Position: Make Planning Research Rigorous Again!

A Comprehensive Guide to Rigorous Research in LLM-Based Planning

Michael Katz · Harsha Kokel · Christian Muise · Shirin Sohrabi · Sarath Sreedharan

Core claim: LLM-based planning is promising, but progress requires the vocabulary, tools, benchmarks, validators, and evaluation protocols developed by the automated planning community.
Goal: avoid known pitfalls and make results reproducible, diagnostic, and comparable

Position in one sentence

LLM-based planning should not abandon rigor. It should inherit it.

Planning methodology helps separate:
- language understanding
- model construction
- search and optimization
- execution and tool use

so progress is clearer and more trustworthy.

Why this matters now

- LLMs are increasingly used for planning, acting, and tool use.
- Planning has 60+ years of lessons about representations, algorithms, and evaluation.
- Current evaluations often repeat known pitfalls: contamination, fuzzy semantics, unreliable validation, and mismatched baselines.

→ Goal: reproducible, diagnostic, comparable results.

What planning contributes

Formal problems
- Explicit assumptions about states, actions, goals, observability, dynamics, and solutions.

Shared languages and tools
- PDDL/RDDL, domains, generators, parsers, planners, validators.

Evaluation culture
- IPC-style protocols: common resources, scalable difficulty, strong baselines.

Pitfall 1: unclear problem definitions

Problem
- A multi-step task is not automatically a planning problem.
- Problem's computational complexity matters.
- A planner's computational complexity matters
- Soundness, completeness, and optimality are not interchangeable.

What to do
✓ Clearly scope and define: Define states, actions, goals, assumptions, and a solution.
✓ Match baselines by computational problem and guarantees.
✓ Validate outputs independently.

Pitfall 2: benchmark issues

Problem
- Public benchmarks & scraped datasets may overlap with training data.
- Random splits can include isomorphic or structurally similar instances.
- Harder benchmarks are not better if correctness becomes subjective.

What to do
✓ Understand the domains you experiment with
✓ Use fresh generator-backed instances and structure-aware splits.
✓ Scale difficulty by plan length, objects, or domain parameters.
✓ Use auditable validation.

Pitfall 3: opaque evaluation

Problem
- Success rate alone hides quality, effort, and cost.
- Planner names do not define configurations or guarantees.
- Unsound outputs can be falsely counted as solved.

What to do
✓ Report quality, runtime, memory, search effort, calls, retries, and repairs.
✓ Specify algorithm, heuristic, configuration, and problem type.
✓ Validate independently; compare with identical resources.

Practical rigor checklist

1. Define the planning problem and assumptions.
2. State formal guarantees and computational complexity when possible.
3. Use strong, relevant baselines with matching guarantees.
4. Validate solutions with an independent sound validator.
5. Report full budgets: time, memory, LLM calls, retries, tools.
6. Evaluate across diverse domains with scaling and novelty controls.

Rigorous LLM-planning pipeline

Make claims precise → evaluate fairly → expose failure modes → support reproducible progress

Takeaway: LLM-based planning should inherit the rigor of automated planning.

Contact: michael.katz1@ibm.com