When does the planner matter?

Comparing POMDP solvers for active search on graphs

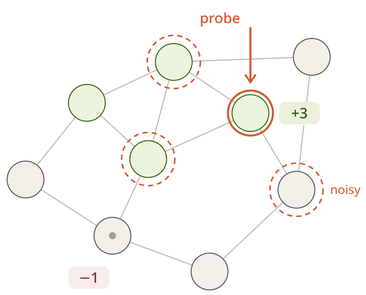
The same belief, three solvers, and why the belief decides

Andrei Mosescu
TU Delft, CSE3000 Research Project
Supervisors: Frans Oliehoek, Andreea Minculescu

1. Active search on graphs as a POMDP

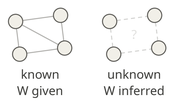
Task. Sequentially probe nodes to find as many positive-labeled nodes as possible within a fixed query budget. On a graph, connected nodes tend to share labels, so a single probe also reshapes what is believed about its neighbours, the structure that makes planning worthwhile in principle.

positive · negative / unqueried · probe · neighbour revealed



Stop action
agent ends the episode
+20 all positives found
−10 stops too early

Topology variants



Why a POMDP? The agent never sees the true labels: each probe returns only a noisy version of one node's label. Every decision is therefore made from a belief, a distribution over possible label (and, under unknown topology, graph) configurations, revised after each observation. The action set also includes Stop, so the agent decides not just which node to probe but when to quit.

Two Variants. Known topology: the graph is given and only labels are uncertain.
Unknown topology: the graph must be inferred too, and each probe additionally returns noisy local edges. This is a much larger, higher-dimensional observation that, as we will see, is far harder for the belief to track.

2. Research question and method

Prior work. Garnett et al. showed that less-myopic planning can beat myopic planning by a wide margin; Jiang et al. gave efficient budget-aware nonmyopic policies; Wang et al. brought active search to graphs. The shared lesson concerns planning depth: it pays off when the posterior carries useful structure.
Research question. Those results all vary how deeply the agent plans. Holding the POMDP formulation and the belief representation fixed, does the choice of online solver itself change performance and under which conditions?

Method. POMCP, DESPOT and POMCPOW are compared on one environment under tight controls: identical graph instances, query budget, particle count, wall-clock planning budget, and the same bootstrap particle filter. The three span the main design axes — rollouts (POMCP), bounded scenario search (DESPOT), and observation widening (POMCPOW) — so any architecture-driven difference has room to surface. Solver, MH rejuvenation, graph size and observation noise are varied so any real solver difference has every chance to surface, in easy, noisy, and topology-uncertain regimes. Every configuration uses 30 graph seeds, with recall reported at 95% confidence intervals.

Independent variables

Variable	Values	Applies to
Solver	POMCP, DESPOT	Both variants
	POMCPOW	Unknown topology
MH rejuvenation	on, off	Both variants
Graph size n	10, 20, 30, 50, 100, 150	Known topology
	20, 30, 40	Unknown topology
Obs. noise ε_{obs}	0.0, 0.1, 0.2	Known topology
Topo. noise $\varepsilon_{\text{topo}}$	0.0, 0.1, 0.2	Unknown topology

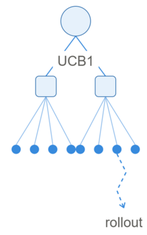
Fixed: K = 10,000 particles, 30 seeds, γ = 0.9, 1.0 s/step, budget B = n.

Prior work moves along planning *depth* (Garnett, Jiang); this work moves along *which solver* at fixed depth. Orthogonal questions — so a null result here complements, rather than contradicts, theirs.

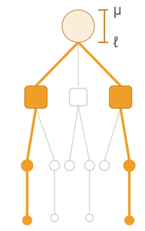
3. Three solvers, one shared belief

Three architecturally different searches; one identical starting belief (b). If they query differently, it must come from the tree; if they query the same, the belief is what governs.

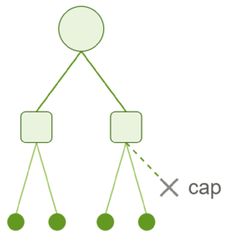
POMCP.POMCP starts each simulation by picking a particle from the belief. It walks down a tree of action–observation histories, choosing actions with UCB to balance exploration/exploatation trade-off. When it hits a leaf, it does a random rollout to estimate the value.



DESPOT. samples a fixed set of scenarios at the start of each step, so its estimates have lower variance — the randomness lives in those scenarios, not in a rollout. Its tree keeps all actions but only the observations the scenarios actually produce. It searches using an upper and a lower bound, and the gap between them tells it where to expand next. to guide expansion.



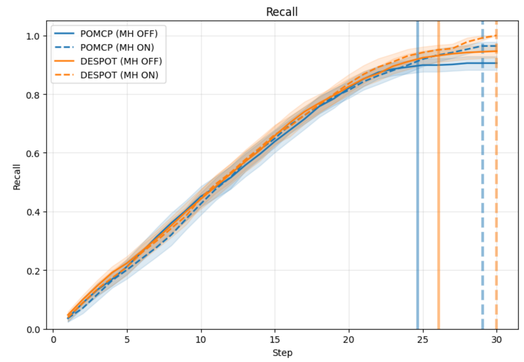
POMCPOW. Extends POMCP with progressive widening, which caps observation branching when the observation space is large. It uses weighted particles inside the tree. I added it for the unknown case, where there are many more possible observations.



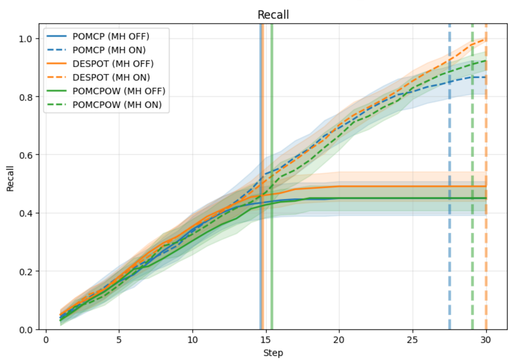
Common belief mechanism. All three plan from the same belief produced by the bootstrap particle filter: a set of sampled possible worlds. After each real query, particles are reweighted by how well they explain the new observation, then resampled.
MH rejuvenation. Metropolis–Hastings is added as a resample-move step to restore particle diversity after collapse. It proposes small local changes, mainly flipping the label of one unqueried node. The acceptance rule uses the posterior score, but the prior is a heuristic rather than the exact graph generator: it prefers fewer positives and rewards neighbouring nodes that share a label, matching the sparsity and homophily of graph active search.

4. Result 1: solvers query identically

Carry-forward Recall — Known Topology, obs_noise=0.1, 30 nodes



Carry-forward Recall — Unknown Topology, topo_noise=0.1, 30 nodes



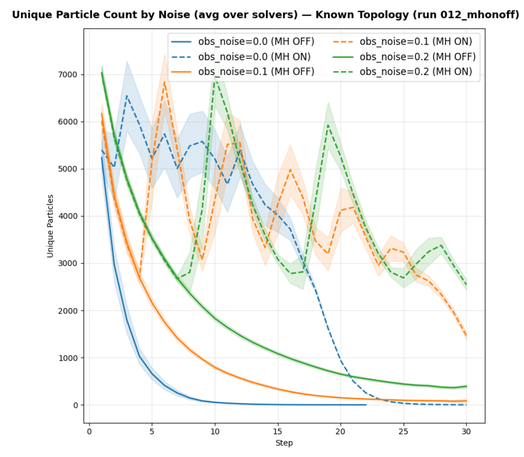
Per-step recall is one curve; only the stop step differs. Carry-forward recall, $n = 30$, $\varepsilon_{\text{obs}} = 0.1$ (left: known, right: unknown topology). Curves overlap until episodes begin to stop; vertical lines mark mean stop steps. *Colours = solver; solid = MH off, dashed = MH on.*

What you are seeing. As you can see, the solvers query at the same per-step rate. There's no real difference in which nodes they pick. What does differ is when they stop. DESPOT's bounds keep querying looking attractive, so it goes longer, while POMCP and POMCPOW use rollouts that turn pessimistic, so they stop sooner.

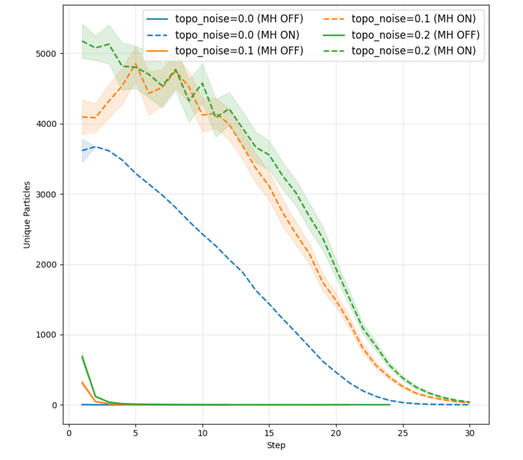
Unknown. On the unknown side, without Metropolis–Hastings, you can see the solvers quit far too early because of the collapsed belief.

5. Result 2: the bottleneck is belief collapse

Unique Particle Count by Noise (avg over solvers) — Known Topology (run 012_mhonoff)



Unique Particle Count by Noise (avg over solvers) — Unknown Topology (30 nodes)



The shared filter collapses — identically across solvers. Unique-particle count over time (left: unknown, right: known topology). Without MH the belief degenerates to a handful of worlds within a few steps. **Here colours = noise level, not solver.** Solid = MH off, dashed = MH on.

Collapse dominates. Without MH the filter loses almost all unique worlds within a few steps (faster under unknown topology). With no uncertainty left to represent, extra search depth has nothing to act on.

Noise paradox. More noise slows collapse (softer likelihoods keep more worlds alive), so recall rises with noise — but a slower-collapsing belief is not a more accurate one; the gain is bought with extra probes.

6. Interpretation

Belief collapse is the bottleneck, not the solver. Every solver plans over the particle belief, never the true graph and labels. Once that belief collapses, the unrepresented uncertainty is gone and all three search the same impoverished posterior, so their probe choices coincide.

What solver choice actually moves. With this bootstrap filter, the only consistent solver difference is *when* the agent stops. POMCP, DESPOT and POMCPOW hit positives at nearly the same per-query rate; they value continuing differently, which shifts the stopping point and, through carry-forward averaging, the episode-level recall.

Why MH does not fix it. Rejuvenation restores particle *diversity* but not belief *quality*: its proposal flips unqueried labels under a sparsity-and-homophily heuristic, not the true graph generator. So it keeps the agent probing longer without making any single probe smarter; diversity is not a faithful posterior.

7. Where performance is won

In this formulation, performance is won or lost in the belief representation, before the planner ever matters.

Directions for improvement.

1. **Collapse-resistant belief updates.** Better proposals, structured or Rao–Blackwellised representations, or learned priors; most urgent in unknown topology, where high-dimensional observations collapse the filter within a few steps. This is the factor that dominates every result.
2. **More structured graphs** Testing larger graphs and more structured ones, like scale-free or community graphs, might let solver differences finally show up.
3. **Stopping calibration.** Because the solver gaps are stopping artifacts, calibrating the Stop action is a cheap, immediate lever: it improves reward without touching probe selection.

References: Garnett et al., *Bayesian Optimal Active Search and Surveying*; Jiang et al., *Efficient Nonmyopic Active Search*; Silver and Veness, *Monte-Carlo Planning in Large POMDPs*; Ye et al., *DESPOT*; Sunberg and Kochenderfer, *POMCPOW*.