

Advancements in Hierarchical Task Network Planning Research Since 2020

Executive summary

Hierarchical Task Network (HTN) planning research since 2020 is characterized by (a) standardization and benchmarking via the Hierarchical Domain Definition Language (HDDL) and repeated International Planning Competition HTN tracks, (b) renewed progress in solver technology through translation-based methods, symbolic and SAT/MaxSAT-style optimal search, and improved heuristic search techniques, and (c) a visible expansion toward verification, repair, uncertainty, time, and learning-oriented hybridization. ¹

Two competition cycles anchor empirical evaluation. IPC 2020 introduced a dedicated hierarchical planning competition, motivating a common input language (HDDL), a decomposition-aware plan output format, a verifier, and an initial benchmark suite. ² IPC 2023 repeated HTN tracks, reusing most IPC 2020 domains while adding new ones, and reported winners across total-order and partial-order tracks with multiple evaluation modes (agile, satisficing, optimal). ³

On solver technology, major advances include: competitive translations from HTN to classical planning and new translation techniques for both totally ordered and partially ordered HTNs; symbolic optimal solving for totally ordered HTNs; and renewed attention to heuristic and structural issues such as loop detection, landmark generation, and limitations of canonical optimal-search paradigms (e.g., A* incompleteness under HTN-specific cyclicities). ⁴

HTN-centric “reasoning tasks around planning” form a prominent post-2020 theme: plan verification compiled into planning; plan repair comparisons; model correction/model repair; and complexity analyses for plan verification, bounded plan existence, and related decision problems in lifted vs grounded HTN settings. ⁵

Extensions to uncertainty and time gained clearer formalisms and first-generation solving approaches: fully observable nondeterministic (FOND) HTN formalization; practical strong-solution methods for FOND HTN; and a probabilistic extension analyzed from a knowledge-representation perspective. Temporal HTN work coalesces around a proposed HDDL 2.1 extension to capture temporal and numeric constructs needed for operational domains, though standardization remains in-progress. ⁶

Learning and hybrid approaches appear in three identifiable patterns: (1) RL tooling that converts HDDL problems into Gym-style environments (including multi-agent protocol extensions) to study planning-learning combinations, (2) domain/model generation and method acquisition using large language models (LLMs), with evidence that hierarchical modeling remains substantially harder for LLMs than classical (non-hierarchical) modeling on shared benchmarks, and (3) HTN planning systems that query LLMs for decompositions while maintaining symbolic soundness constraints. ⁷

Foundations and major technical advances in HTN solving

A key enabling factor for post-2020 progress is benchmark and language consolidation around HDDL and competition-driven infrastructure. IPC 2020 explicitly motivated a unified input language, decomposition-aware output format, and verification tooling, and described the need to create a comprehensive benchmark set because one “no such set existed before.” ² The same benchmark lineage continues in IPC 2023 HTN tracks, where organizers report reusing most IPC 2020 domains and adding new domains for both total-order and partial-order tracks. ⁸

Major solver-side advances since 2020 cluster into four technique families (often composed in pipelines):

Translation-based HTN solving and verification-as-a-backstop.

Recent work improves the competitiveness of translating HTN problems into classical planning, emphasizing that translation must address expressivity mismatches and that prior translations were not “on par” with dedicated HTN planners. It introduces new translation techniques for totally ordered and partially ordered HTNs, including results on reducing action blow-ups (linear-vs-exponential action generation in at least one comparison to prior encodings). ⁹ Translation-based systems can rely on post-hoc HTN plan verification when the translation over-approximates the HTN solution set (a pattern visible in TOAD’s workflow description). ¹⁰

Symbolic and SAT/MaxSAT-driven optimal HTN solving.

Optimal HTN planning receives renewed attention post-2020. A representative AAAI paper proposes a symbolic-search approach for optimal totally ordered HTN planning and links the challenge to the need for informative admissible heuristics that respect task-network structure. ¹¹ More recent ICAPS work (and associated preprints) introduces SAT/MaxSAT-style greedy optimal search approaches for TOHTN, including variants described as SAT-based (SibylSat) and MaxSAT-based (SibylSatOpt). ¹²

Heuristic-search refinements for HTN progression and structure.

Loop detection is evaluated as a performance technique for heuristic progression search on IPC 2020 benchmarks and reported to increase performance in both total-order and partial-order settings for the PANDA system. ¹³ Landmark generation is revisited for HTN planning at ICAPS 2025, targeting “facts that must hold true” and “tasks or methods that must be included” in every solution, proposing updated landmark-generation methods relative to earlier approaches. ¹⁴

Reassessment of canonical optimal-search assumptions.

ICAPS 2025 work analyzes how hierarchy-induced cyclicities can break expected guarantees: it proves incompleteness of A* even with a perfect heuristic for totally ordered HTN problems under the studied conditions and highlights cycle types induced by task hierarchies as key structural factors. ¹⁵

Mermaid timeline of selected milestones (2020–2025):

```

timeline
    title Selected milestones in HTN planning since 2020
    2020 : HDDL standardization + IPC 2020 HTN competition infrastructure
    (language, verifier, benchmarks)
    2021 : Symbolic optimal TOHTN; loop detection for HTN progression; FOND HTN
```

formalization and complexity

2022 : Plan verification compiled into HTN planning; translation techniques improved to match dedicated planners

2023 : Temporal HDDL 2.1 proposal (time + numeric constructs); IPC 2023 HTN tracks reuse and extend benchmarks

2024 : Practical strong-solution approach for FOND HTN; complexity and model correction results for HTN artifacts

2025 : Probabilistic HTN formalization/complexity; A* incompleteness; HTN landmarks revisited; HTN plan repair comparisons; RL/LLM hybrids (HDDL Gym, ChatHTN)

16

New problem formulations and theoretical advances

A prominent post-2020 theme is expanding HTN planning beyond “plan existence” to a broader set of tasks: verification, repair, model correction, and richer reasoning problems over hierarchical artifacts.

Nondeterministic and policy-based HTN planning.

ICAPS 2021 formalizes fully observable nondeterministic HTN planning and studies solution criteria and complexity for HTN problems with nondeterministic outcomes. ¹⁷ This line supports later solver work that treats solutions as policies (mapping histories or states to actions/method choices) rather than fixed action sequences. ¹⁸

HTN plan verification as a first-class reasoning task.

Compiling HTN plan verification into HTN planning is presented as a way to reuse planning technology to decide whether an action sequence is a valid HTN plan; the ICAPS 2022 paper positions verification as hard in HTN settings (up to NP-hard) and evaluates a compilation against prior approaches. ¹⁹ Subsequent complexity-focused work studies lifted HTN plan verification and bounded plan existence (and variants) more systematically, including ICAPS 2025 results that emphasize tighter bounds and advantages of working in lifted (non-grounded) representations for some tasks. ²⁰

Bounded existence, optimality verification, and structural tractability.

AAAI 2024 work analyzes the computational complexity of plan verification, bounded plan optimality verification, and bounded plan existence for HTN planning, explicitly treating lifted vs grounded representations. ²¹ IJCAI 2025 extends complexity study with additional structural and parameterized perspectives, reporting polynomial-time algorithms for specific structural classes and an algorithmic lifting meta-theorem from primitive to general task networks (under stated preconditions). ²²

Model correction and modeling assistance framed as planning.

SoCS 2024 work proposes correcting hierarchical domains with missing actions and frames the problem as producing an optimal set of corrections given a flawed hierarchical domain and a plan; it presents both the problem and a practical solving approach. ²³

Plan repair as a differentiated family of semantics.

ICAPS 2025 work compares multiple HTN plan repair algorithms and argues they correspond to different

definitions of the plan repair problem (hence distinct search spaces and solvable cases), with an empirical evaluation on benchmark problems. ²⁴ A parallel HPlan analysis (2024) similarly compares plan repair approaches (qualitatively and quantitatively) and helps contextualize the semantic differences. ²⁵

Learning, neural, and symbolic–subsymbolic hybrids

Work since 2020 shows three integration patterns: learning for heuristics/control, learning for model construction, and interactive hybrids where symbolic HTN planning provides constraints around learned generation.

RL + hierarchical planning via problem-to-environment toolchains.

HDDLgym is presented as a Python tool that generates OpenAI Gym environments from HDDL domains/problems, enabling RL policies to interact with hierarchical planning structures and explicitly supporting multi-agent scenarios via an agent-centric extension/protocol layered on HDDL. ²⁶ The paper frames the tool as filling a gap between RL evaluation ecosystems (Gym) and hierarchical planning models (HDDL/HTN). ²⁷

HTN guidance for multi-agent RL and bidirectional coupling.

A representative line integrates HTN planning with multi-agent reinforcement learning for cooperative strategies, explicitly claiming HTN can guide exploration (reducing exploration space) while MARL can assist planning. ²⁸ This line is not anchored to the planning competitions, but reuses HTN/HDDL constructs to structure multi-agent decision making. ²⁹

LLMs for hierarchical model generation and decomposition.

Two distinct roles are visible:

1) LLMs generating hierarchical models.

The L2HP framework extends L2P-style LLM-driven model generation to hierarchical planning. On the PlanBench dataset, it reports that parsing success is roughly comparable between classical and hierarchical settings (on the order of ~36%), but hierarchical syntactic validity is substantially lower (reported as ~1% vs ~20% in one reported comparison). ³⁰ The results imply that hierarchical constraints (task/method structure and decomposition correctness) remain a major barrier for naive PDDL-to-HDDL style generation workflows. ³¹

2) LLMs generating decompositions inside an HTN planner.

ChatHTN interleaves symbolic HTN planning with queries to an LLM that proposes decompositions when symbolic methods are missing; it claims soundness of produced plans despite approximate LLM decompositions and provides an open-source implementation. ³²

Mermaid schematic of a common hybrid architecture pattern (symbolic HTN + learned proposals + verification/feedback):

```
graph TD
    A[Goal tasks / initial task network] --> B[Symbolic HTN solver]
    B --> C[Decomposition search over task networks]
    C --> D{Need guidance?}
```

```

D --> E[Symbolic guidance: heuristics, landmarks, loop detection]
D --> F[Learned guidance: RL policy or LLM decomposition proposal]
F --> C
E --> C
C --> G[Candidate primitive plan + decomposition trace]
G --> H[HTN verification / execution]
H --> I[Traces + failures + metrics]
I --> F

```

33

Unspecified or limited literature: learned domain-independent heuristics for HTN planning at top-tier ML venues (e.g., NeurIPS ³⁴ main track) appear less prominent in this source set than in planning venues; similarly, large-scale neural benchmark suites specialized for HTN (analogous to widely used classical-planning learning suites) are not strongly represented in primary sources cited here.

Probabilistic, temporal, and multi-agent HTN extensions

Nondeterminism and FOND HTN planning.

The ICAPS 2021 formalization of fully observable nondeterministic HTN planning provides a base for later practical methods. ³⁵ IJCAI 2024 work claims the first practical approach for strong solutions to HTN problems with nondeterministic action outcomes, introducing a search algorithm and a compilation that relaxes a FOND HTN problem to a deterministic one so that existing deterministic HTN grounders and heuristics can be reused. ³⁶

Probabilistic HTN planning.

KR 2025 formalizes probabilistic hierarchical planning problems by building upon nondeterministic HTN formalisms and attaching probability distributions to outcomes, then studies complexity under different observability conditions. ³⁷ This provides a knowledge-representation baseline rather than a full solver ecosystem. In the present source set, solver-focused probabilistic HTN work at top planning conferences is **unspecified or limited literature** relative to deterministic and FOND HTN work.

Temporal and numeric HTN planning via HDDL 2.1.

A temporal/numeric extension proposal argues that HDDL (as originally defined) lacks constructs similar to PDDL 2.1 needed for real-world applications requiring temporal and numerical constraints. It proposes HDDL 2.1, inspired by PDDL 2.1 and ANML, and positions the effort as an open-source project with syntax, benchmarks, and a parser integrated into PDDL4J. ³⁸ Within this source set, temporal HTN planning remains centered on proposal/standardization efforts; large-scale standardized shared-task evaluation for temporal HTN is **unspecified or limited literature**.

Multi-agent HTN planning and agent-centric HDDL.

HDDL Gym proposes an agent-centric extension/protocol that modifies HDDL domain typing and operator conventions to support multi-agent configurations (including centralized and decentralized planning in the tool design) and demonstrates usage on both IPC-derived domains and Overcooked-style environments. ³⁹ More generally, IPC infrastructure remains largely centered on single-agent HTN semantics with total-order/partial-order method structures; multi-agent semantics for HDDL itself are not presented as a

standardized competition language in the cited IPC sources, so multi-agent HDDL standardization is **unspecified or limited literature**.

Benchmarks, datasets, evaluation trends, and reproducibility

Competition benchmark suites and their structure.

The IPC 2020 HTN competition is explicitly described as creating shared infrastructure: a common input language (HDDL), a decomposition-aware plan-output format, a verifier, and a benchmark set assembled because none existed as a common reference point. ² The IPC 2020 domains repository lists both total-order and partial-order domain sets and documents licensing on a per-domain basis (where available). It also notes that some benchmark instances can be too hard to ground under “reasonable time and memory constraints,” making grounding scalability an evaluation bottleneck. ⁴⁰

IPC 2023 HTN tracks reuse much of the IPC 2020 domain set while adding new domains (two new total-order and two new partial-order domains are listed on the track site). ⁴¹ The AI Magazine IPC 2023 overview reports that the HTN track selected 22 total-order domains and 10 partial-order domains for evaluation, with most drawn from IPC 2020 and a small set of new domains introduced. ⁴²

Evaluation modes and leaderboards.

The IPC 2023 HTN track website reports winners and runner-ups across total-order satisficing/agile/optimal and partial-order satisficing/agile/optimal tracks, and provides per-domain performance tables. ⁴³ These track designs expose a persistent empirical distinction between total-order and partial-order HTN planning (with separate competition tracks and different domain sets). ⁴⁴

Reproducibility practices and limitations.

IPC 2023 HTN tracks explicitly use containerization (Apptainer) to promote reproducibility and simplify compilation, and they publish planner repositories “as used in the IPC evaluation.” ⁴⁵ However, the same track documentation notes that some planner configurations cannot be redistributed as containers when they require proprietary components such as IBM CPLEX. ⁴⁵ This creates a structural reproducibility constraint even when source code is available.

A contrasting practice is visible in research papers that explicitly release experimental setups as citable artifacts (e.g., Zenodo “experimental setup” releases for a FOND HTN solver evaluation). ⁴⁶

Tools and datasets beyond competitions.

HDDL Gym positions IPC HTN domains as a reusable resource base for RL experiments in hierarchical contexts. ²⁶ HDDL Parser (language server/toolkit) claims validation against IPC 2023 hierarchical domains and reports detection of a “critical error” in at least one domain, indicating that benchmark curation and model validation remain nontrivial even with standardized syntaxes. ⁴⁷

Table of key post-2020 papers and artifacts

Work (title)	Lead authors	Year	Venue	Problem formulation	Core method	Datasets / benchmarks	Miscellaneous
HDDL: An Extension to PDDL for Expressing Hierarchical Planning Problems	Daniel Höller ⁴⁸ et al.	2020	AAAI Conference on Artificial Intelligence ⁴⁹	Domain language for HTN planning (HDDL)	Language + formalization enabling shared tooling	Used as the standard language in IPC HTN benchmarks	Enfin la ac be ec (a im
Symbolic Search for Optimal Total-Order HTN Planning	Gregor Behnke ⁵¹ & David Speck ⁵²	2021	AAAI Conference on Artificial Intelligence ⁴⁹	Optimal planning for totally ordered HTN	Symbolic-search approach for optimal TOHTN	Compared against optimal/heuristic baselines in the paper	Re en an su pr ap op 5
Loop Detection in the PANDA Planning System	(Höller) & (Behnke)	2021	International Conference on Automated Planning and Scheduling ⁵⁴	Deterministic HTN progression search	Exact/approx loop detection for HTN progression	IPC 2020 benchmark sets	Re in pe PA TO tr 20 be 1
Fully Observable Nondeterministic HTN Planning—Formalisation and Complexity Results	Dillon Chen ⁵⁵ & (Bercher)	2021	ICAPS	HTN with nondeterministic action outcomes (FOND HTN formalization)	Formalization + solution criteria + complexity	Primarily theoretical	En fin un lin lit is co
Compiling HTN Plan Verification Problems into HTN Planning Problems	(Höller) et al.	2022	ICAPS	HTN plan verification	Compilation from verification to planning; evaluation vs prior approaches	Benchmark set introduced/used for verification evaluation	Re co ou pr ve ap ev

Work (title)	Lead authors	Year	Venue	Problem formulation	Core method	Datasets / benchmarks	M fin re
Making Translations to Classical Planning Competitive with Other HTN Planners	(Behnke) et al.	2022	AAAI	HTN→classical translation, TO and PO	New translation versions; grounded translation; reduced blow-ups	Evaluated relative to state-of-the-art planners	CL th pe ga tr re th pl pe re ac tr PO ex pr
HDDL 2.1: Towards Defining a Formalism and a Semantics for Temporal HTN Planning	Damien Pellier ⁵⁶ et al.	2023	arXiv (proposal)	Temporal/ numeric HTN planning	Proposal to extend HDDL with PDDL 2.1/ ANML-inspired constructs; benchmarks + parser mentioned	Mentions benchmarks and parser integrated in PDDL4J	En fin un lin lit (p st fo
Modeling Assistance for Hierarchical Planning: Correcting Hierarchical Domains with Missing Actions	Songtuan Lin ⁵⁷ et al.	2024	International Symposium on Combinatorial Search ⁵⁸	HTN model correction given flawed domain + plan	Computes optimal set of corrections under assumptions	Experimental evaluation reported in paper	Re pr ap co m in do
Plan verification / bounded plan existence / bounded optimality verification complexity	(Lin) et al.	2024	AAAI	Complexity of HTN reasoning tasks in lifted vs grounded settings	Complexity bounds for multiple tasks	Theoretical focus with formal results	En fin un lin lit (c th

Work (title)	Lead authors	Year	Venue	Problem formulation	Core method	Datasets / benchmarks	M fin re
Solving FOND HTN in practice: grounding, search, heuristics, benchmarks	Mohammad Yousefi ⁵⁹ & (Bercher)	2024	IJCAI	FOND HTN strong solutions (policy solutions)	Search + compilation relaxing FOND HTN to deterministic to reuse tools	Includes benchmarks and “experimental setup” artifact	CL pr ap st sc re ex se Ze
A Structural Complexity Analysis of HTN Planning	Cornelius Brand ⁶¹ et al.	2025	IJCAI	Verification, plan existence, reachability in HTN	Structural and parameterized complexity; tractable fragments	Theoretical focus	En fin un lin lit (th 2
On the Incompleteness of A* for Total-Order HTN Planning	(Yousefi) et al.	2025	ICAPS	Optimal search behavior in TOHTN	Identifies hierarchy-induced cycles; proves A* incompleteness under studied conditions	Experimental setup reported as public	Re er av ce fo in pr
Landmark Generation in HTN Planning Revisited	Victor Scherer Putrich ⁶² et al.	2025	ICAPS	HTN landmarks (facts/tasks/ methods necessary in all solutions)	Updated landmark generation techniques	Evaluation in paper	Re la ge re ex ap
SibylSatOpt: MaxSAT-based greedy optimal search for TOHTN	Gaspard Quenard ⁶³ et al.	2025	ICAPS	Optimal TOHTN solving	Greedy optimal search using MaxSAT	Evaluation (paper)	En fin un lin lit ac so ab 6

Work (title)	Lead authors	Year	Venue	Problem formulation	Core method	Datasets / benchmarks	M fin re
Probabilistic HTN Planning: Formalization and Complexity	(Yousefi) et al.	2025	KR ⁶⁵	Probabilistic HTN under observability assumptions	Formalization + computational complexity analysis	Theory-centric	En fin un lin lit (fo co
HDDL Gym: Generating Gym environments from HDDL	Ngoc La ⁶⁶ et al.	2025	ICAPS	RL + hierarchical planning; multi-agent HDDL protocol	Automatic HDDL→Gym tool; multi-agent extension protocol	Demonstrations include IPC-derived domains + Overcooked	Re de de po en st hi pl co
HTN Plan Repair Algorithms Compared	Paul Zaidins ⁶⁷ et al.	2025	ICAPS	HTN plan repair with differing semantics	Theoretical mapping between algorithms and definitions + empirical comparison	Benchmark planning problems	Re di so pr ru co be al
ChatHTN: Interleaving approximate LLM decompositions with symbolic HTN planning	Héctor Muñoz-Avila ⁶⁸ et al.	2025	NeuS ⁶⁹ (PMLR)	Symbolic HTN with LLM-generated decompositions	Sound hybrid planner + open-source implementation	Evaluation: unspecified or limited literature in accessible sources	CI so de ap LL pr so im ³

Table of influential open-source HTN tools and repositories

Tool / system	Language	License	Core features (reported)	Repo / official artifact
SHOP3	Common Lisp	Mozilla Public License (MPL)	HTN planner; releases mention HDDL-format plan generation; supports repeatable RNG for replicable experiments	70
PANDA π engine (pandaPIengine)	C / C++	BSD-3-Clause	HTN planning engine; configurable compilation options (SAT/BDD; ILP heuristics requiring CPLEX); pipeline expects parser+grunder	71
PANDA π parser (pandaPIparser)	C++	BSD-3-Clause	Parses HDDL; outputs PANDA internal format; translates to (J)SHOP2 and HPDL (with limits for partial orders)	72
PANDA compilation tools for PGR/repair/verification (pandaPIpgrRepairVerify)	C++	BSD-3-Clause	Compiles plan/goal recognition, plan repair, and verification into HTN problems; guides solution prefixes	73
TOAD planning system	C++	Unspecified or limited literature (license file not shown in repo root)	Translates TOHTN to classical planning; may require verification step; uses Fast Downward-style SAS+ output	74
HDDLgym	Python (+ notebooks)	MIT	HDDL→Gym environment generation; multi-agent protocol; demonstrations on IPC domains and Overcooked	75

Tool / system	Language	License	Core features (reported)	Repo / official artifact
HDDL Parser / language server toolkit (koala-planner/HDDL-Parser)	Rust	Unspecified or limited literature (license not shown in repo root)	CLI + language server binaries; type checking and error localization; validated on IPC 2023 domains per repo text	76
IPyHOP	Python	BSD-3-Clause	Re-entrant planning / replanning from partial decomposition trees; produces solution/ decomposition tree	77
T-HTN timeline-based multi-robot planner	C++ (repo reports mixed languages)	Unspecified or limited literature (no LICENSE shown in repo root listing)	Timeline-based multi-robot HTN planner; HDDL variant usage noted in repo	78
ChatHTN implementation	Python (built on PyHOP per repo text)	Apache-2.0	Hybrid LLM + symbolic HTN planner; open-source reference implementation	79

Applications and real-world deployments

Service coordination and modular planning systems.

SH is described as an open-source, domain-independent system for HTN planning with a modular architecture, supporting parsing, grounding, plan generation, and integration functions intended for real-world challenges. [80](#) The existence of an openly maintained codebase suggests a focus on engineering and deployment concerns beyond competition evaluation. [81](#)

Safety- and norm-constrained procedural planning.

Dynamic HTN planning for distributed data transfer and utilization is proposed as a way to incorporate legal and ethical norms, representing data-transfer tasks as total-order HTN decomposition rules and updating plans online when new information changes plan validity. [82](#)

Human-centered companion and intention tracking.

HTN planning is applied to intention tracking in general aviation companion systems, exploring the use of HTN-encoded procedural knowledge to track pilot intention and address challenges in onboard companion technologies. [83](#)

Planning-learning experimental deployments.

HDDL Gym’s demonstrations include IPC-derived domains and an Overcooked setting, positioning the tool as a bridge between hierarchical planning models and RL training/evaluation workflows. ²⁶ This is best interpreted as a research deployment (tool demonstration + evaluation) rather than an industry deployment.

Unspecified or limited literature: documented, peer-reviewed industrial deployments of HTN planners since 2020 are not strongly represented in the planning-venue primary sources cited here, aside from software-engineering oriented system papers (e.g., SH) and domain papers in workshops or applied venues.

Open research directions and recommended next steps

Open research directions

Grounded in the cited post-2020 literature, open directions cluster around limitations exposed by benchmarks, theory, and emerging hybrid use cases:

Scaling grounding and lifted reasoning.

Benchmark repositories explicitly warn that some HTN instances are too hard to ground within reasonable resource limits, which implies that future progress benefits from (a) more scalable grounders, (b) tighter lifted reasoning methods, and (c) evaluation protocols that separate grounding bottlenecks from search bottlenecks. ⁸⁴

Understanding and exploiting hierarchy-induced structure.

Formal results show that hierarchy-induced cycles can break completeness of otherwise standard algorithms (A*), motivating HTN-specific cycle handling, dominance pruning, and admissible/consistent heuristic design aligned with HTN semantics. ⁸⁵

Bringing uncertainty beyond nondeterminism into solver ecosystems.

While nondeterministic and probabilistic HTN formalisms and complexity analyses exist (ICAPS 2021; IJCAI 2024; KR 2025), solver ecosystems and standardized benchmark tracks for probabilistic/partially observable HTN remain comparatively underdeveloped in this source set. ⁸⁶

Temporal HTN standardization and evaluation.

HDDL 2.1 proposals indicate demand for temporal/numeric constructs, but the source set centers on proposals, parsers, and benchmark introductions rather than mature, widely adopted competition tracks or standardized semantics across multiple planners. ³⁸

Modeling assistance and validation as core infrastructure.

Both modeling assistance (domain correction) and tooling for real-time HDDL validation point to a “planning lifecycle” view: correctness and maintainability of hierarchical models are research problems, not only solving problems. ⁸⁷

Neuro-symbolic integration with measurable guarantees.

LLM-in-the-loop HTN planning (ChatHTN) claims soundness under interleaving symbolic and approximate

decompositions, and L2HP-style model generation reports significant validity gaps for hierarchical models. This suggests research space in (a) constrained generation with formal validators, (b) decomposition synthesis under HTN semantics, and (c) benchmarks that measure hierarchical correctness distinct from classical correctness. ⁸⁸

Recommended next steps for a researcher entering HTN post-2020

1) Establish a baseline experimental stack anchored in IPC HTN benchmarks and HDDL parsing/validation. IPC 2020 and IPC 2023 provide domain suites and public repositories and document the infrastructure (HDDL, verifier expectations, benchmark composition). ⁸⁹

2) Select one solver family as an implementation substrate: * heuristic progression search systems (PANDA-family tooling and associated papers), ⁹⁰ * translation-based systems plus verification (TOAD-style workflows; AAAI 2022 translation improvements), ⁹¹ * symbolic/SAT/MaxSAT optimal solvers for TOHTN. ⁹²

3) Treat verification, repair, and model correction as part of the research toolbox rather than separate “afterthought” topics, because post-2020 literature shows them as central to correctness and to translation/approximation pipelines. ⁹³

4) For learning/hybrid research, prefer tool-supported pipelines and publish artifacts: * Use HDDLGym if the objective involves RL interaction and multi-agent extensions within HDDL-derived tasks. ²⁶ * If using LLMs for hierarchical modeling, incorporate strict validators and report hierarchical validity separately from parsing success, consistent with the failure profiles reported in L2HP/PlanBench-style evaluations. ⁹⁴

5) For reproducibility, containerize and document dependencies; explicitly declare any proprietary solver requirements (e.g., CPLEX), and when possible release citable experiment artifacts (e.g., Zenodo experimental setups). This aligns with IPC container practices and observed limitations (configurations requiring proprietary solvers cannot always be redistributed). ⁹⁵

¹ ² ¹⁶ ⁸⁹ <https://ipc2020.hierarchical-task.net/publications/IPC2020Booklet.pdf>
<https://ipc2020.hierarchical-task.net/publications/IPC2020Booklet.pdf>

³ ⁸ ⁴¹ ⁴³ ⁴⁵ ⁵² ⁵⁷ ⁹⁵ <https://ipc2023-htn.github.io/>
<https://ipc2023-htn.github.io/>

⁴ ⁹ <https://cdn.aaai.org/ojs/21203/21203-13-25216-1-2-20220628.pdf>
<https://cdn.aaai.org/ojs/21203/21203-13-25216-1-2-20220628.pdf>

⁵ ¹⁹ ⁶³ ⁶⁸ ⁹³ <https://bercher.net/publications/2022/Hoeller2022VerificationViaCompilation.pdf>
<https://bercher.net/publications/2022/Hoeller2022VerificationViaCompilation.pdf>

⁶ ¹⁷ ³⁵ ⁵⁹ ⁶² ⁸⁶ <https://ojs.aaai.org/index.php/ICAPS/article/view/15949>
<https://ojs.aaai.org/index.php/ICAPS/article/view/15949>

⁷ ²⁶ ²⁷ ³⁹ ⁶⁷ <https://arxiv.org/pdf/2505.22597>
<https://arxiv.org/pdf/2505.22597>

10 74 91 <https://github.com/toad-planner-dev/ToadPlanningSystem>
<https://github.com/toad-planner-dev/ToadPlanningSystem>

11 53 92 <https://cdn.aaai.org/ojs/17396/17396-13-20890-1-2-20210518.pdf>
<https://cdn.aaai.org/ojs/17396/17396-13-20890-1-2-20210518.pdf>

12 58 <https://arxiv.org/pdf/2411.02035>
<https://arxiv.org/pdf/2411.02035>

13 33 90 <https://fai.cs.uni-saarland.de/hoeller/papers/2021-Hoeller-ICAPS-LD.pdf>
<https://fai.cs.uni-saarland.de/hoeller/papers/2021-Hoeller-ICAPS-LD.pdf>

14 34 <https://www.inf.ufrgs.br/~agpereira/putrichicaps2025.pdf>
<https://www.inf.ufrgs.br/~agpereira/putrichicaps2025.pdf>

15 49 85 <https://yousefi.ai/static/papers/Yousefi2025Incompleteness.pdf>
<https://yousefi.ai/static/papers/Yousefi2025Incompleteness.pdf>

18 36 60 <https://www.ijcai.org/proceedings/2024/751>
<https://www.ijcai.org/proceedings/2024/751>

20 <https://ojs.aaai.org/index.php/ICAPS/article/view/36102>
<https://ojs.aaai.org/index.php/ICAPS/article/view/36102>

21 <https://ojs.aaai.org/index.php/AAAI/article/view/30000>
<https://ojs.aaai.org/index.php/AAAI/article/view/30000>

22 <https://www.ijcai.org/proceedings/2025/0489.pdf>
<https://www.ijcai.org/proceedings/2025/0489.pdf>

23 61 69 87 <https://bercher.net/publications/2024/Lin2024HTNModelFixing.pdf>
<https://bercher.net/publications/2024/Lin2024HTNModelFixing.pdf>

24 54 <https://arxiv.org/pdf/2504.16209>
<https://arxiv.org/pdf/2504.16209>

25 <https://www.cs.umd.edu/~nau/papers/goldman2024comparative.pdf>
<https://www.cs.umd.edu/~nau/papers/goldman2024comparative.pdf>

28 29 <https://arxiv.org/pdf/2306.08359>
<https://arxiv.org/pdf/2306.08359>

30 <https://arxiv.org/abs/2511.18165>
<https://arxiv.org/abs/2511.18165>

31 94 https://icaps25.icaps-conference.org/files/HPlan/HPlan2025_paper_3.pdf
https://icaps25.icaps-conference.org/files/HPlan/HPlan2025_paper_3.pdf

32 65 88 <https://proceedings.mlr.press/v288/munoz-avila25a.html>
<https://proceedings.mlr.press/v288/munoz-avila25a.html>

37 51 <https://proceedings.kr.org/2025/84/kr2025-0084-yousefi-et-al.pdf>
<https://proceedings.kr.org/2025/84/kr2025-0084-yousefi-et-al.pdf>

38 <https://arxiv.org/pdf/2306.07353>
<https://arxiv.org/pdf/2306.07353>

40 48 50 66 84 <https://github.com/panda-planner-dev/ipc2020-domains>
<https://github.com/panda-planner-dev/ipc2020-domains>

42 44 <https://ai.dmi.unibas.ch/papers/taitler-et-al-aimag2024.pdf>
<https://ai.dmi.unibas.ch/papers/taitler-et-al-aimag2024.pdf>

46 55 <https://zenodo.org/records/11172885>
<https://zenodo.org/records/11172885>

47 <https://bercher.net/publications/2025/Yousefi2025HDDLVSPlugin.pdf>
<https://bercher.net/publications/2025/Yousefi2025HDDLVSPlugin.pdf>

56 73 <https://github.com/panda-planner-dev/pandaPIprRepairVerify>
<https://github.com/panda-planner-dev/pandaPIprRepairVerify>

64 <https://ojs.aaai.org/index.php/ICAPS/article/view/36124>
<https://ojs.aaai.org/index.php/ICAPS/article/view/36124>

70 <https://github.com/shop-planner/shop3>
<https://github.com/shop-planner/shop3>

71 <https://github.com/panda-planner-dev/pandaPIengine>
<https://github.com/panda-planner-dev/pandaPIengine>

72 <https://github.com/panda-planner-dev/pandaPIparser>
<https://github.com/panda-planner-dev/pandaPIparser>

75 <https://github.com/HDDLGym/HDDLGym>
<https://github.com/HDDLGym/HDDLGym>

76 <https://github.com/koala-planner/HDDL-Parser>
<https://github.com/koala-planner/HDDL-Parser>

77 <https://github.com/YashBansod/IPyHOP>
<https://github.com/YashBansod/IPyHOP>

78 <https://github.com/virajparimi/thtn>
<https://github.com/virajparimi/thtn>

79 <https://github.com/hhhhmmmmm02/ChatHTN>
<https://github.com/hhhhmmmmm02/ChatHTN>

80 <https://www.sciencedirect.com/science/article/pii/S235271102400150X>
<https://www.sciencedirect.com/science/article/pii/S235271102400150X>

81 <https://github.com/PlanX-Universe/sh>
<https://github.com/PlanX-Universe/sh>

82 <https://pdfs.semanticscholar.org/e238/145a6c6fe1956bca943ee77c747e7478d7f6.pdf>
<https://pdfs.semanticscholar.org/e238/145a6c6fe1956bca943ee77c747e7478d7f6.pdf>

83 <https://dl.acm.org/doi/fullHtml/10.1145/3623809.3623877>
<https://dl.acm.org/doi/fullHtml/10.1145/3623809.3623877>