

Tradução Automatizada de PDDL para planejamento em HTN baseada em Grandes Modelos de Linguagem

Isadora Schwaab¹, André G. Pereira¹

¹Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

ischwaab@inf.ufrgs.br, agpereira@inf.ufrgs.br

Abstract. *This work investigates the use of Large Language Models (LLMs) to translate classical planning problems in PDDL into hierarchical representations in HDDL. We developed a tool to automate request submission, process the responses, and evaluate the rate of solvable problems. The experiments involved prompt variations with different levels of exemplification and request complexity. The results show that the evaluated models can effectively act as planning assistants, achieving success rates of up to 100% in certain scenarios, thus highlighting the potential of the proposed approach.*

Resumo. *Este trabalho investiga o uso de Grandes Modelos de Linguagem (LLMs) para traduzir problemas de planejamento clássico em PDDL para representações hierárquicas em HDDL. Desenvolvemos uma ferramenta para automatizar o envio de requisições, processar as respostas e avaliar a taxa de problemas solucionáveis. Os experimentos envolveram variações de prompts com diferentes níveis de exemplificação e complexidade. Os resultados demonstram que os modelos avaliados podem atuar de forma eficaz como auxiliares de planejamento, alcançando taxas de até 100% de problemas resolvidos em determinados cenários, o que evidencia o potencial da abordagem proposta.*

1. Introdução

O conceito de planejamento é explorado no campo da Inteligência Artificial (IA) como o processo de deliberação que escolhe e organiza uma sequência de ações pela antecipação de seu efeito em um ambiente, visando atingir um objetivo pré-determinado. Planejamento clássico descreve um segmento desse campo de estudo, no qual os problemas (situações descritas formalmente por um estado inicial e um objetivo final) se restringem a ambientes determinísticos, discretos, finitos, estáticos e totalmente observáveis. Problemas deste tipo podem ser solucionados automaticamente - desde que descritos em um formalismo adequado - com algoritmos de busca genéricos e com auxílio de funções heurísticas também genéricas (que não exigem conhecimento de domínio).

Contudo, observamos que formalismos que incorporam conhecimento de domínio são potencialmente mais poderosos. Problemas de planejamento hierárquico, por exemplo, não se baseiam em explorar todo um espaço de estados na busca por um estado específico, mas em completar uma sequência pré-determinada de tarefas que levará ao resultado desejado. Isso facilita a busca pela solução, pois evita a exploração de estados e ações desnecessárias. Assim, automatizar definição de uma rede inicial de tarefas para um problema clássico qualquer é uma forma de introduzir conhecimento de domínio.

Modelos de aprendizado de máquina, sobretudo Grandes Modelos de Linguagem (LLMs), representam uma transformação significativa no estudo da IA. Seu modo de funcionamento não se caracteriza pelo raciocínio e ponderação cuidadosa ao resolver problemas, mas pelo caráter intuitivo e rápido. Considerando isso, nesse trabalho estudamos a criação uma ferramenta eficiente de transformação de problemas de planejamento clássico em problemas de planejamento hierárquico, tendo as LLMs como base para a criação de redes iniciais de tarefas adequadas.

2. Background

Nessa seção, apresentamos conceitos fundamentais para o entendimento do trabalho, incluindo definições de planejamento clássico e hierárquico.

2.1. Planejamento clássico e Planning Domain Definition Language

O planejamento clássico concentra-se em domínios finitos, determinísticos e totalmente observáveis. Um domínio desse tipo pode ser definido por dois conjuntos finitos: o conjunto de estados atingíveis e o conjunto de ações (com suas pré-condições e efeitos no ambiente). Um problema de planejamento clássico, derivado de determinado domínio, é especificado por um estado inicial e uma fórmula objetivo. A solução pode ser obtida por meio de um algoritmo de busca genérico, que parte do estado inicial e expande o espaço de estados — relacionados entre si pelas ações e seus efeitos — até alcançar um estado que satisfaça a fórmula objetivo. Um meio amplamente utilizado para a definição de problemas clássicos é a linguagem formal PDDL (Planning Domain Definition Language) [Haslum et al. 2019]. Em PDDL, cada estado é descrito por um conjunto de predicados; já os predicados e ações disponíveis são definidos em um arquivo de domínio, enquanto as descrições do estado inicial e da fórmula objetivo ficam no arquivo do problema.

2.2. Planejamento Hierárquico, Hierarchical Domain Definition Language e PANDA

No planejamento hierárquico, surgem conceitos adicionais fundamentais para a definição de domínios: métodos, que vinculam tarefas, e tarefas, que podem ser decompostas por métodos ou, se forem primitivas, executadas diretamente como ações. A ideia do planejamento HTN (Hierarchical Task Network) é decompor uma rede inicial de tarefas: uma tarefa não primitiva pode ser substituída, por meio de um método associado, por uma sequência de novas tarefas. O objetivo da busca é aplicar métodos de modo a gerar uma sequência coerente de tarefas primitivas (executáveis). Uma rede de tarefas não vazia, mas na qual nenhum método pode ser aplicado, é considerada uma falha, levando ao backtracking na busca. Neste trabalho, consideramos apenas sequências de tarefas totalmente ordenadas.

A HDDL (Hierarchical Domain Definition Language)[Höller et al. 2020] é uma linguagem proposta com base na sintaxe PDDL, mas para planejamento hierárquico. As descrições dos estados também seguem a lógica de predicados, e a sintaxe de definição para métodos, tarefas e rede de tarefas é semelhante à definição de ações em PDDL. A linguagem também é base para múltiplos planejadores autônomos, como o sistema PANDA [Höller et al. 2021].

3. Proposta

Buscamos automatizar a “tradução” de problemas de planejamento entre os formalismos PDDL e HDDL, utilizando LLMs para criar as redes iniciais de tarefas. A premissa é explorar a capacidade das LLMs em auxiliar a definição de tarefas por facilitar a incorporação de conhecimento de domínio através de seu caráter intuitivo. O sucesso dos experimentos indicaria que o uso de LLMs pode viabilizar uma automatização mais eficiente de solução de problemas de planejamento clássico quaisquer.

Desenvolvemos uma ferramenta em Python que gera automaticamente prompts para um conjunto de problemas em PDDL, envia cada um deles a um LLM para obter a versão correspondente em HDDL e, em seguida, submete as respostas ao sistema PANDA, registrando os resultados (erro de parsing, erro de grounding, tempo limite excedido, problema insolúvel ou resolvido). Os prompts, exemplificados na caixa abaixo, mantêm uma estrutura comum, mas podem variar quanto ao número de recursos incluídos: além do problema em PDDL e das definições de domínio em PDDL e HDDL, é possível acrescentar exemplos de sintaxe HDDL, traduções completas de outras tarefas do mesmo domínio ou traduções parciais que deixam em aberto apenas a rede inicial de tarefas.

```
1 You are a highly-skilled professor in AI planning that can convert any planning problem written in Planning Domain
  ↳ Definition Language (PDDL) to its Hierarchical Task Network (HTN) representation (in HDDL), elaborating an
  ↳ initial task network that helps finding a solution more efficiently.
2 I am going to provide you 6 files that contain, respectively:
3 1. a blocksworld planning problem with a classical planning representation in PDDL (between the <pddl-problem> and
  ↳ </pddl-problem> tags);
4 2. a representation of the blocksworld planning domain, written in PDDL (between the <domain-pddl> and
  ↳ </domain-pddl> tags);
5 3. a Hierarchical Task Network (HTN) representation of the blocksworld planning domain, written in Hierarchical
  ↳ Domain Definition Language - HDDL (between the <domain-hddl> and </domain-hddl> tags);
6 4. a part of the HTN representation for the given PDDL problem (between the <partial-hddl-problem> and
  ↳ </partial-hddl-problem> tags);
7 5. an example of another blocksworld planning problem, written in PDDL (between the <example-pddl> and
  ↳ </example-pddl> tags);
8 6. an example of 'translation' of that PDDL example problem to HDDL (its HTN representation, between the
  ↳ <example-hddl> and </example-hddl> tags);
9 Provide the full Hierarchical Task Network representation for the given PDDL problem (first file), including a
  ↳ coherent initial task network to the part that was given (fourth file).
10 Be aware of the following constraints:
11 - provide the HDDL representation of the problem inside <hddl-problem> and </hddl-problem> tags;
12 - do not provide any additional text or explanation.
13 Pay attention to the following concern on object definition: HTN representation is case-sensitive, while PDDL is
  ↳ not. This means that the form (uppercase or lowercase) of the predicates, tasks and methods in a PDDL problem
  ↳ doesn't necessarily have to be the same as in the PDDL domain, but the form in the HDDL problem must be the same
  ↳ as the form defined in the HDDL domain;
14
15 <pddl-problem>
16 ...
```

4. Experimentos

Os experimentos foram conduzidos em 6 domínios de planejamento distintos, cada um composto por 10 problemas adaptados de uma coleção de benchmarks da Competição Internacional de Planejamento (IPC). Foram avaliadas 6 variações de prompts, divididas em dois grupos: nas variações 1, 2 e 3, requereu-se a tradução completa do problema; já nas variações 4, 5 e 6, apenas a rede inicial de tarefas foi solicitada (com parte da tradução já fornecida). Entre essas variações, 1 e 4 não incluíam exemplos, 2 e 5 apresentavam apenas um exemplo de sintaxe (um problema HDDL qualquer do mesmo domínio), e 3 e 6 continham um exemplo completo de tradução (um problema PDDL qualquer e sua versão em HDDL, também do mesmo domínio).

Cada variação foi testada em 3 iterações com dois modelos: Gemini 2.5 Flash e Gemini 2.5 Flash-Lite. Os resultados estão apresentados nas Tabelas 1 e 2, em que cada célula indica a taxa de problemas resolvidos pelo planejador PANDA, calculada

Tabela 1. Taxas de problemas resolvidos retornados pelo Gemini 2.5 Flash

Domínio	Prompt 1	Prompt 2	Prompt 3	Prompt 4	Prompt 5	Prompt 6
Barman-BDI	3.45%	76.67%	100.00%	90.00%	75.86%	63.33%
Blocksworld-GTOHP	55.17%	83.33%	66.67%	51.72%	82.76%	80.00%
Depots	6.67%	23.33%	26.67%	36.67%	20.69%	31.03%
Logistics-Learned-ECAI-16	3.33%	96.67%	96.67%	66.67%	86.67%	93.33%
Rover-GTOHP	23.33%	90.00%	90.00%	76.67%	73.33%	80.00%
Satellite-GTOHP	16.67%	70.00%	70.00%	56.67%	70.00%	83.33%

Tabela 2. Taxas de problemas resolvidos retornados pelo Gemini 2.5 Flash-Lite

Domínio	Prompt 1	Prompt 2	Prompt 3	Prompt 4	Prompt 5	Prompt 6
Barman-BDI	0.00%	93.33%	96.67%	83.33%	100.00%	100.00%
Blocksworld-GTOHP	0.00%	0.00%	0.00%	20.00%	0.00%	0.00%
Depots	0.00%	0.00%	6.67%	20.69%	16.67%	20.00%
Logistics-Learned-ECAI-16	0.00%	96.67%	96.67%	100.00%	70.00%	63.33%
Rover-GTOHP	0.00%	58.62%	60.00%	75.86%	79.31%	80.00%
Satellite-GTOHP	0.00%	0.00%	6.67%	46.67%	20.00%	26.67%

sobre o total agregado das 3 iterações (cerca de 30 execuções), após a tradução com a configuração correspondente. As variações observadas evidenciam que o desempenho da ferramenta depende diretamente tanto do domínio de planejamento quanto do modelo utilizado. Além disso, a inclusão de exemplos mostrou-se mais relevante para a tradução completa das tarefas do que para a elaboração da rede inicial.

O modelo Flash-Lite revelou-se praticamente incapaz de realizar traduções corretas sem exemplos. Entretanto, apresentou resultados satisfatórios em determinados domínios quando os prompts eram mais completos. Ainda assim, em domínios como Blocksworld e Depots, seu desempenho foi limitado por falhas no ordenamento da rede inicial; no caso de Depots, esse problema também afetou o modelo Flash. Por outro lado, o modelo Flash, embora não atingisse taxas de 100% de sucesso com a mesma frequência que o Flash-Lite, obteve uma média de desempenho superior, mostrando-se globalmente mais consistente na tradução correta dos problemas.

5. Conclusão

O experimento buscou avaliar a utilidade das LLMs como auxiliares em planejamento, visando explorar um novo meio para resolver problemas clássicos de forma mais eficiente. Os resultados evidenciam o forte potencial dos modelos mais avançados nesse papel, sobretudo quando orientados por prompts completos e bem exemplificados.

Referências

- Haslum, P., Lipovetzky, N., Magazzeni, D., and Muise, C. (2019). *An Introduction to the Planning Domain Definition Language*.
- Höller, D., Behnke, G., Bercher, P., and Biundo, S. (2021). The PANDA framework for hierarchical planning. *KI-Künstliche Intelligenz*.
- Höller, D., Behnke, G., Bercher, P., Biundo, S., Fiorino, H., Pellier, D., and Alford, R. (2020). HDDL: an extension to PDDL for expressing hierarchical planning problems. In *AAAI Conference on Artificial Intelligence*.