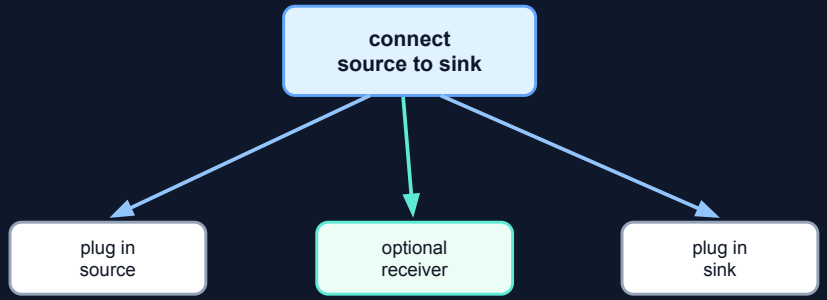
Hierarchical Planning in AI

One abstract idea, many concrete realizations



Based on Bercher, Alford & Höller (IJCAI 2019)

Yanal Doghouz

Why this belongs in AI

AI planning

How can an agent choose actions that achieve its goals?

Knowledge representation

How do we encode what the agent knows about tasks, actions, constraints, and procedures?

Autonomous systems

Robots, assistants, workflows, services, and agents need plans that are not only possible, but structured.

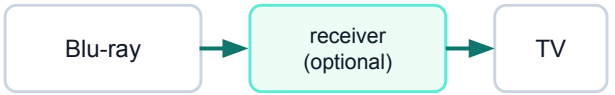
Hierarchical planning sits at the intersection of search, action, abstraction, and procedural knowledge.

Sources: Bercher et al. 2019; Ghallab, Nau & Traverso 2004

Two ways to tell a planner what you want

State-based goal

Task-based goal



Connect Blu-ray to TV

Specify structure, not just outcome

Same domain. Different notion of success.

Running example adapted from Bercher et al. 2019

Roadmap for the lecture

1. **Classical planning**
   The state-based baseline
2. **HTN planning**
   Tasks, methods, and refinement
3. **Why it matters**
   Expressivity, complexity, and valid solutions
4. **Solving and variants**
   Search styles, insertion, sharing, HGN/GTN
5. **Modern AI connection**
   Robotics, workflows, and LLM agents

Classical planning: the baseline

P = (F, A, s_I, g)

Four ingredients

F: facts about the world
A: actions
s_I: initial state
g: goal description

Tiny running example

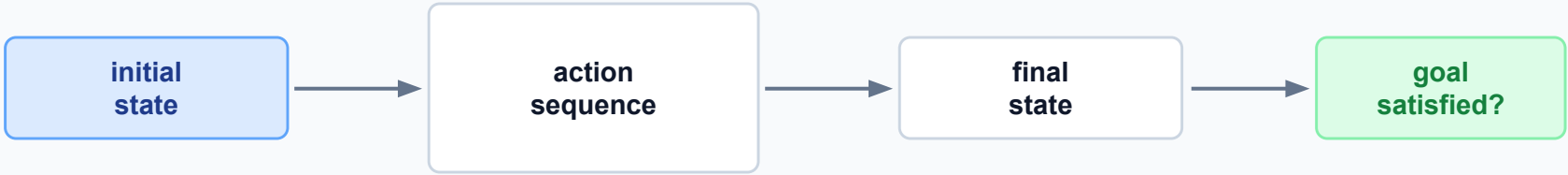
Facts might include:
free(port1)
free(port2)
compatible(port1, port2)
connected(port1, port2)
hasSignal(TV)

Goal: hasSignal(TV)

Classical planning is primarily state-based: reach a world state satisfying the goal.

Source: Bercher et al. 2019, Sec. 2.1

How a classical plan is judged



Example action: plugIn

pre: free(p1), free(p2), compatible(p1,p2) add: connected(p1,p2) del: free(p1), free(p2)

A classical solution is an executable action sequence that reaches a goal state.

Source: Bercher et al. 2019, Sec. 2.1

What classical planning leaves out

A final state does not tell us everything about the procedure.

Classical question

Did we reach the goal state?

Procedural question

Was the solution structured correctly?
Did it follow an allowed procedure?
Can we encode expert knowledge directly?

Same goal state does not imply same solution structure.

Source: Bercher et al. 2019, Sec. 2.2

Hierarchical planning: the core idea

Classical planning

Facts
Actions
Initial state
Goal state

Hierarchical planning

Facts
Actions
Initial state
Initial task network
Task hierarchy

The planner must refine abstract tasks into executable actions.

Source: Bercher et al. 2019, Sec. 2.2

Three core ingredients

Primitive tasks

Executable actions
Directly affect the state

Example:
plugIn(cable, port, device)

Compound tasks

Abstract tasks
Not executed directly
Must be refined

Example:
connect(Blu-ray, TV)

Methods

Rules for decomposing compound tasks into smaller task networks

Example:
connect -> direct
connect -> via receiver

Primitive = action **Compound = abstraction** **Method = decomposition rule**

Source: Bercher et al. 2019, Sec. 2.2-2.3

Decompose one task

Initial compound task:

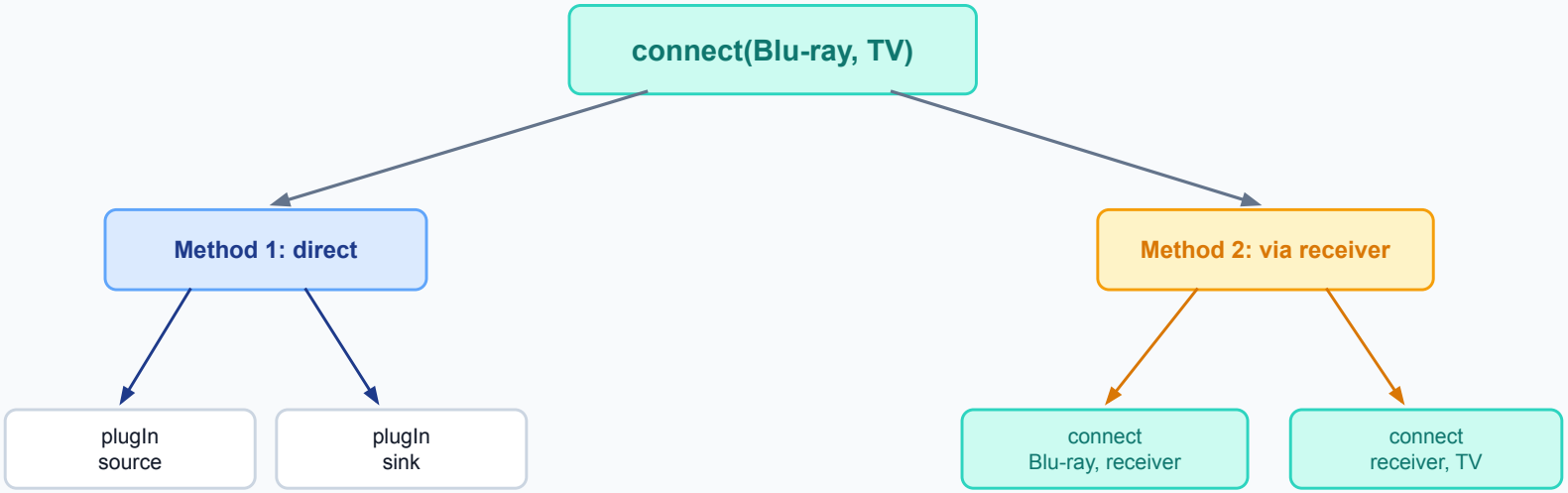
connect(Blu-ray, TV)

Question:

What are reasonable ways to break this task down?

Running example adapted from Bercher et al. 2019

Example: decomposing an abstract task



The planner chooses decompositions until only primitive executable tasks remain.

Source: Bercher et al. 2019, Fig. 1 and Sec. 2.2

What counts as a valid HTN solution?

Key distinction

A plan is not valid just because it works. It must also be reachable by allowed decomposition.

Source: Bercher et al. 2019, Def. 3

The decisive difference

Plan A

Reaches the goal state
Generated by allowed decomposition

=> valid HTN solution

Plan B

Reaches the goal state
Not generated by the hierarchy

=> not a valid HTN solution

Hierarchy changes the solution set, not only the search process.

Source: Bercher et al. 2019, Sec. 2.3

Why this matters: expressivity and complexity

Expressivity

What kinds of solution structures can be represented?

Can the model rule out plans that merely work but violate the intended procedure?

Complexity

More expressive models are often harder to solve.

Different hierarchical formalisms can have very different computational behavior.

Changing the solution criterion changes the formalism itself.

In general HTN planning, the expressive power is strong enough to model undecidable problems.

Sources: Bercher et al. 2019; Erol, Hendler & Nau 1996

How HTN problems are solved

Search in hierarchical space

Manipulate task networks directly
Choose decompositions
Enforce ordering and executability

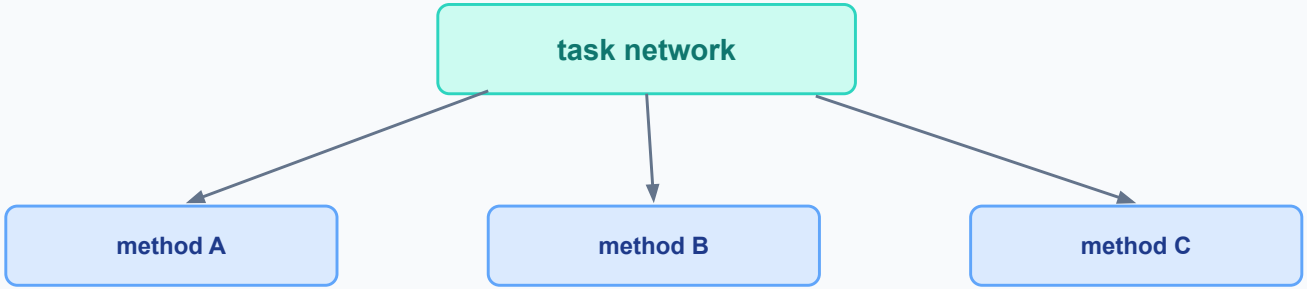
Compile into another formalism

Bound the hierarchy
Translate into classical planning, SAT, or ASP
Use existing solvers

Solve the hierarchy directly, or encode it into something else.

Source: Bercher et al. 2019, Sec. 2.4

Search style 1: decomposition-based search



Idea

Search over possible refinements until an executable primitive task network is produced.

Representative systems discussed in the survey include PANDA and FAPE.

Sources: Bercher et al. 2019; Bercher et al. 2014; Bit-Monnot et al. 2020

Search style 2: progression-based search

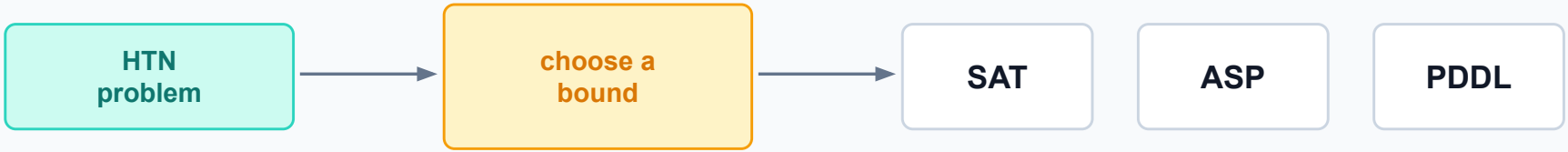
Idea

Mix refinement with execution in a left-to-right, state-progressing manner.

SHOP and SHOP2 are classic progression-oriented HTN planners.

Sources: Bercher et al. 2019; Nau et al. 2003

A third route: bounded compilation



Why bounds are needed

General HTN planning is too expressive for a single complete translation into a decidable target formalism without restrictions.

Source: Bercher et al. 2019, Sec. 2.4

The central design question

Is hierarchy a rule or a suggestion?

Hierarchy as a rule

The hierarchy defines what counts as a correct solution.

Not every working plan is acceptable.

Hierarchy as guidance

The hierarchy guides the planner, but flexibility may be allowed.

Inserted or shared structure may be acceptable.

The variants differ by how they answer this question.

Source: Bercher et al. 2019, Sec. 3

Variant 1: task insertion (TIHTN)

Standard HTN

Every primitive action in the final plan must come from decomposition.

Strict structural control.

TIHTN

The planner may insert actions directly.

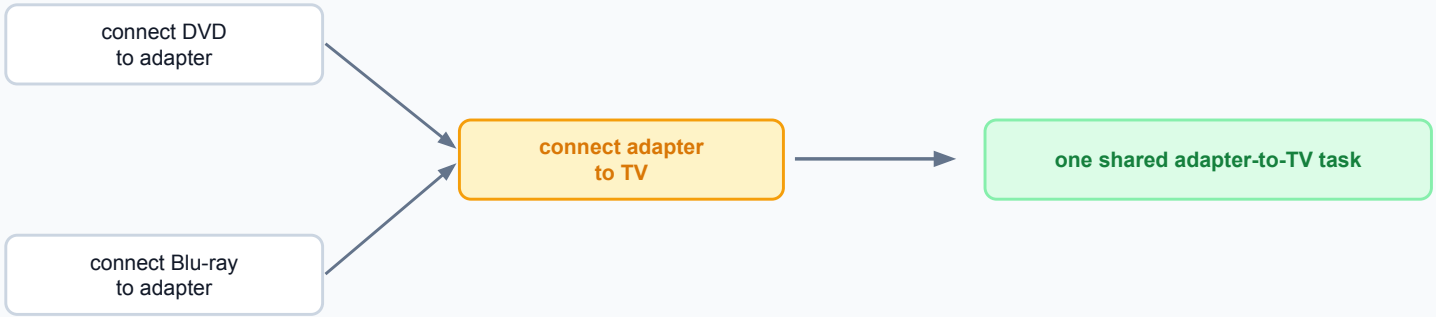
More flexibility, but the hierarchy is less strict.

Key question: can the planner add actions the hierarchy did not explicitly require?

Source: Bercher et al. 2019, Sec. 3.1

Variant 2: task sharing

When two branches need the same subtask, should it be duplicated or shared?



Idea: identical subtasks do not always need to be executed or represented redundantly.

Source: Bercher et al. 2019, Sec. 3.2

Variant 3: compound tasks with state semantics

Standard compound task

Abstract placeholder
Refined by methods
No direct preconditions/effects

Enriched compound task

Can have preconditions
Can have effects
Supports reasoning at higher abstraction levels

Benefit: detect state-level problems before fully decomposing every abstract task.

Source: Bercher et al. 2019, Sec. 3.3

Alternative hierarchy designs: HGN and GTN

HTN

Hierarchy over tasks

Refine compound tasks into subtasks and actions.

HGN

Hierarchy over goals

Refine goals into structured subgoals.

GTN

Goals + tasks

Combine goal decomposition and task decomposition.

Hierarchy need not live only on tasks.

Source: Bercher et al. 2019, Sec. 3.4-3.5

Where hierarchical planning shows up

Robotics

procedures for acting in the physical world

Space missions

activity planning for Mars rovers

Web services

compose services into executable workflows

Narratives

generate structured stories and discourse

Assistants

step-by-step instructions for users

Workflows

model and verify process structure

The common theme: abstract procedures must be refined into executable behavior.

Source: Bercher et al. 2019, Introduction

Modern AI connection: LLM agents also decompose tasks

LLM-agent planning

Recent LLM-agent work often uses:
- task decomposition
- plan selection
- external tools
- reflection and memory

Classical planning lesson

Decomposition is useful, but structure alone is not enough.

We also need executability, constraints, and verifiability.

Hierarchical planning gives a formal language for a pattern that modern agents use informally.

Sources: Huang et al. 2024; Bercher et al. 2019

LLM decomposition vs HTN decomposition

Dimension	LLM-style decomposition	HTN-style decomposition
Representation	Natural language plans	Formal task networks
Strength	Flexible, broad prior knowledge	Verifiable structure and constraints
Weakness	May hallucinate or skip feasibility	Requires explicit modeling effort
Best use	Open-ended assistance	Domains where correctness matters

Impressive framing: modern agents rediscover decomposition, but classical planning tells us what it means formally.

Sources: Huang et al. 2024; Bercher et al. 2019

What this paper contributes

A map of the landscape

HTN and variants
Task vs goal hierarchies
Alternative solution criteria

A comparison framework

What changes?
What becomes more expressive?
What becomes harder?

Pointers to practice

Search styles
Compilation methods
Representative planners

The contribution is conceptual organization, not one new algorithm.

Source: Bercher et al. 2019

Main takeaways

1. Classical planning is state-based; hierarchical planning adds task structure.
2. In HTN planning, a plan must not only work - it must be a valid refinement.
3. Changing the role of hierarchy changes expressivity and complexity.
4. Hierarchical planning is a family of formalisms, not one single method.

Hierarchy changes what counts as a plan.

Discussion

Prompt 1

When should hierarchy define correctness, and when should it only guide search?

Prompt 2

Is extra expressivity worth the added complexity?

Prompt 3

How should planners and LLM agents be combined?

Questions?

Acronyms

Formalism	Main hierarchy	Key property
Classical planning	No hierarchy	Goal-state based
HTN	Task hierarchy	Valid decomposition required
TIHTN	Task hierarchy + insertion	More flexible. Hierarchy less strict
HGN	Goal hierarchy	Goals decompose into subgoals
GTN	Task + goal hierarchy	Combined structure

References and further reading

Bercher, P.; Alford, R.; Höller, D. (2019). A Survey on Hierarchical Planning - One Abstract Idea, Many Concrete Realizations. IJCAI.

Ghallab, M.; Nau, D.; Traverso, P. (2004). Automated Planning: Theory and Practice. Morgan Kaufmann.

Erol, K.; Hendler, J.; Nau, D. (1996). Complexity Results for HTN Planning. Annals of Mathematics and AI.

Nau, D. et al. (2003). SHOP2: An HTN Planning System. Journal of Artificial Intelligence Research.

Höller, D. et al. (2019). HDDL - A Language to Describe Hierarchical Planning Problems.

Bit-Monnot, A. et al. (2020). FAPE: A Constraint-based Planner for Generative and Hierarchical Temporal Planning.

Huang, X. et al. (2024). Understanding the Planning of LLM Agents: A Survey.