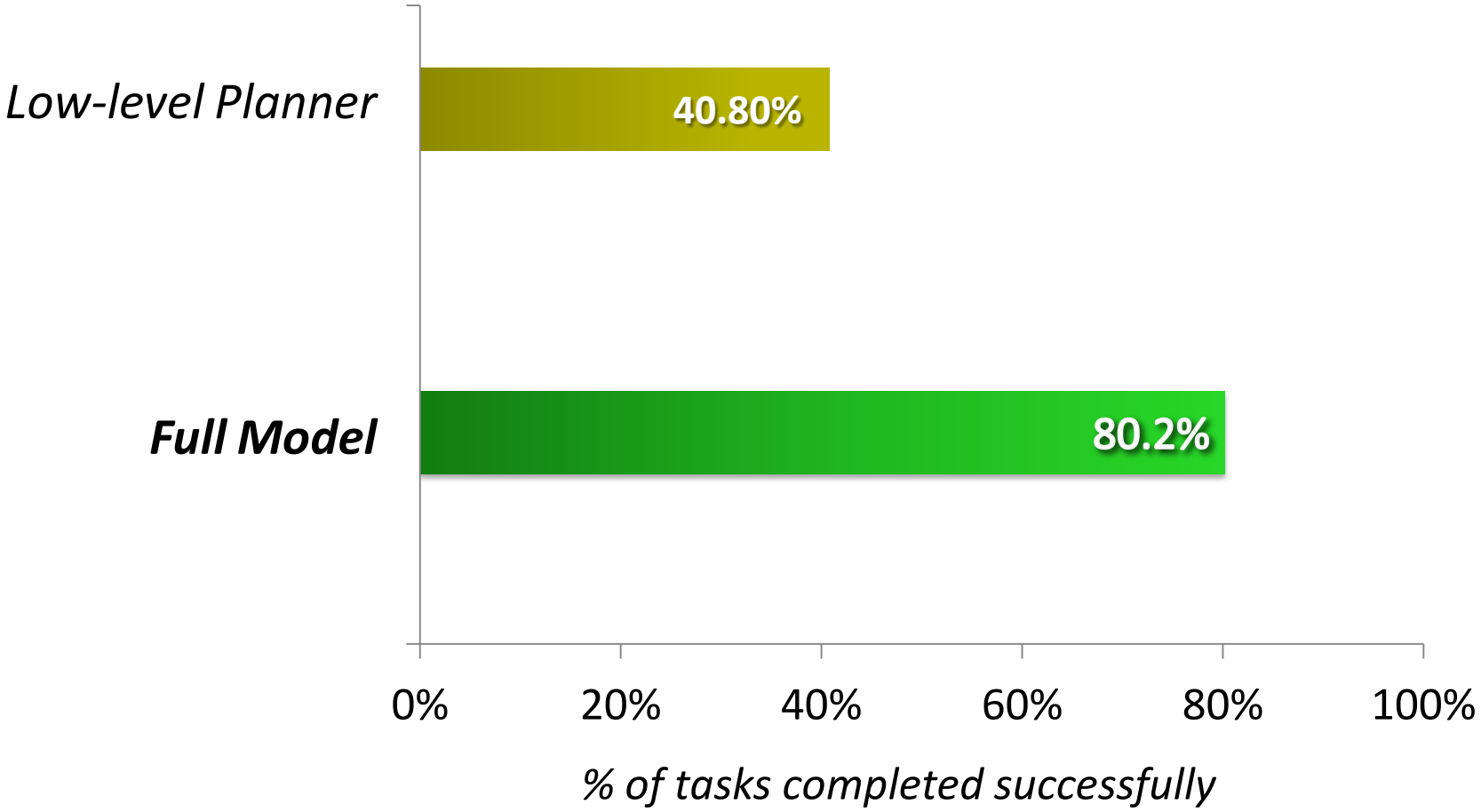
Full Model

As described so far

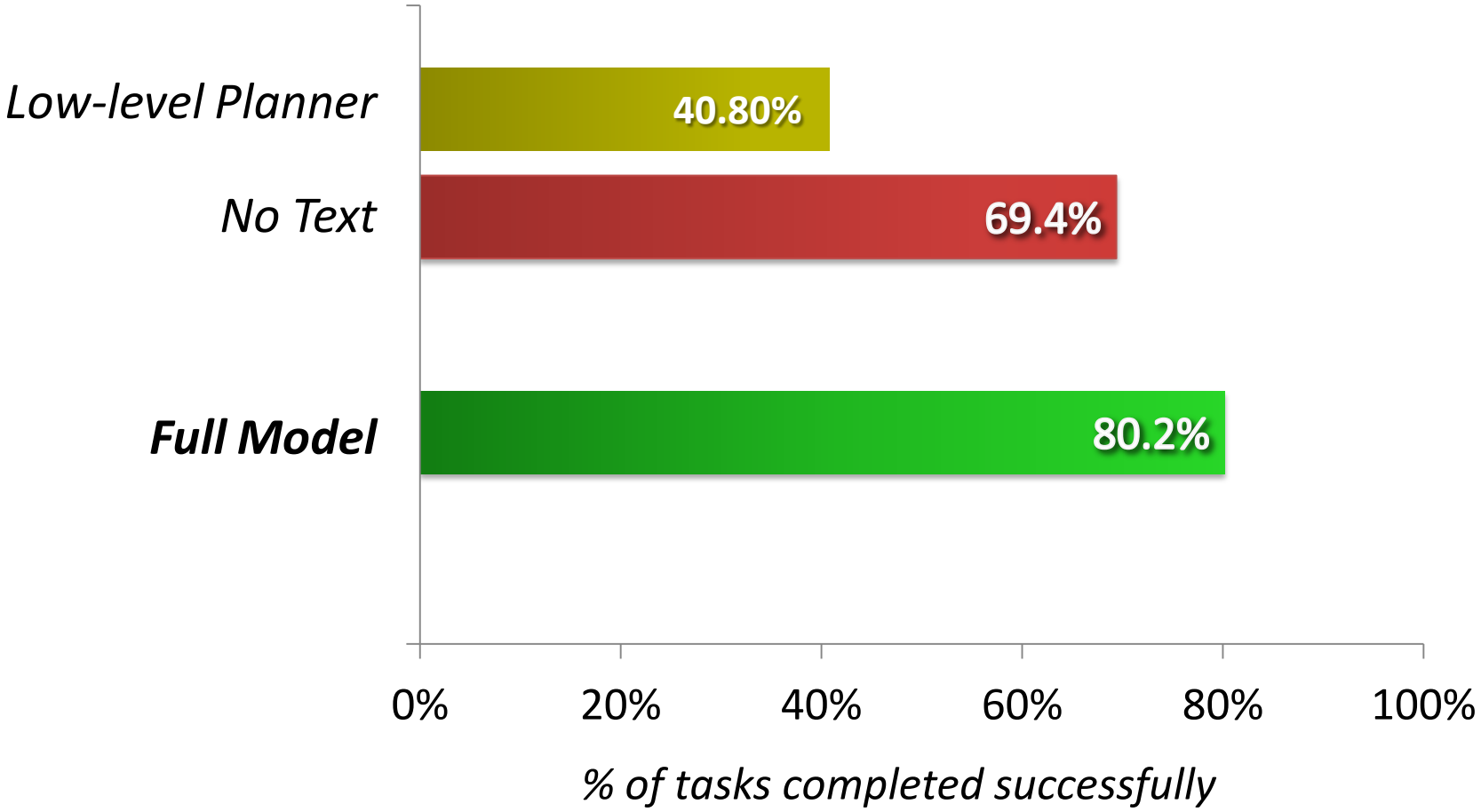
Manual Text Connections

Manually annotate all connections implied by the text Use second half of model with the manual connections

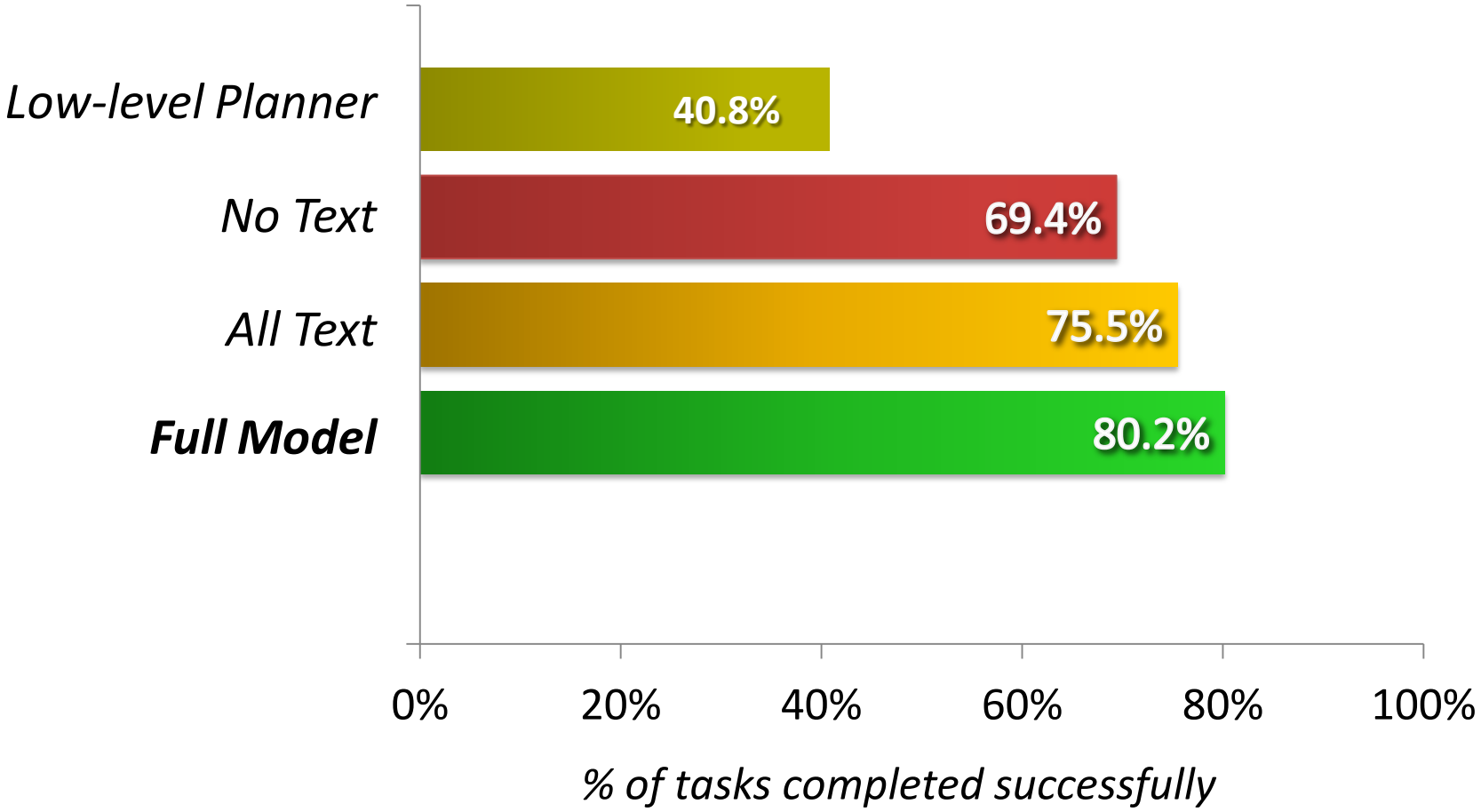
Results


Low-level Planner
40.80%
Full Model
80.2%
0%
20%
40%
60%
80%
100%
% of tasks completed successfully

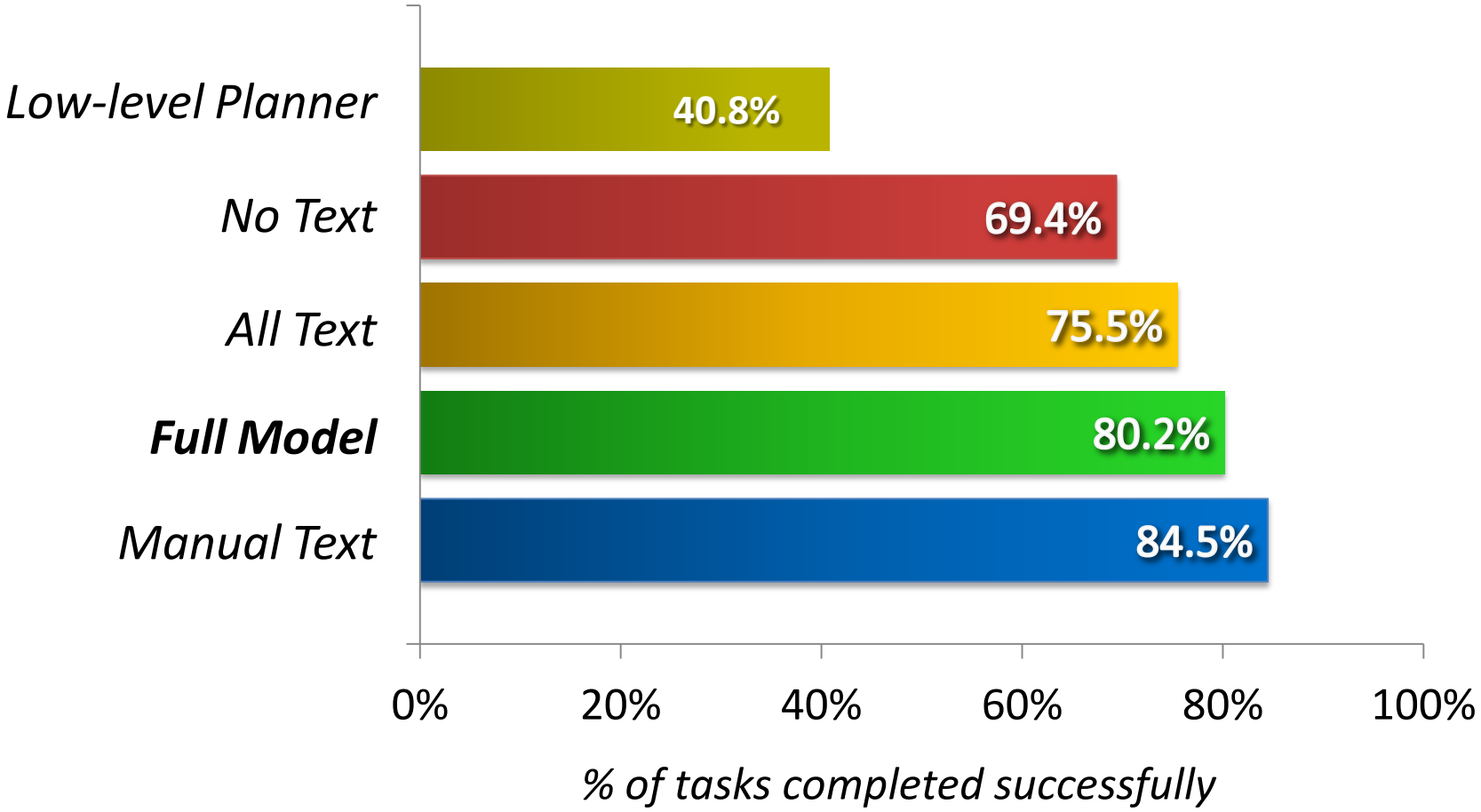
Results


Low-level Planner
40.80%
No Text
69.4%
Full Model
80.2%
0%
20%
40%
60%
80%
100%
% of tasks completed successfully

Results

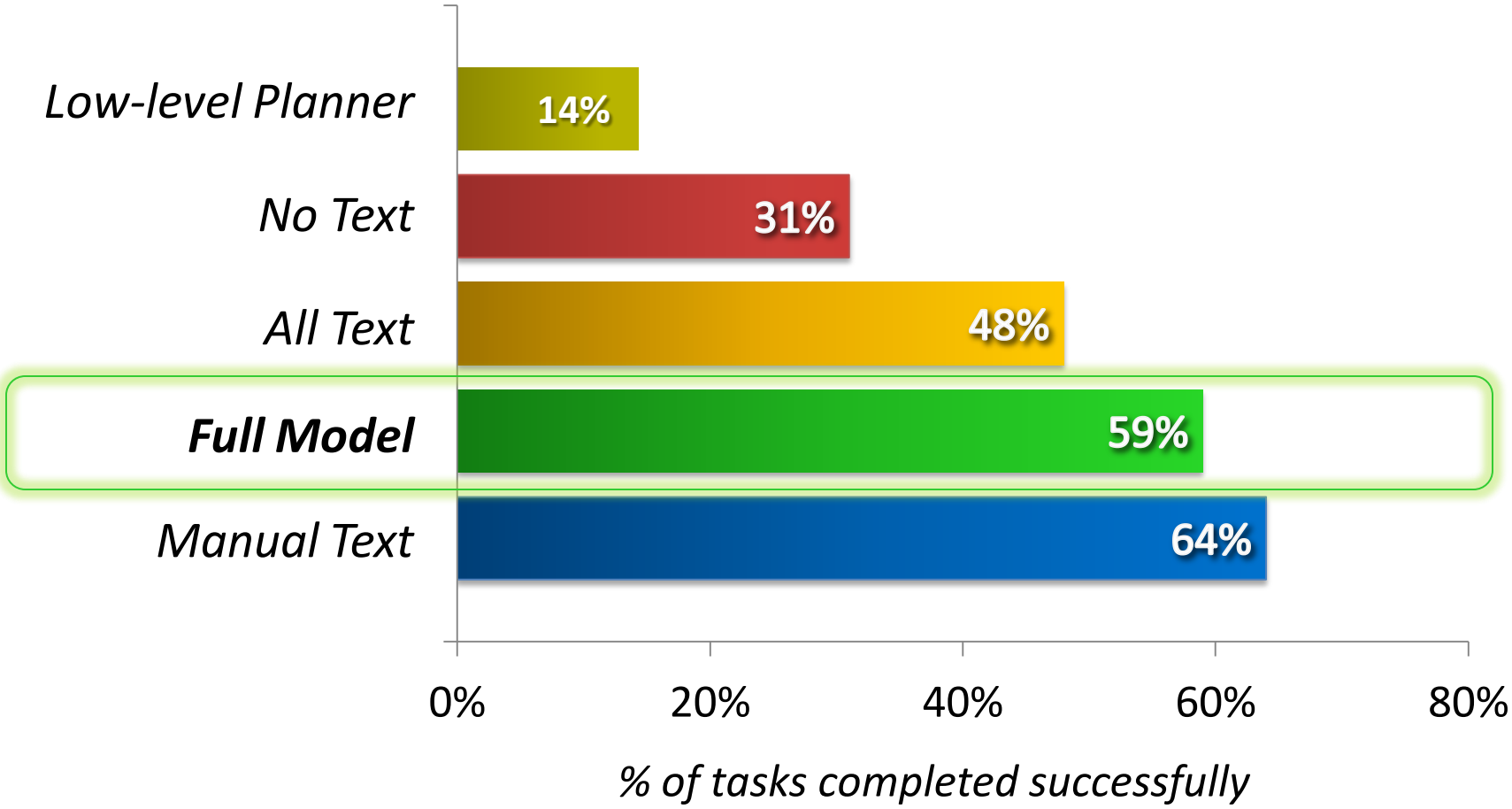

Low-level Planner
40.8%
No Text
69.4%
All Text
75.5%
Full Model
80.2%
0%
20%
40%
60%
80%
100%
% of tasks completed successfully

Results



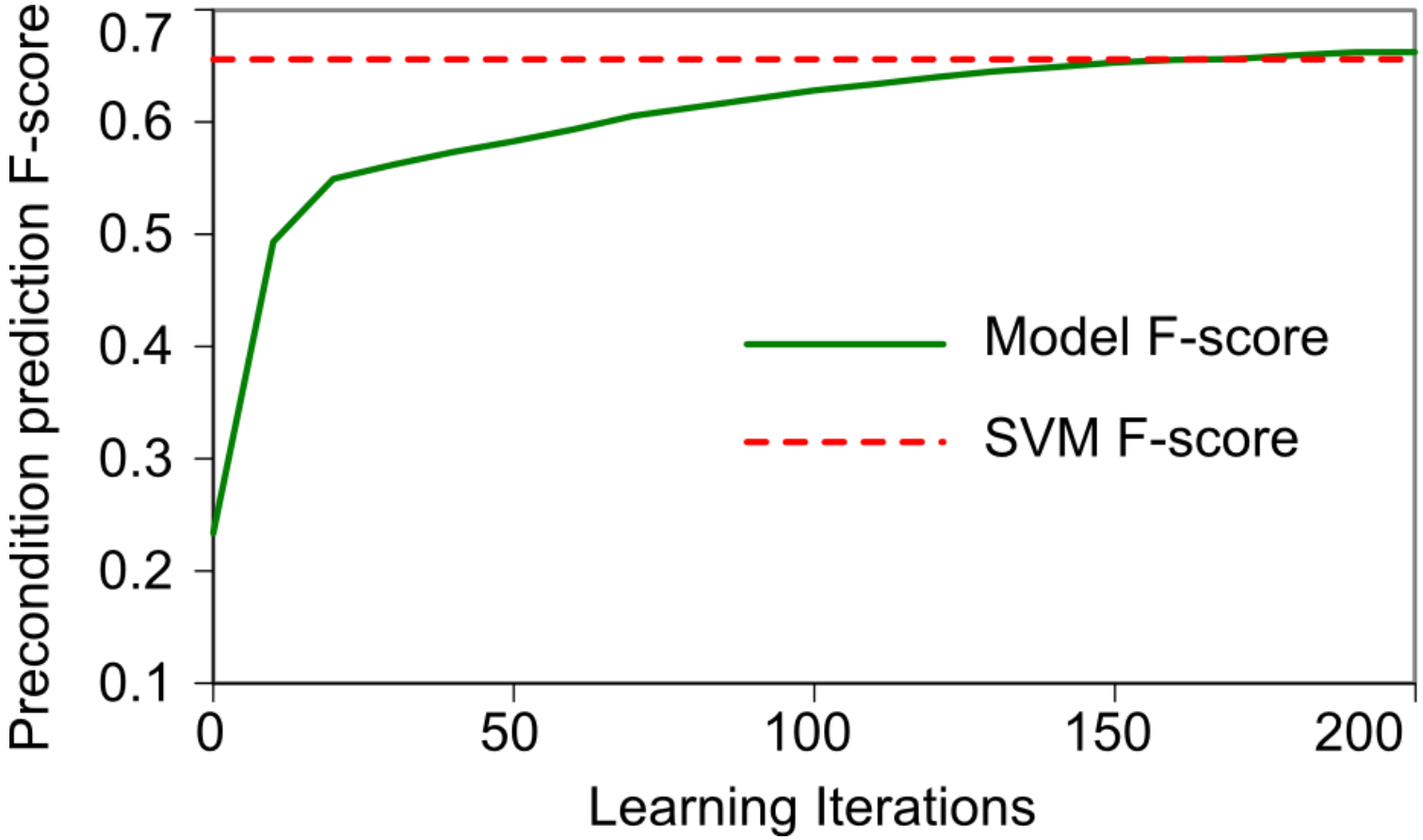
Very close to upper bound

Results: Tasks Longer Than 35 Actions



Almost twice the performance of No Text

Results: Text Analysis



Conclusion

- Our method can learn to ground textual descriptions of precondition relations
- Precondition relationship information can improve performance on complex planning tasks

Code and data available at:

http://groups.csail.mit.edu/rbg/code/planning/