UNIVERSITY OF LATVIA
FACULTY OF
COMPUTING

Systematicity, Planning, LLMs and Robots

Guntis V. Strazds

guntis_vilnis.strazds@lu.lv

October 25, 2023

Supervisor: Prof. Guntis Bārzdiņš, Dr.sc.comp

UNIVERSITY OF LATVIA

Systematicity, Planning, LLMs and Robots

1. Introduction
2. Interests and goals
3. About last year
4. Transformers for RL
5. Code gen
6. LLMs
7. Sim envs
8. TAMP
9. Recap
10. Conclusion
11. xtra slides

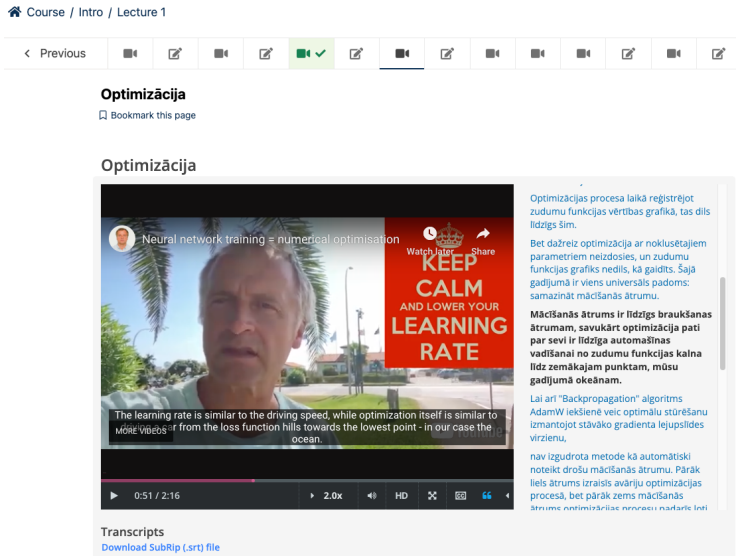
About Me

- Software Engineer @ EDI
  (1/2 time: nominally 20hrs/week ; 3 days / week)
  project: AIMS5.0
- Senior Specialist ("Vecākais eksperts") @ LU
  project: Language Technologies Initiative
  `https://digital-strategy.ec.europa.eu/lv/policies/language-technologies`
- Laboratory for Perceptual and Cognitive Systems @ DF
  (LU DF Uztveres un kognitīvo systēmu laboratorija)
  `https://www.lpcs.lu.lv/`

Artificial Intelligence in Manufacturing leading to Sustainability and Industry5.0

All in all, the 20 *AIMS5.0* UCs highlight the project's interdisciplinary nature, as they span across 9 industrial domains (including 2 cross-domain UCs) and are covered within 2 different WPs, as shown in Figure 15.

Use case	Lead	WP	Task	Domains	Title	TRL Start	TRL end
1	AIDI	5	T5.1	Manufacturing	Development of Green AI-Lifecycle Management Approaches	4	7
2	PCL	5	T5.1	Consumer Electronics	AI Enhanced Value Chain	5	7
3	HQ	5	T5.1	Cross Domain	Servitization and service delivery enabled by Zero administration	5	7
4	BMW	5	T5.2	Automotive	AI based improved connection between production and logistics		6
5	IDEKO	5	T5.3	Machinery	AI techniques at different layers in Machine Tool Domain	4	6
6	EDI	5	T5.3	Manufacturing	Reconfigurable AI-based automated conveyer feeding by a robotic system	4	6
7	Signify	5	T5.3	Manufacturing	Digital Automated Luminaire Manufacturing platform	4	7
8	SU	5	T5.3	Automotive	AI supporting and strengthening human and manufacturing cycles.	4	6
9	FUH	6	ST6.1.1	Semiconductor	Energy-aware Scheduling of Batch Processing Machines in Wafer Fabs	5	7
10	SYSTEMA	6	ST6.1.2	Semiconductor	MES concept based on AI Improving Production Cycle: AI-Prod_MES	5	7
11	TIAG	6	ST6.1.3	Food production	AI-supported Industrial IoT for Indoor Food Production	4	7
12	WUW	6	ST6.1.4	Semiconductor	Human-Centred AI for the Optimization of Robust and Competitive Semiconductor Manufacturing Networks	5	7
13	HOST	6	ST6.2.1	Robotics	Intelligent Sensors improving Robustness of Automated Wafer Transportation and Storage Systems	5	7
14	FMX	6	ST6.2.2	Cross Domain	Speeding-up the Benefit of Automation by Virtual Commissioning through Digital Twins	5	7
15	IFD	6	ST6.3.1	Semiconductor	Productionized Machine Learning models at scale	5	7
16	TRT	6	ST6.3.1	Aviation	AI enhanced production and performance of RF transceivers for civilian SATCOM and radar applications	4	7
17	FhG-IISB	6	ST6.3.3	Semiconductor	Intelligent Contamination Management & Prescriptive Control Plans and Process Flows	5	7
18	IFAG	6	T6.1.5	Semiconductor	Using semantic Data Model as base for AI optimized economic & ecologic/societal supply chain improvements	5	7
19	NXP	6	T6.1.5	Semiconductor	Extension of Digital Reference Ontology for Smart Sustainable Business Planning and incident handling	3	7
20	IFAT	6	T6.3.2	Semiconductor	Harmonization of Automated Digitized Equipment Control	5	7



Big Picture Interests and Goals

- Thesis topic (title as originally proposed):
  Tracking the state of the world [with Deep Learning models]

- **Generalizable Systematic Problem Solving**
  - Autonomous agents - goal attainment with a variety of goals and environment configurations
  - Skills learned from experience - e.g. by interacting with simulated environments
  - Ideally: predictable, explainable and easily adjustable behavior
  - (**new**): applied to robotics use cases
  - -> Learning Generalized Policies (w/ hybrid neuro-symbolic methods?)

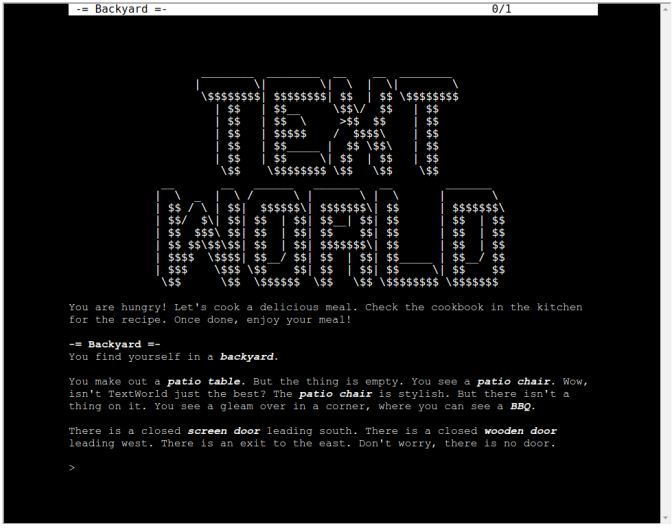
- **Need to Publish (still...)**

- **Adaptive** (partially observable and dynamic world)
- **Systematically Compositional**
  - Intelligently recombine from a repertoire of skills
- **Sequential Decision Making**
  - Perceive: sensory signals -> beliefs about state of the world
  - Think / plan (if required & time allows) ; choose an action
  - Act: attempt the intended action
    - Observe: see what happens
    - Adjust: react / re-plan accordingly
    - (maybe also learn something about the environment or the task)

TextWorld projects

- 1st idea
  - Tooling for experimenting with generalized policy learning for TextWorld
  - ... together with some initial results demonstrating usefulness
  - (Nice to Have) also integrate Minigrid
- 2nd idea
  - (Not very well defined at the time of my presentation last year) Eventually became:
    - Assess performance of pretrained LLMs on original ('obfuscated') vs. straightforward ('LLM friendly') output from TextWorld games
    - Compare performance of finetuned LLMs vs. baseline Transformer and RL models trained using simplified ('Transformer friendly') or fully abstract ('logic-programming friendly') observations
    - Compare performance of new Transformer variants specialized for sequential decision making

TextWorld – A Platform for Text Adventure Games



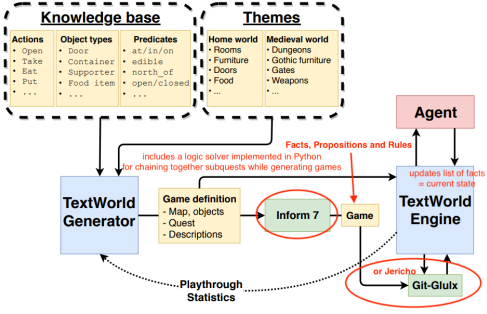
https://www.microsoft.com/en-us/research/project/textworld/try-it/

TextWorld as a Planning Domain

POMDP Game Generation

Framework for generating many different game instances, with controllable size and complexity

- Rules for generating world configurations
- Rules for quest generation, chaining subgoals
- Rules and logic engine for updating world state
- Templates for textual descriptions



(click here for list of available challenges)

Figure from Côté et al. 2019 - **TextWorld: A Learning Environment for Text-Based Games**

TextWorld - Challenging for RL

- Partial visibility - Player can see only what's in current room, only in opened containers
- Large and fairly complex action space
- Objects can be ON or IN other objects, or carried (= IN Inventory)
- Object state attributes based on object type:
  - open / closed / locked
  - cut / chopped / sliced / diced
  - roasted / baked / fried / ...
- Hierarchical type system supports multi-inheritance
- Limited inventory capacity (requires planning)
- Verb + direct object + instrument
  - **peel the purple potato with the knife** $\rightarrow$ **a peeled purple potato**
- Complex NL expressions: ***the sliced roasted yellow Idaho potato***

TextWorld projects

- 1st idea
  - Tooling for experimenting with generalized policy learning for TextWorld
  - ... together with some initial results demonstrating usefulness
  - (Nice to Have) also integrate Minigrid
- 2nd idea
  - Assess performance of pretrained LLMs on original ('obfuscated') vs. straightforward ('LLM friendly') output from TextWorld games
  - Compare performance of finetuned LLMs vs. baseline Transformer and RL models trained using simplified ('Transformer friendly') or fully abstract ('logic-programming friendly') observations
  - Compare performance of new Transformer variants specialized for sequential decision making
  - + (April 2023) Try using NN model to make progress in games if ASP solver is taking too long

ICLP 2023

The 39th International Conference on Logic Programming
`https://iclp2023.imperial.ac.uk/program`
Imperial College London, UK, July 9 - 15, 2023

- Registered abstracts for 2 potential submissions
  - 1) toolbox idea -> a paper for the Doctoral Consortium
  - 2) the 2nd project -> a short paper for the main conference
- Intense work on first one, then the other, from Jan 31 - April 27
  But, in the end, didn't actually submit either one
- Lessons learned:
  - it takes s lot of work to prepare a decent paper
  - prematurely submitting a placeholder abstract is not a good idea

ASP-based planning for TextWorld


```
% ------ CUT ------
% chop :: $in(f, I) & $in(o, I) & $sharp(o) & uncut(f) -> chopped(f)
% dice :: $in(f, I) & $in(o, I) & $sharp(o) & uncut(f) -> diced(f)
% slice :: $in(f, I) & $in(o, I) & $sharp(o) & uncut(f) -> sliced(f)

% can chop, slice or dice cuttable items in inventory
%  if player also has a knife (a sharp object)

cut_state(X,uncut,t) :- cut_state(X,uncut,t-1), not act(do_cut(t,_,X,_),t).

0 {do_cut(t,V,F,O):cutting_verb(V),cuttable(F),sharp(O),
    in(O,inventory,t-1),in(F,inventory,t-1) } 1 .

cut_state(X,chopped,t) :- cut_state(X,uncut,t-1), act(do_cut(t,chop,X,_),t).
cut_state(X,diced,t) :- cut_state(X,uncut,t-1), act(do_cut(t,dice,X,_),t).
cut_state(X,sliced,t) :- cut_state(X,uncut,t-1), act(do_cut(t,slice,X,_),t).
```

Fragment from ASP-based oracle for TextWorld cooking domain

Solves all tw-cooking games (FTWC and GATA datasets)

- FTWC training/validation/test: 4400 / 222 / 512 games
- GATA training/validation/test: 1000 / 200 / 200 games

Transformers specialized for RL / Policy learning

Yang et al. 2023 - Foundation Models for Decision Making: Problems, Methods, and Opportunities

Li et al. 2023 - A Survey on Transformers in Reinforcement Learning [arxiv:2301.03044]

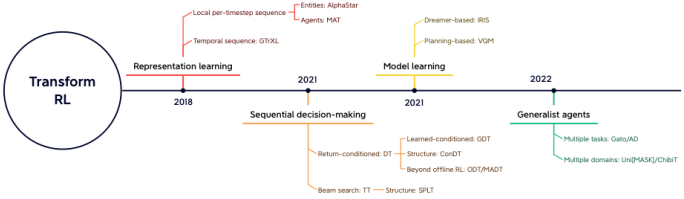


Figure 2: The taxonomy of TransformRL. The timeline is based on the first work related to the branch.

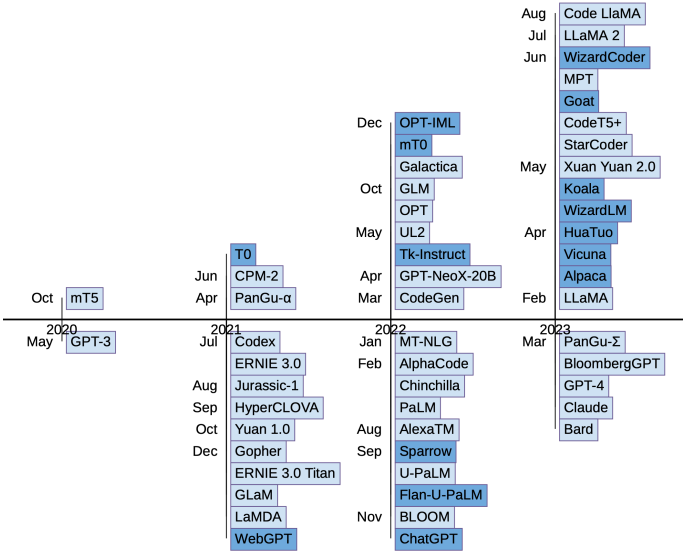
Learning Generalized Policies via code generation

- (Hoped for) benefits:
  - More systematic generalization to unseen / OOD env variations
  - Learning new tasks from a small number of examples
  - Inspectability / verifiability
- Methods for code gen or program induction
  - Dataset of input : output examples -> program
  - Oracle or Learned Black-box policy -> examples, traces -> code
  - Rule set or State machine or Logic Program induction (Inductive Logic Programming)
  - Abstracting / generalizing from specific -> more general
  - From a collection of narrowly specific programs -> library of subroutines / API
  - LLM to directly generate code from task descriptions
  - LLM to generate policy sketches -> code skeleton

The rise of the LLMs

More, Bigger, Better

Fig.2 from Naveed et al. 2023 - A Comprehensive Overview of Large Language Models



Foundation models for embodied agents

Models that combine multiple modalities into one huge model, resulting in "generalist models", which can do all kinds of things, including controlling robots
Recent examples:

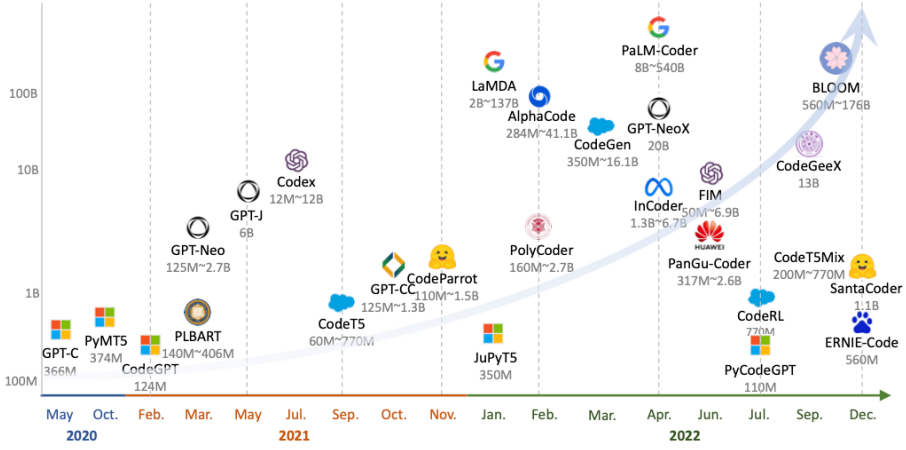
https://palm-e.github.io/

https://www.deepmind.com/blog/rt-2-new-model-translates-vision-and-language-into-action

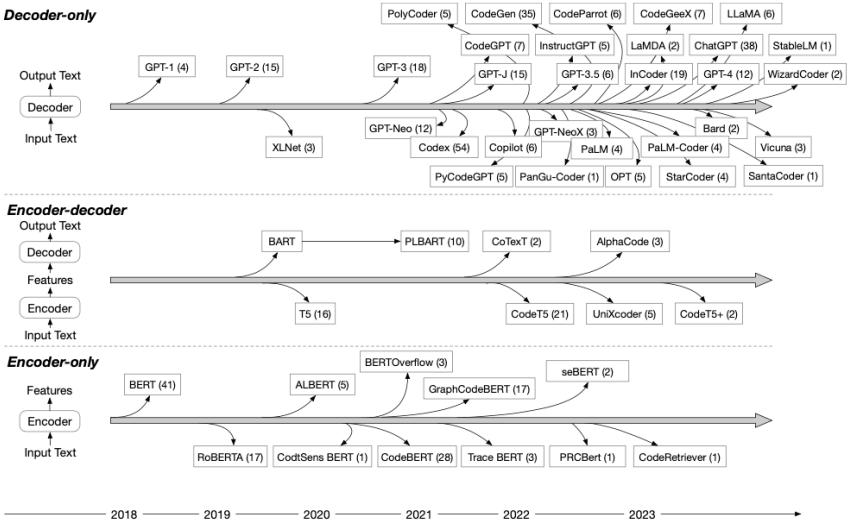
https://www.deepmind.com/blog/scaling-up-learning-across-many-different-robot-types

LLMs for Natural Language -> Code

Fig.2 from Zan et al. 2023 - Large Language Models Meet NL2Code: A Survey



CodeGen LLMs (incl. 2023)



Numbers in parens are count of papers using the model

Fig.4 from Hou et al. 2023 - Large Language Models for Software Engineering: A Systematic Literature Review

LLMs (and LVLMs) for Planning

- Xie et al. 2023 **Translating Natural Language to Planning Goals with Large-Language Models** [2302.05128]
- Wang et al. 2023 **Describe, Explain, Plan and Select: Interactive Planning with Large Language Models Enables Open-World Multi-Task Agents** [2302.01560]
- Li ea 2022 - **Pre-Trained Language Models for Interactive Decision-Making** [2202.01771]
- Pallagani ea 2022 - **Plansformer: Generating Symbolic Plans using Transformers** [2212.08681]
- Song ea 2022 - **LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models** [2212.04088]
- Huang ea 2022 - **Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents** [2201.07207]

Prompt "engineering"

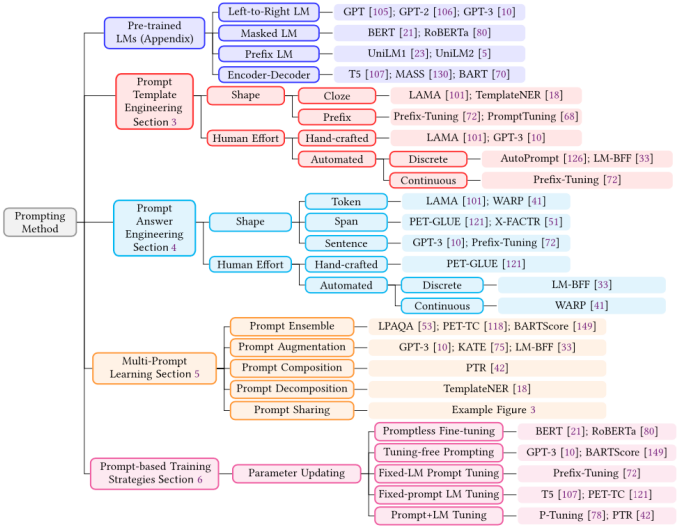


Fig.1 from Liu et al. 2023 - Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing

Prompt engineering: Voyager

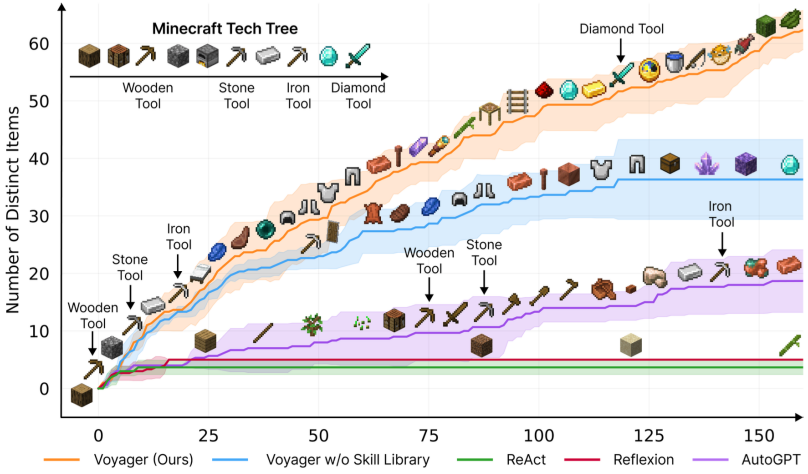


Figure 1: VOYAGER discovers new Minecraft items and skills continually by self-driven exploration, significantly outperforming the baselines. X-axis denotes the number of prompting iterations.

Wang et al. 2023 - Voyager: An Open-Ended Embodied Agent with Large Language Models

Prompt engineering: Voyager

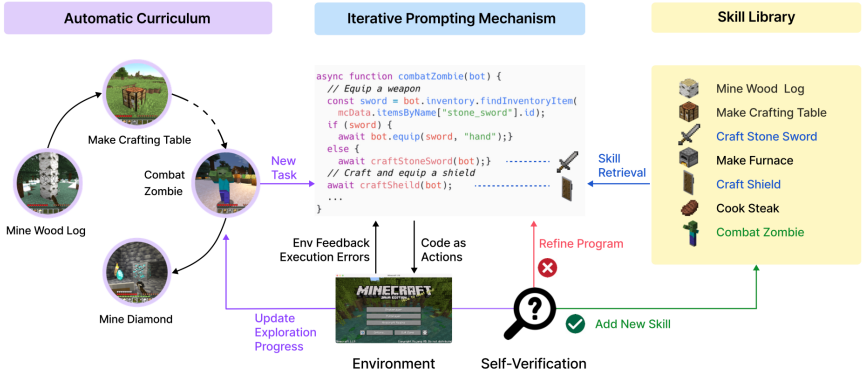


Figure 2: VOYAGER consists of three key components: an automatic curriculum for open-ended exploration, a skill library for increasingly complex behaviors, and an iterative prompting mechanism that uses code as action space.

Wang et al. 2023 - Voyager: An Open-Ended Embodied Agent with Large Language Models

Simulators and benchmarks for Robotics

Click for online slide deck: **Robotics simulation envs and benchmarks**

Task and Motion Planning

Garrett et al. 2021 - Integrated Task and Motion Planning [2010.01083]

Wang et al. 2020 - Learning compositional models of robot skills for task and motion planning [2006.06444]

Lin et al. 2023 - Text2Motion: From Natural Language Instructions to Feasible Plans [2303.12153]

Envall at al. 2023 – Differentiable Task Assignment and Motion Planning [2304.09734]

Recap of topics

	Topics
13	Logic programming (incl. Planning w/ASP)
26	Inductive Logic Programming
26	CodeGen and code induction
13	LLMs: analysis, surveys, prompt engineering
17	Planning with LLMs (incl hybrid / TAMP)
14	TAMP ; also probabilistic or hierarchical planning
14	Transformers for RL/SDM, and/or more compositional or more efficient
11	Causal learning, causal RL, Meta-RL
7	Neuro-symbolic, hybrid (general)
21	Cognitive Architectures, CogSci, BrainSci, philosophy
14	Other/Misc (diffusion, sim2real, XAI, NAS, MoE, GraphNN)
7	TextWorld or Interactive Fiction

(Topics of papers read 12.2022 - 10.2023)

What next?

- Generalized policy for TextWorld games via LLM -> code gen
  - RQ? Can LLM code gen produce a program that generalizes better than:
    1) LLMs used directly for plannnig/action selection
    2) RL or Transformer-RL baseline models
- TAMP ; hybrid neuro-symbolic, hierarchical planning

Questions/Discussion...

guntis_vilnis.strazds@lu.lv

TextWorld and Language Models

Interactive Text Games = Natural Language-based observations and actions:

- **Observations** – textual descriptions of what the player can currently see
- **Actions** – player specifies what to do next via natural language commands
- **State representation** – LLMs can facilitate encoding history of observations to a representation of the current state of the world
- **Common-sense knowledge** is helpful – which items (based on their names) can be picked up and carried, opened, sliced and diced, cooked?
- LLM as **World model** – predict effects of potential actions given current world state

Disentangling Parsing vs Planning & Reasoning

TextWorld - challenging for ML (easy for humans):

- Understanding/tracking state of the world from NL descriptions
- Incomplete initial information about the world (not fully observable)
- Reasoning/Learning about effects of actions given current world state
- Planning / acting systematically (long-range) to achieve dynamically specified goals

Research questions:

Which of the above are most problematic (for baseline RL, Imitation Learning, for LLMs)? How systematic are LLMs w/respect to planning? Can non-huge models be directly trained to do more systematic planning? Future: might similar methods be applied to improve the systematicity of mega-scale models like LLMs for reasoning and planning?

TextWorld -> Focusing on planning and systematicity

Abstracting away from natural language

- 1) Original, intentionally somewhat obfuscated NL descriptions
- 2) Straightforward Natural Language descriptions ("LLM friendly")
- 3) Simplified, schematic textual descriptions ("Transformer friendly")
- 4) Maximally abstract observations ("Logic-programming friendly")

(With decreasing complexity of NLP parsing, total vocabulary size also decreases, as well as the number of potential actions the model needs to consider.)

TextWorld - Abstracting away from NL (1)

- Original:
  -= Kitchen =- You've entered a kitchen.
  You can make out an opened fridge nearby. The fridge is empty! What a waste of a day! You can see an open conventional looking oven in the room. What a letdown! The oven is empty! You can make out a table. The table is massive. On the table you can see a cookbook. Oh, great. Here's a counter. Unfortunately, there isn't a thing on it. You scan the room, seeing a stove. But there isn't a thing on it. Aw, and here you were, all excited for there to be things on it!
  There is a closed wrought iron door leading south. There is an exit to the east. Don't worry, there is no door. You don't like doors? Why not try going west, that entranceway is not blocked by one.

TextWorld - Abstracting away from NL (2)

- LLM friendly:

  You've just walked into a kitchen.
  You see an opened fridge. The fridge is empty. An open oven, which looks conventional, is nearby. The oven is empty. You can see a table. On the table you can see a cookbook. You see a counter. But there is nothing on it. You see a stove. But there is nothing on it.
  There is a closed wrought iron door leading south. There is an exit to the east. Don't worry, there is no door. Why not try going west, that entranceway is not blocked by a door.

TextWorld - Abstracting away from NL (3)

- Transformer friendly:
  -= kitchen =- IN +open c_ fridge : nothing ; IN +open oven_ oven : nothing ; ON s_ table : o_ cookbook ; ON s_ counter : nothing ; ON stove_ stove : nothing ;
  Exits : south +closed wrought iron door to unknown , east to unknown , west to corridor ;

- Logic-programming:
  at(player, r_0, 6). at(fridge_0, r_0, 6). at(oven_0, r_0, 6). at(s_1, r_0, 6). at(s_2, r_0, 6). open(fridge_0, 6). open(oven_0, 6). on(o_0, s_1, 6). instance_of(o_0, o). instance_of(s_0, s). [...] link(r_0, d_0, unknownS). closed(d_0, 6). south_of(unknownS, r_0), west_of(r_1, r_0). east_of(unknownE, r_0).

Looks a lot like Prolog; based on different formal semantics

Language constructs

- Facts `q(42).`
- Rules `p(X) :- q(X), not r(X).`
- Conditional literals `p :- q(X) : r(X).`
- Disjunction `p(X) ; q(X) :- r(X).`
- Integrity constraints `:- q(X), p(X).`
- Choice `2 { p(X,Y) : q(X) } 7 :- r(Y).`
- Aggregates `s(Y) :- r(Y), 2 #sum{ X : p(X,Y), q(X) } 7.`
- Multi-objective optimization `:~ q(X), p(X,C). [C@42]`

`#minimize { C@42 : q(X), p(X,C) }`

Potassco

https://www.cs.uni-potsdam.de/~torsten/Potassco/Slides/asp.pdf

Lifschitz, V. (2008). Twelve Definitions of a Stable Model
`https://www.cs.utexas.edu/users/vl/papers/12defs.pdf` [1]

- Definitions A and B, in Terms of Translations into Nonmonotonic Logic
- Definition C, in Terms of the Reduct
- Definitions D and E, in Terms of Unfounded Sets and Loop Formulas
- Definition F, in Terms of Circumscription
- Definitions G and H, in Terms of Tightening and the Situation Calculus
- Definition I, in Terms of Equilibrium Logic
- Definitions J and K, in Terms of Modified Reducts
- Definition L, in Terms of Modified Circumscription

[1] In: Garcia de la Banda, M., Pontelli, E. (eds) Logic Programming. ICLP 2008. Lecture Notes in Computer Science, vol 5366. Springer, Berlin, Heidelberg.

Stable models

- A set X of atoms satisfies a propositional theory Φ, written $X \models \Phi$, if $X \models \phi$ for each $\phi \in \Phi$
- The set of all $\subseteq$-minimal sets of atoms satisfying a propositional theory Φ is denoted by $\min_{\subseteq}(\Phi)$
- A set X of atoms is a stable model of a propositional theory Φ, if $X \in \min_{\subseteq}(\Phi^X)$
- If X is a stable model of Φ, then
  - $X \models \Phi$ and
  - $\min_{\subseteq}(\Phi^X) = \{X\}$
- Note In general, this does not imply $X \in \min_{\subseteq}(\Phi)$!

- The reduct, ϕ^X, of a formula ϕ relative to a set X of atoms is defined recursively as follows:
  - $\phi^X = \bot$ if $X \not\models \phi$
  - $\phi^X = \phi$ if $\phi \in X$
  - $\phi^X = (\psi^X \circ \varphi^X)$ if $X \models \phi$ and $\phi = (\psi \circ \varphi)$ for $\circ \in \{\wedge, \vee, \rightarrow\}$
  - If $\phi = \neg\psi = (\psi \rightarrow \bot)$, then $\phi^X = (\bot \rightarrow \bot) = \top$, if $X \not\models \psi$, and $\phi^X = \bot$, otherwise

ASP for Planning

Son et al. 2022- **Answer Set Planning: A Survey**
https://arxiv.org/abs/2202.05793

Dimopoulous et al. 1997. **Encoding planning problems in nonmonotonic logic programs**. In *Proceedings of the Fourth European Conference on Planning*, Lecture Notes in Artificial Intelligence, vol. 1348. 169–181.

Dimopoulous et al. 2019. **plasp 3: Towards effective ASP planning**. Theory and Practice of Logic Programming 19, 3, 477–504.

- Planning problems can be encoded as declarative logic programs – each satisfying model contains a sequence of actions representing a valid plan.
- Non-monotonicity of ASP enables incremental re-planning: incorporating new facts discovered while stepping through the plan.

Planning with Large Language Models?

Evaluating LLM planning and reasoning capabilities

- Borji 2023 - **A Categorical Archive of ChatGPT Failures** [2302.03494]
- Valmeekam et al. - **Large Language Models Still Can't Plan** (A Benchmark for LLMs on Planning and Reasoning about Change) [2206.10498]
  - Correct plans: GPT-3 3/500 (0.6%) ; Instruct-GPT3: 25/500 (5%) ; BLOOM: 1/200 (0.5%)

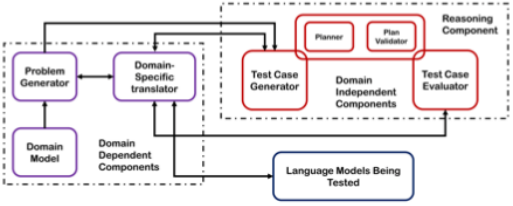


Figure 2: The diagrammatic overview of the overall test framework. Broadly, our system consists of a domain-specific component that allows the generation of various instances of the specific PDDL planning problems and the translation of the PDDL information to text and back. The domain-independent component is responsible for generating the various specific test instances that will be

"LLMs can't plan very well", or can they?

Some success is being reported

- Li ea 2022 - **Pre-Trained Language Models for Interactive Decision-Making** [2202.01771]
  - VirtualHome & BabyAI
  - Improves combinatorial generalization: for out-of-distribution tasks involving new combinations of goals, states or objects, LM pre-training improves task completion rates by 43.6%
- Pallagani ea 2022 - **Plansformer: Generating Symbolic Plans using Transformers**
  - Fine-tuned CodeT5 (a masked language model trained on code) for symbolic plan generation (PDDL for Blocksworld, Towers of Hanoi, Grippers, Driverlog).
  - Towers of Hanoi: (97% valid plans, out of which 95% are shortest length plans) [NOTE: no Natural language in input or output]
- Wang et al. (2023) **Describe, Explain, Plan and Select: Interactive Planning with Large Language Models Enables Open-World Multi-Task Agents**
  - LLM as Explainer and Planner + RL trained goal Selector + goal conditioned low-level controller (RL or IL trained)
  - can robustly accomplish 70+ Minecraft tasks