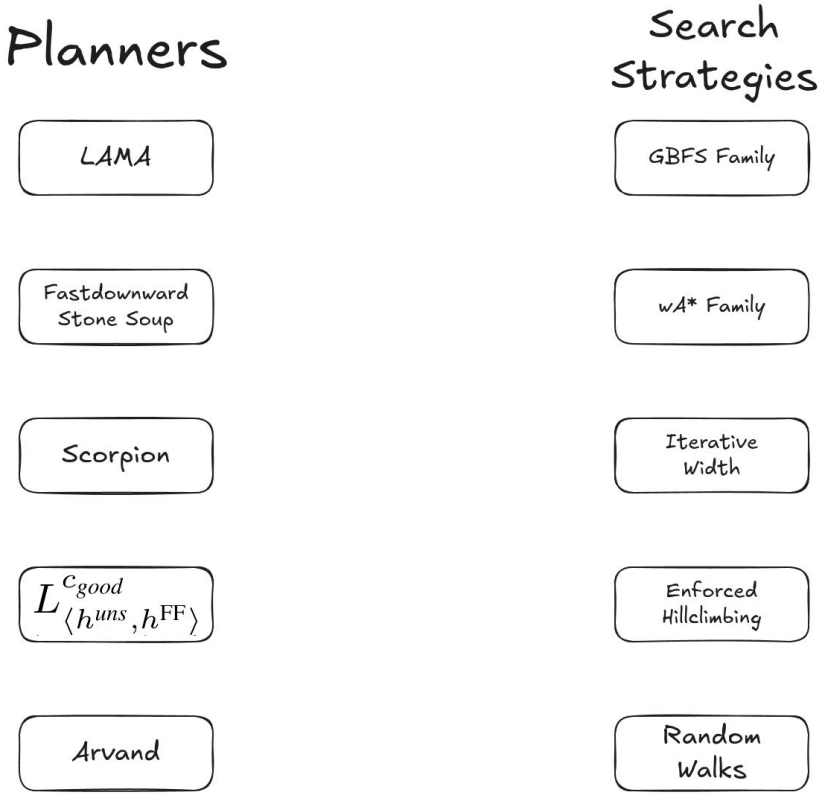
Triangle Search in Planning

Jordan Thayer[1], Sofia Lemons[2], Jendrik Seipp[1]
Linköping University, Sweden [1]
Accenture, Indianapolis, IN, USA [2]
jordan.thayer@liu.se, sofia@snlemons.com, jendrik.seipp@liu.se

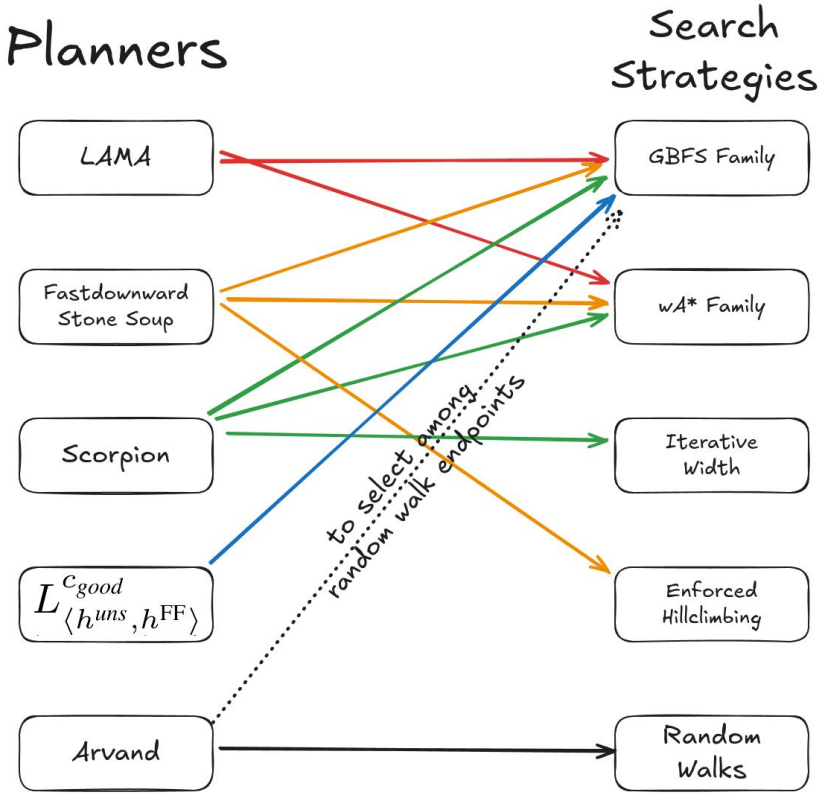
If the Building Catches Fire During the Talk...

- Triangle Search performs well in classic heuristic search benchmarks
- It also performs extremely well in planning
  - better than our usual base strategies
  - Not as well as algorithms with planning-specific enhancements
- This points to two opportunities:
  - Exporting planning-specific search enhancements to general search problems
  - Importing those enhancements to Triangle Search to improve planners

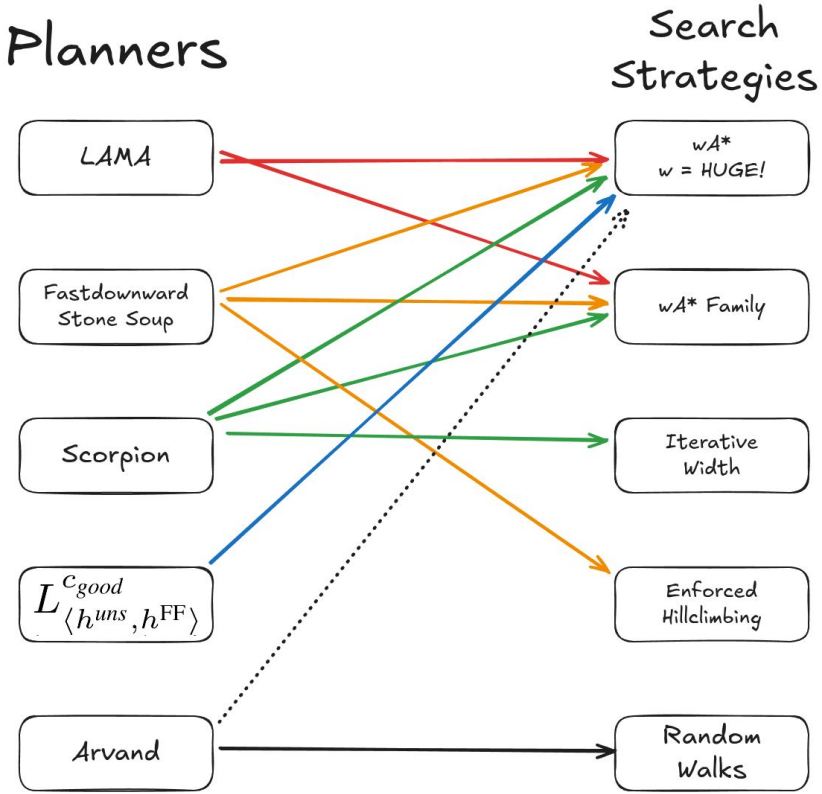
Planning Systems Use Search Under the Hood



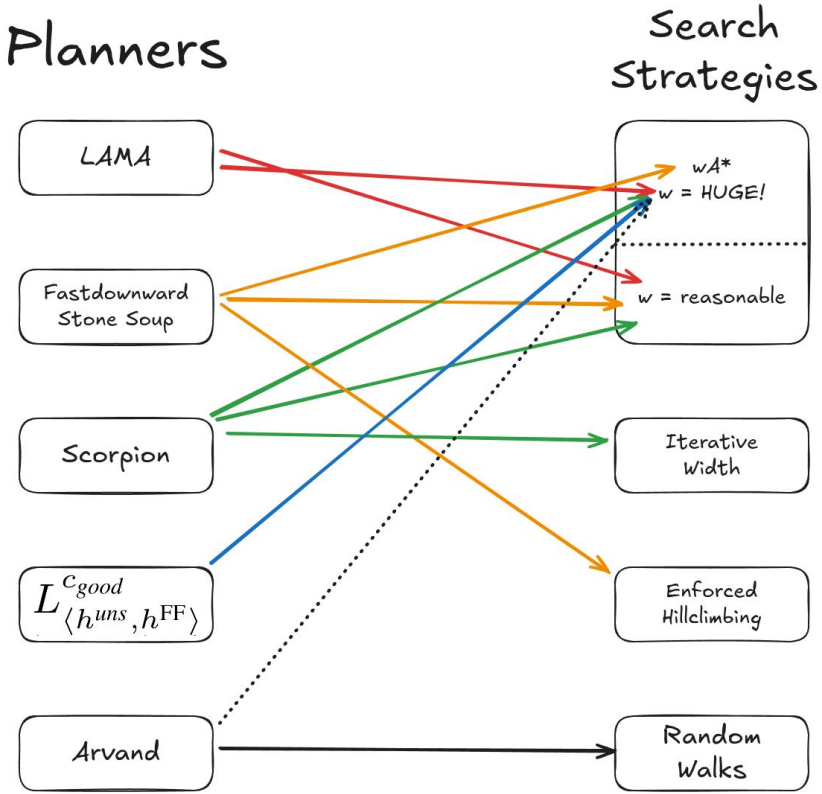
Planning Systems Use Search Under the Hood



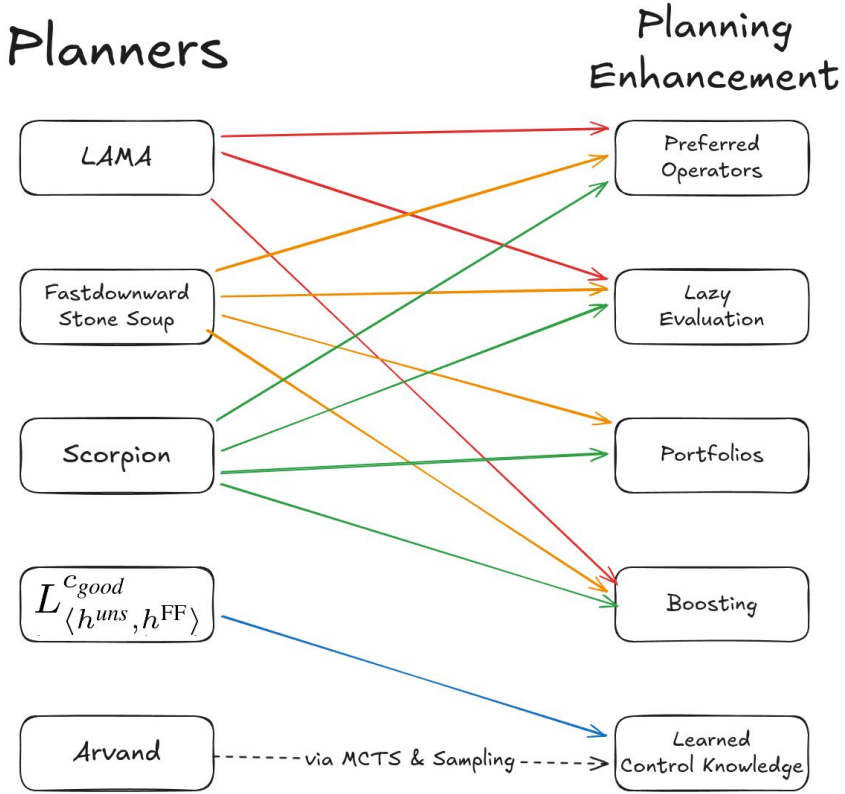
Planning Systems Use Search Under the Hood

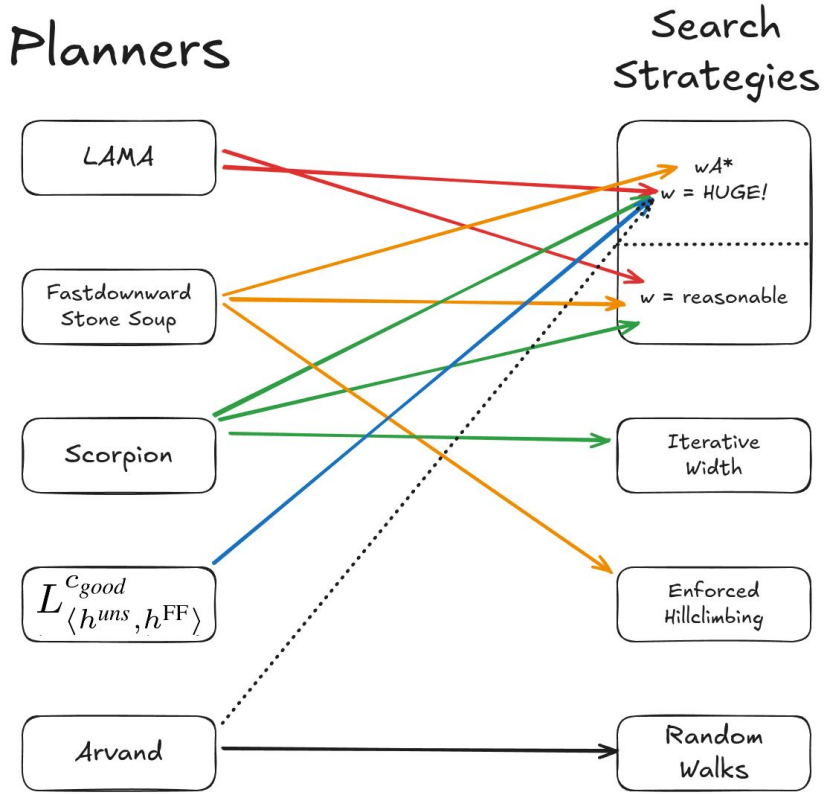


Planning Systems Use Search Under the Hood



Planning Progress is Uneven

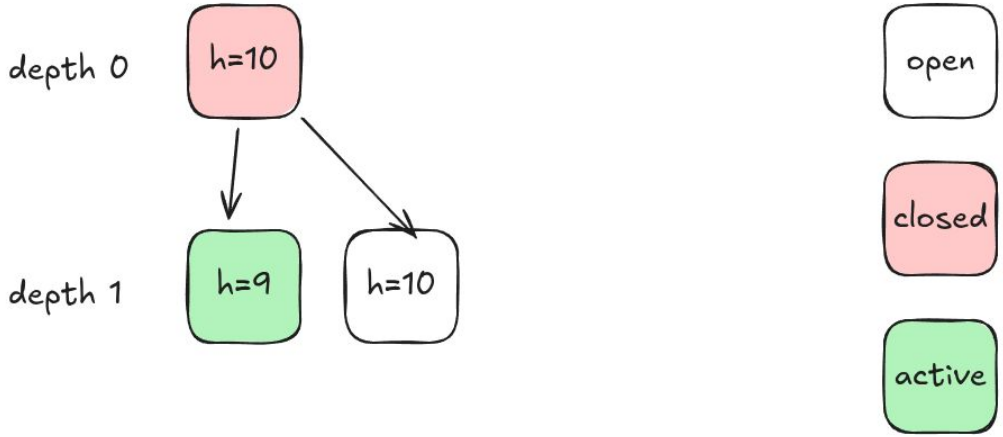




Triangle Search

Triangle search feels a lot like
Enforced hill climbing

We expand a node at each depth,
Starting with the shallowest

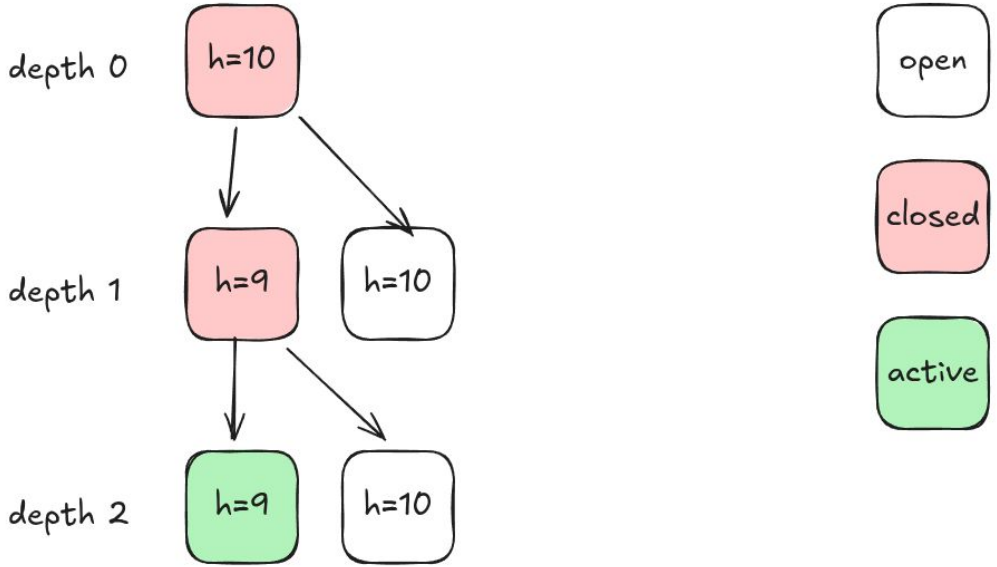


Triangle Search

Triangle search feels a lot like
Enforced hill climbing

We expand a node at each depth,
Starting with the shallowest

Nodes are only compared across
The same search depth



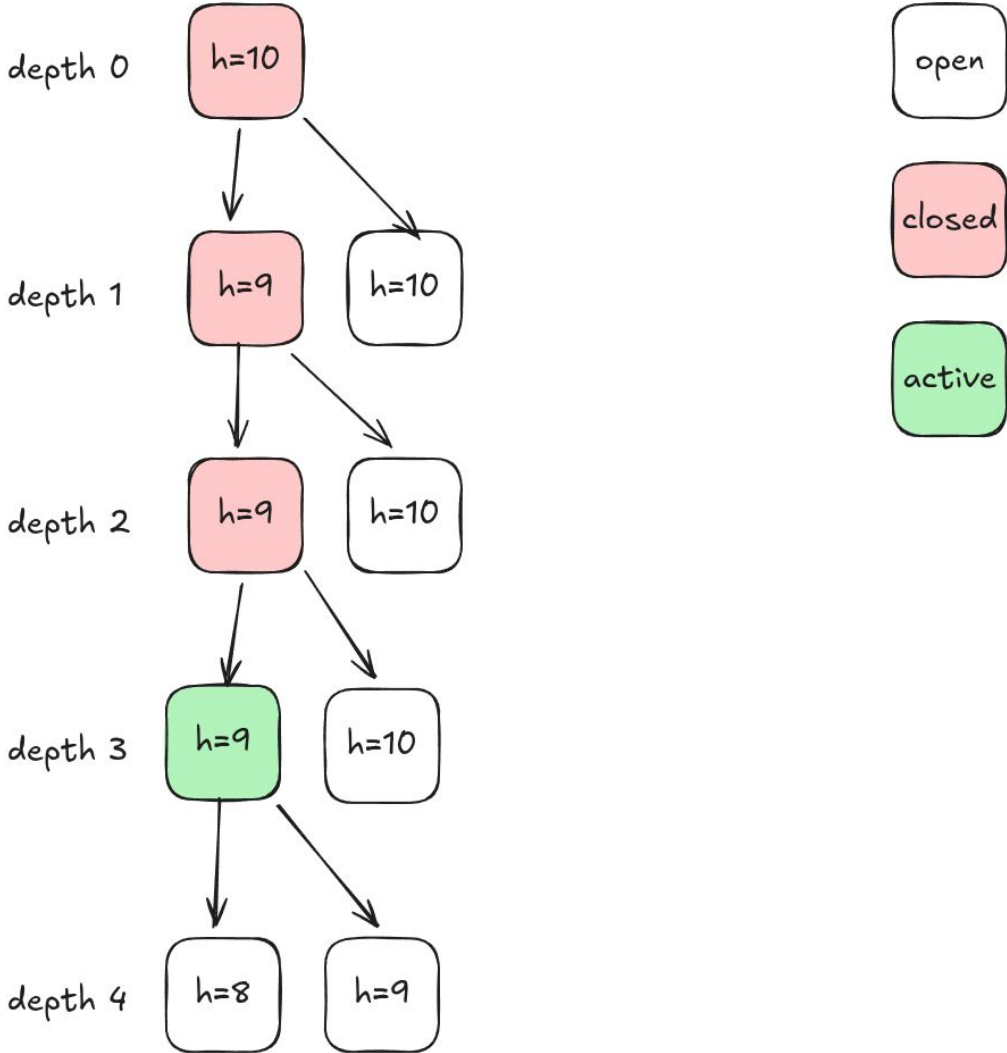
Triangle Search

Triangle search feels a lot like
Enforced hill climbing

We expand a node at each depth,
Starting with the shallowest

Nodes are only compared across
The same search depth

Unexpanded nodes stay in open



Triangle Search

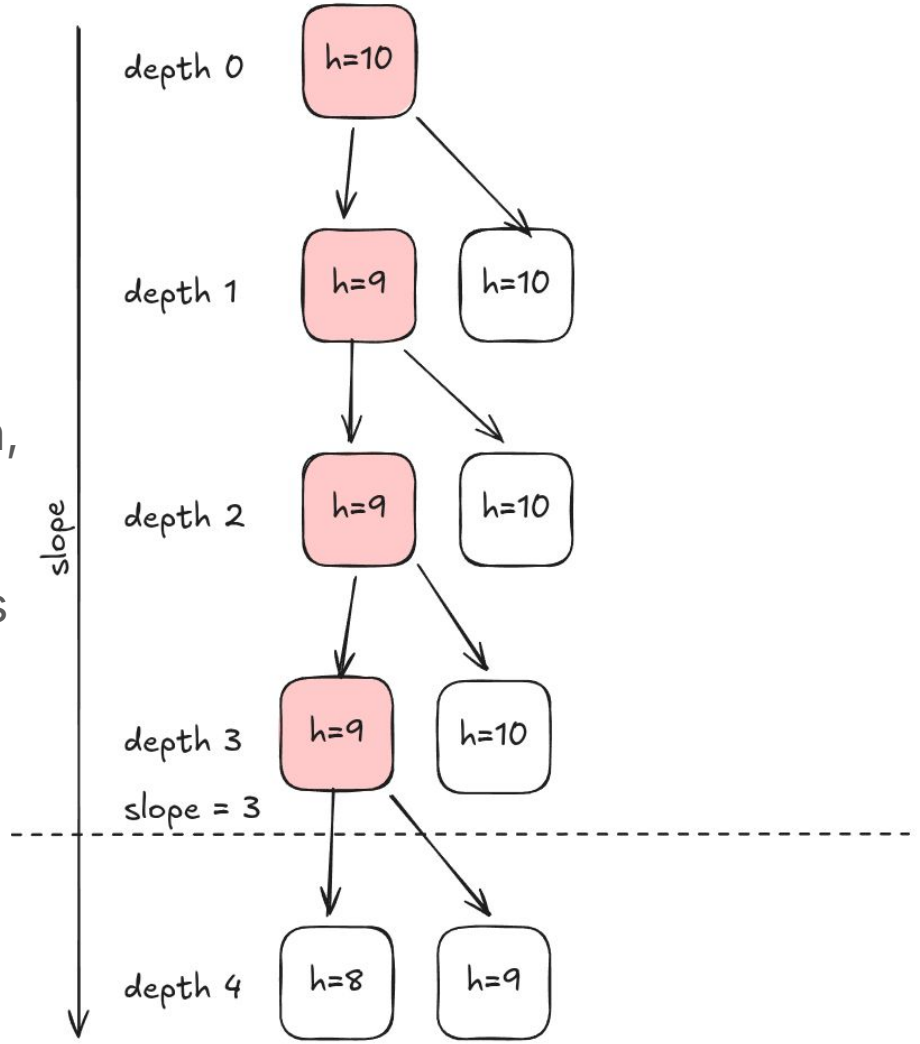
Triangle search feels a lot like
Enforced hill climbing

We expand a node at each depth,
Starting with the shallowest

Nodes are only compared across
The same search depth

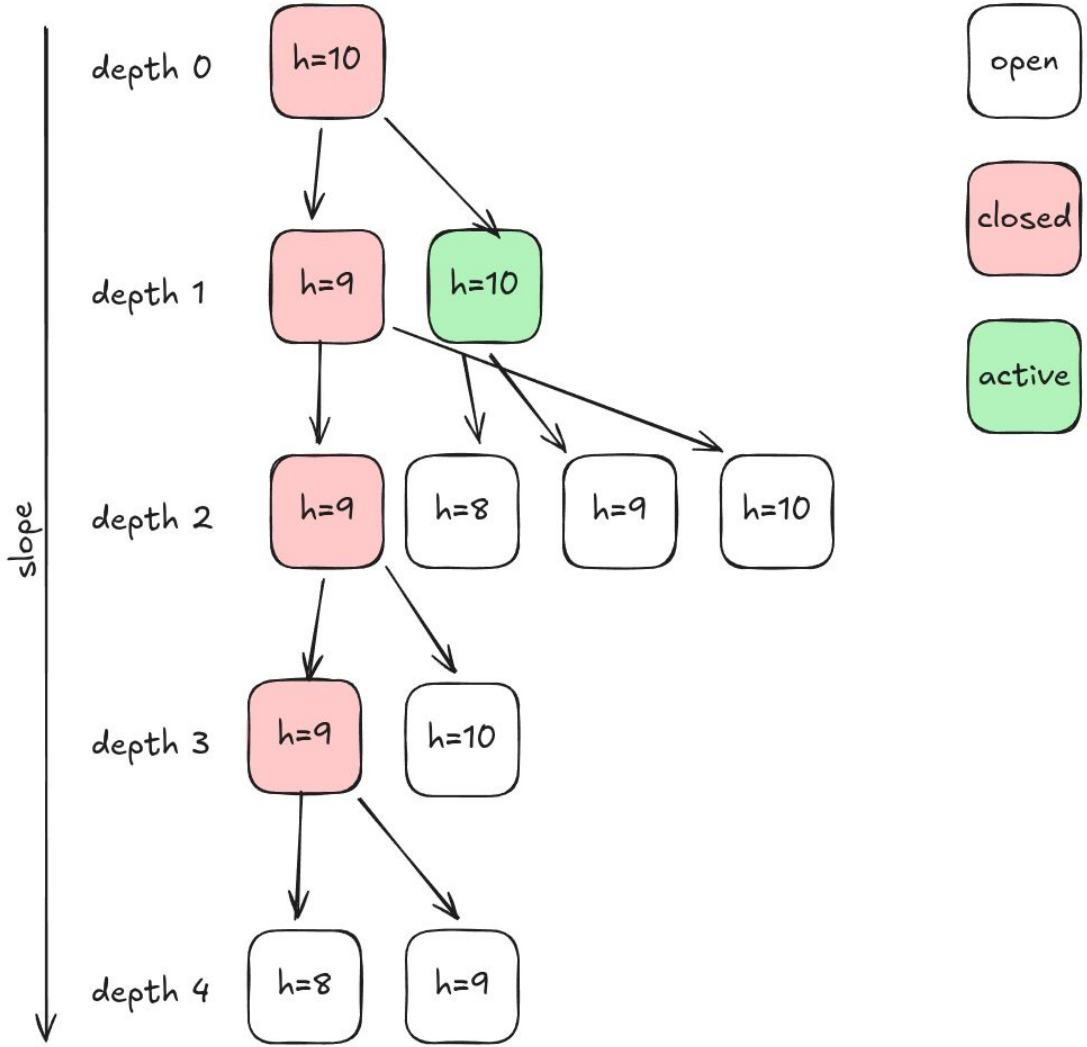
Unexpanded nodes stay in open

We run forward *SLOPE* steps,
Which is a parameter



Triangle Search

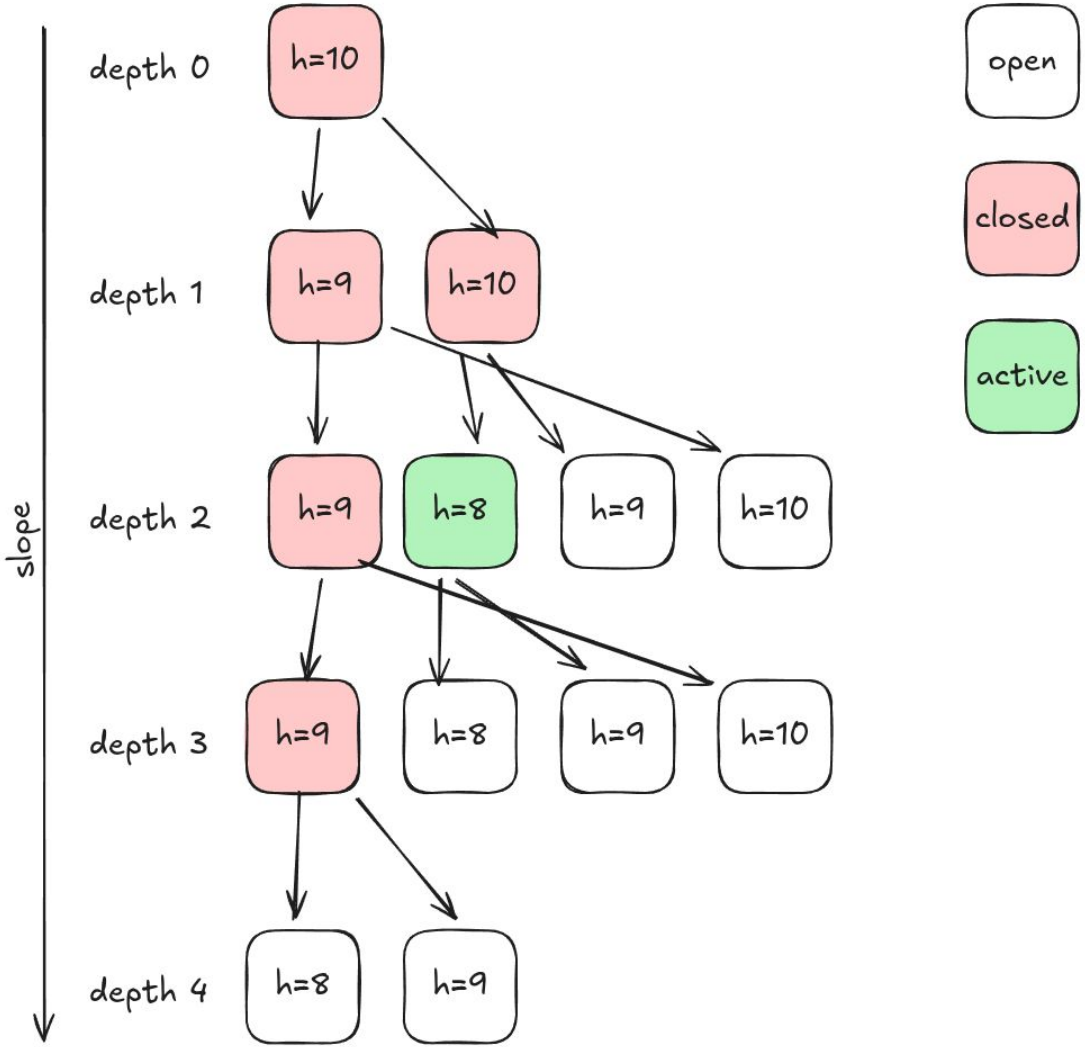
We restart from the shallowest depth with open nodes every iteration



Triangle Search

We restart from the shallowest depth with open nodes every iteration

We can prune shallow layers Where all nodes are closed

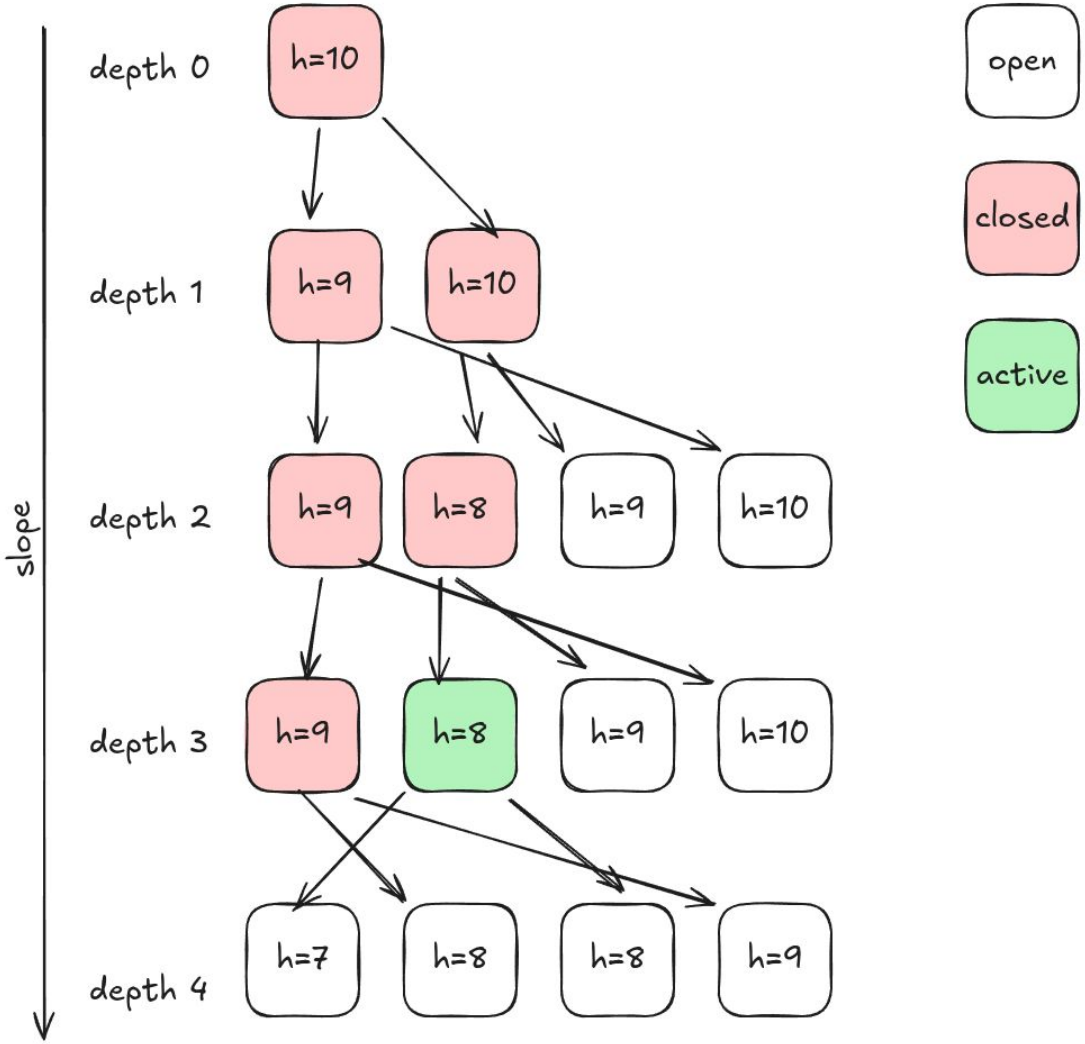


Triangle Search

We restart from the shallowest depth with open nodes every iteration

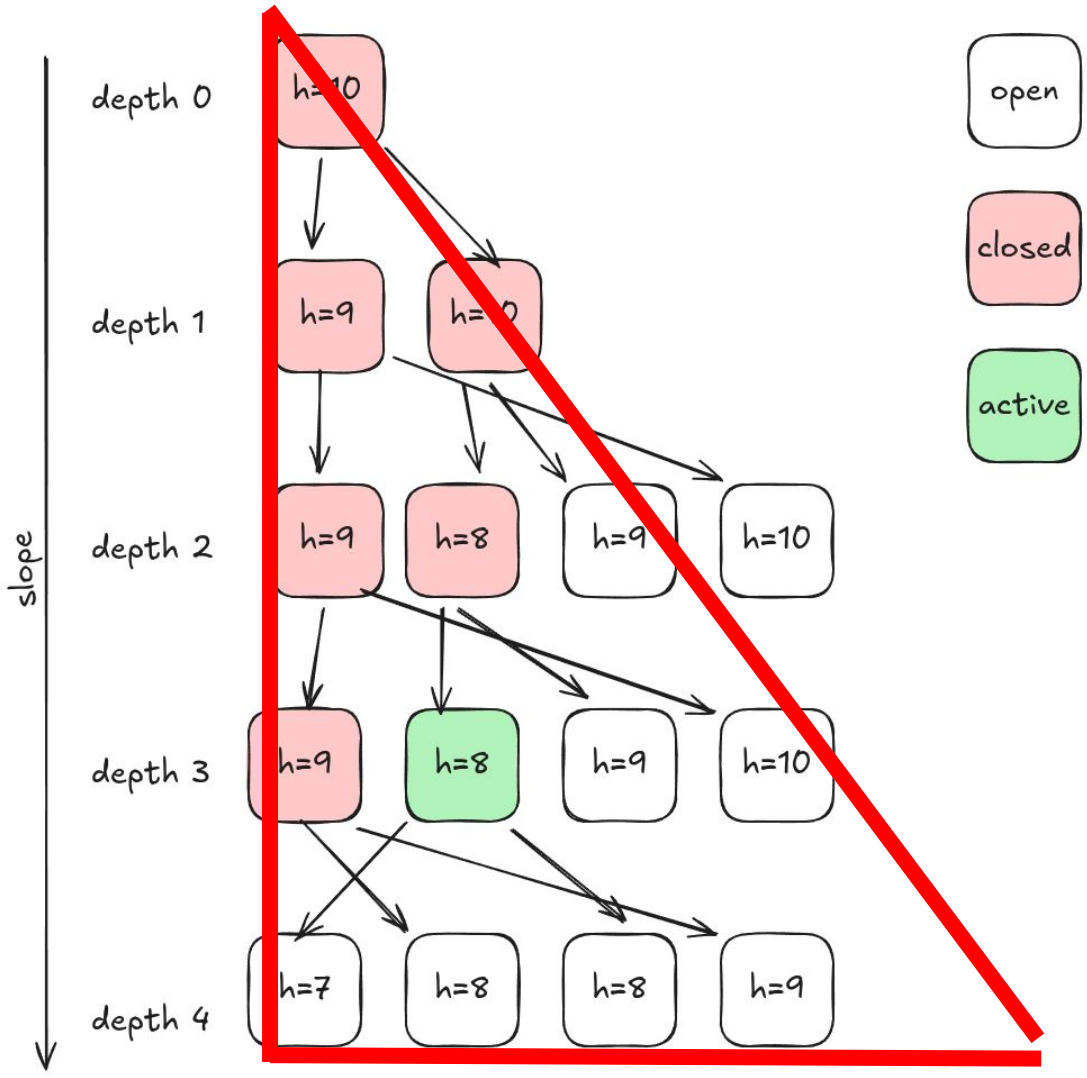
We can prune shallow layers Where all nodes are closed

Going shallow to deep gives us a chance to ‘compete’ with children of siblings at a shallower layer



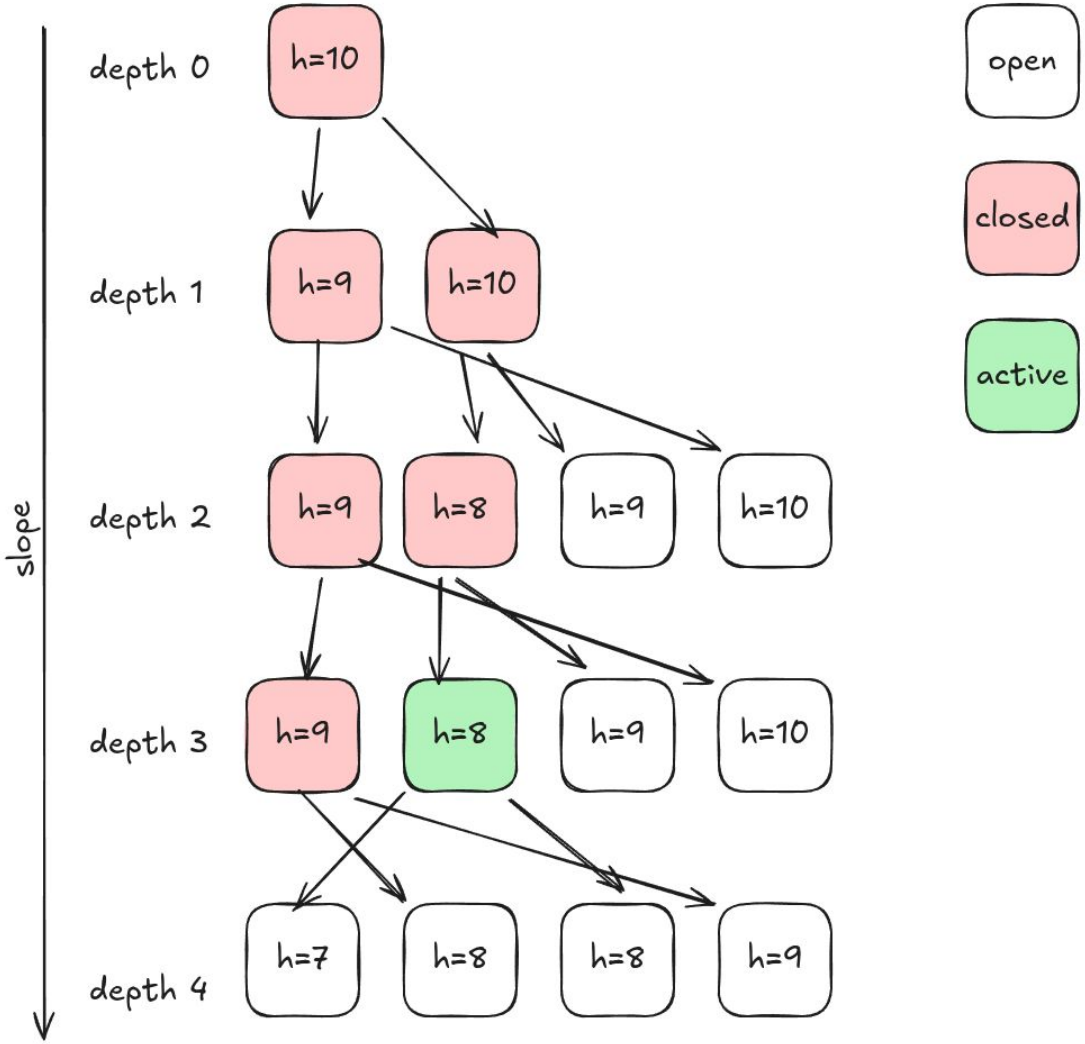
Triangle Search

Why Triangle?
Evocative of the search frontier



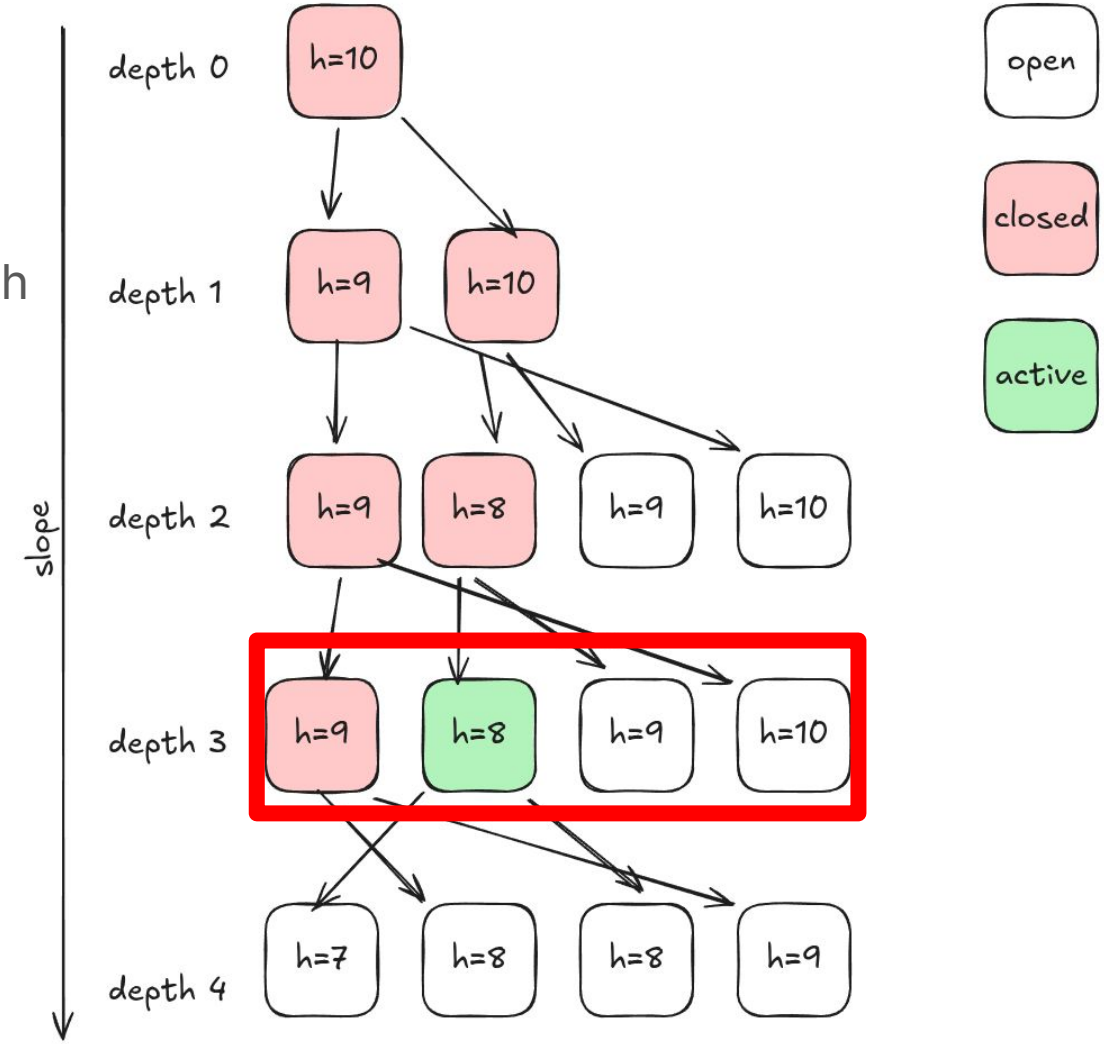
Triangle Search

I think ‘bead curtain’ is more accurate to behavior



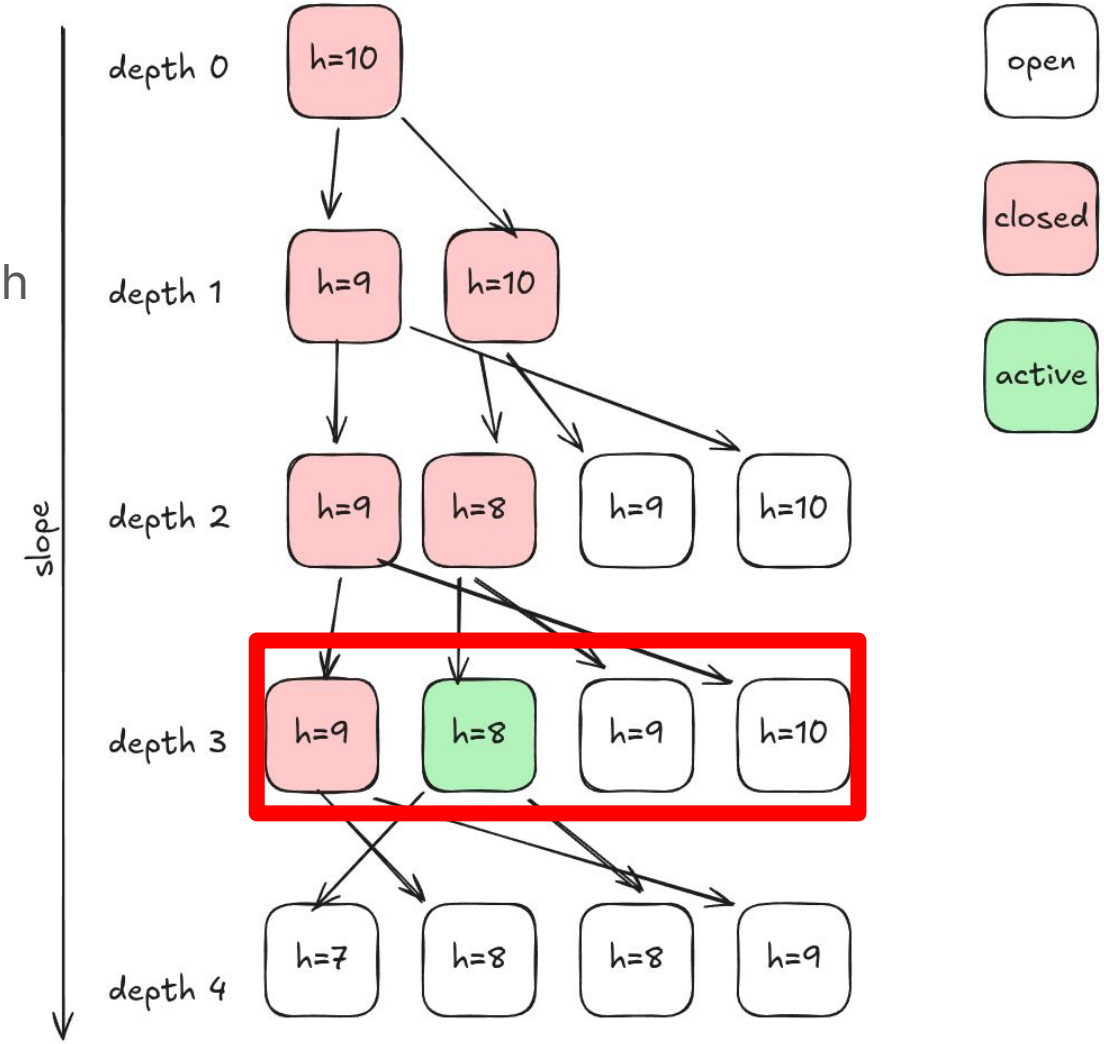
Triangle Search

- Nodes competing at fixed depth



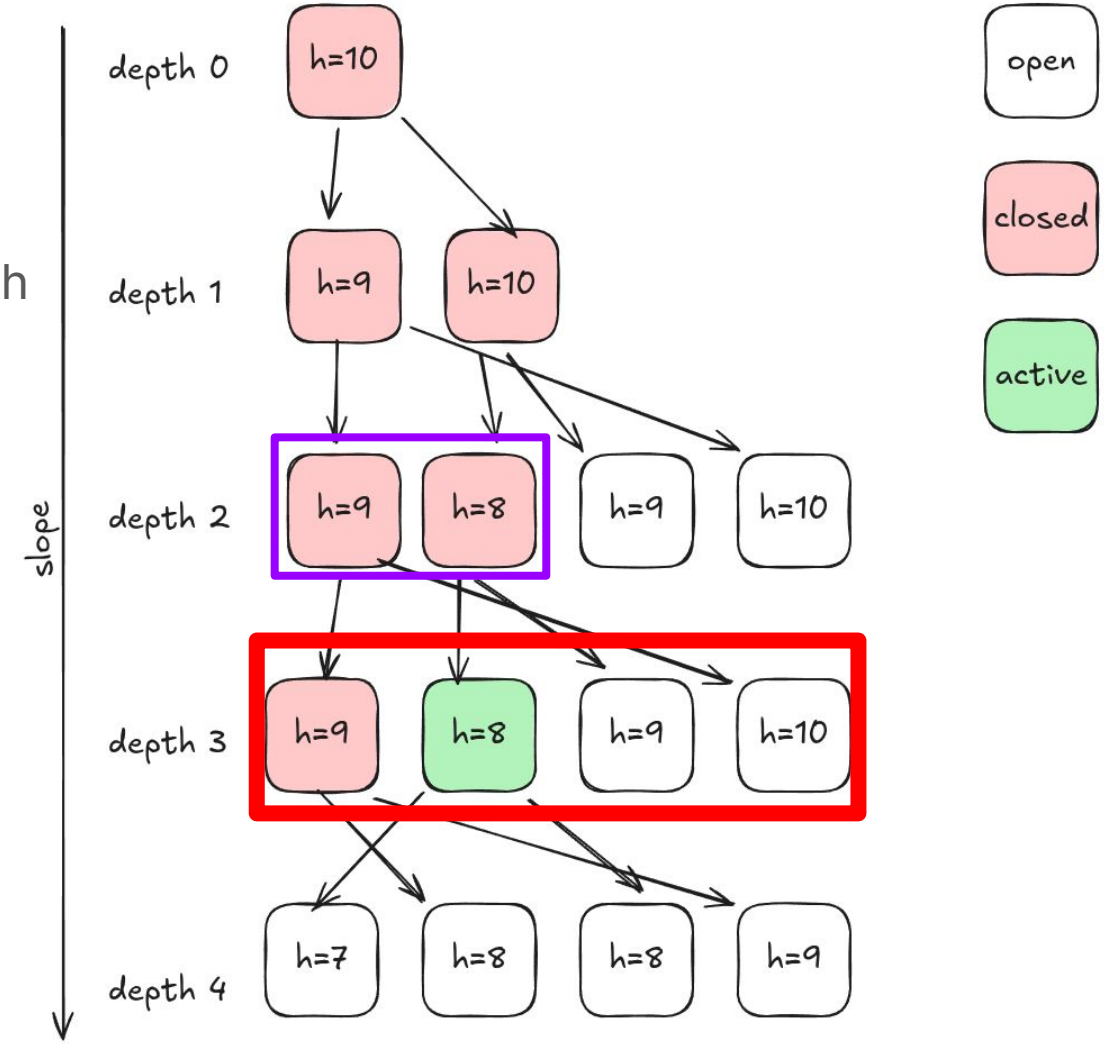
Triangle Search

- Nodes competing at fixed depth
  - A* Epsilon-ish
  - Beam Search-ish
  - Tree Search-ish



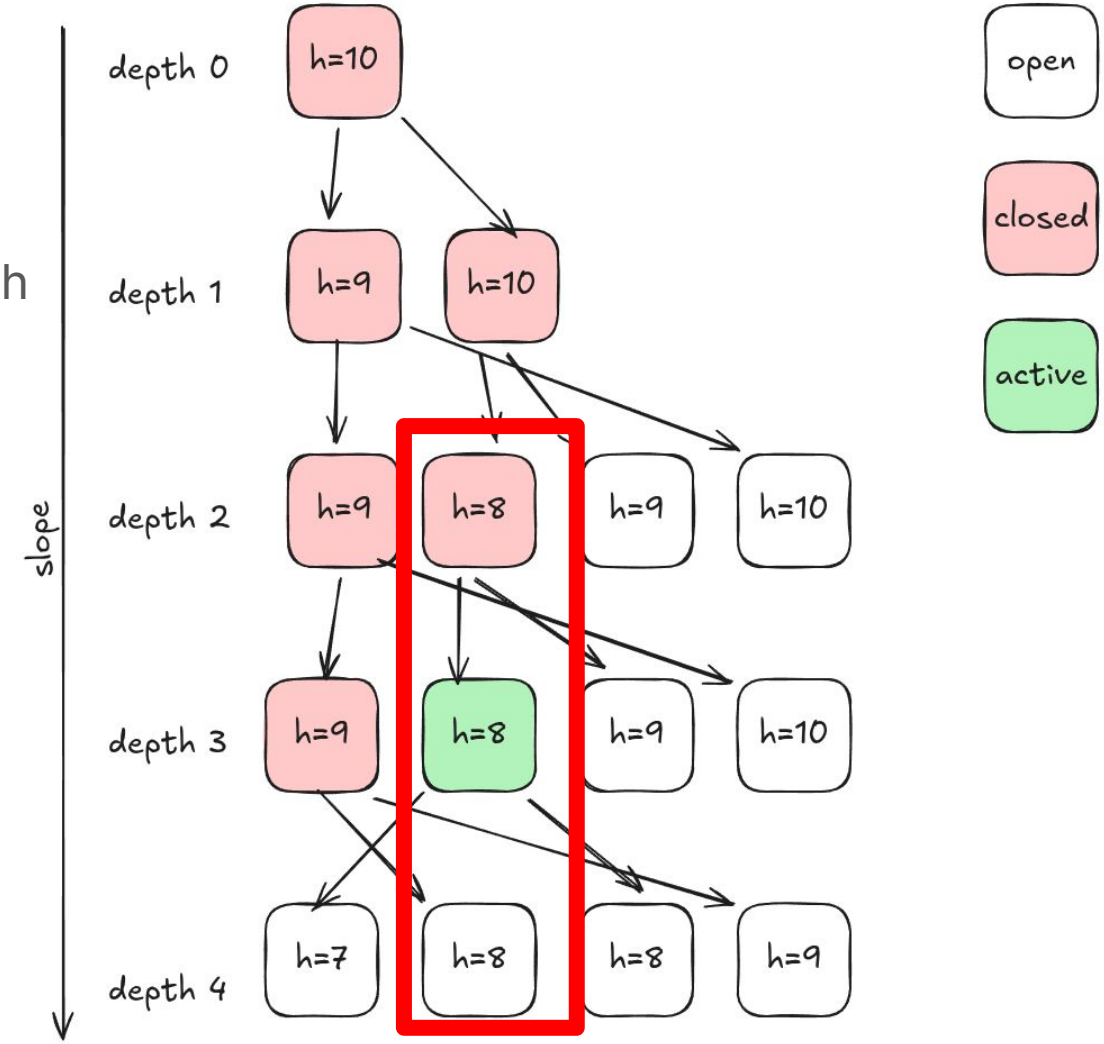
Triangle Search

- Nodes competing at fixed depth
  - A* Epsilon-ish
  - Beam Search-ish
  - Tree Search-ish*



Triangle Search

- Nodes competing at fixed depth
  - A* Epsilon-ish
  - Beam Search-ish
  - Tree Search-ish
- Depth Search Control
  - Window A*
  - d-fenestration



Triangle Search Works Pretty Good! - h_ff

Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
TRI	576	0.85	14.15	0.04
RECT	470	5.08	40.10	1.64
GBFS	458	2.49	36.71	0.05

Triangle Search Works Pretty Good! - h_ff

Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
LAMA	878	0.42	4.24	0.06
TRI	576	0.85	14.15	0.04
RECT	470	5.08	40.10	1.64
GBFS	458	2.49	36.71	0.05

Triangle Search Works Pretty Good! - h_ff

Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
LAMA	878	0.42	4.24	0.06
TRI	576	0.85	14.15	0.04
RECT	470	5.08	40.10	1.64
GBFS	458	2.49	36.71	0.05
LAZY-GBFS	464	2.63	44.36	0.05
GBFS-PO	622	0.65	5.60	0.05
LAZY-GBFS-PO	682	0.56	4.67	0.04

Triangle Search Works Pretty Good! - lm count

Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
LAMA	878	0.42	4.24	0.06
TRI	688	2.15	31.28	0.04
RECT	486	4.03	23.76	4.58
GBFS	605	4.09	26.36	0.05
GBFS-PO	593	3.18	19.04	0.07
LAZY-GBFS	597	4.67	66.07	0.06
LAZY-GBFS-PO	603	4.46	44.03	0.06

Triangle Search Works Pretty Good!

Algorithm	Δ Cov.	Improving Dom.	Improving Inst.
TRI-LM	-190	10	96
TRI-FF	-302	7	40
GBFS-FF	-420	1	4
LAZY-GBFS-FF	-414	1	2
GBFS-FF-PO	-256	4	6
LAZY-GBFS-FF-PO	-196	5	8
GBFS-LM-PO	-285	9	31
LAZY-GBFS-LM-PO	-275	9	33

Triangle Search Works Pretty Good!

Algorithm	Snake	NoMystery	Tidybot	Thoughtful	Parking
TRI-FF	21	18	13	10	9
RECT-FF	0	-11	-2	-6	12
GBFS-FF	0	2	2	11	12
LAZY-GBFS-FF	0	7	4	8	13
GBFS-FF-PO	0	0	2	3	12
LAZY-GBFS-FF-PO	0	7	8	4	13
TRI-LM	6	14	3	10	-6
RECT-LM	-19	-11	-13	-6	-4
LAZY-GBFS-LM-PO	2	8	3	5	7

If the Building Catches Fire During the Talk...

- Triangle Search performs well in classic heuristic search benchmarks
- It also performs extremely well in planning
  - better than our usual base strategies
- This points to two opportunities:
  - Exporting planning-specific search enhancements to general search problems
  - Importing those enhancements to Triangle Search to improve planners

Material for Questions I Think You Might Ask!

Triangle Search Works Pretty Good! - h_ff

Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
TRI	576	0.85	14.15	0.04
RECT	470	5.08	40.10	1.64
GBFS	458	2.49	36.71	0.05
LAZY-GBFS	464	2.63	44.36	0.05
GBFS-PO	622	0.65	5.60	0.05
LAZY-GBFS-PO	682	0.56	4.67	0.04

Triangle Search Works Pretty Good! - h_ff

Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
LAMA	878	0.42	4.24	0.06
TRI	576	0.85	14.15	0.04
RECT	470	5.08	40.10	1.64
GBFS	458	2.49	36.71	0.05

Triangle Search Works Pretty Good! - h_ff

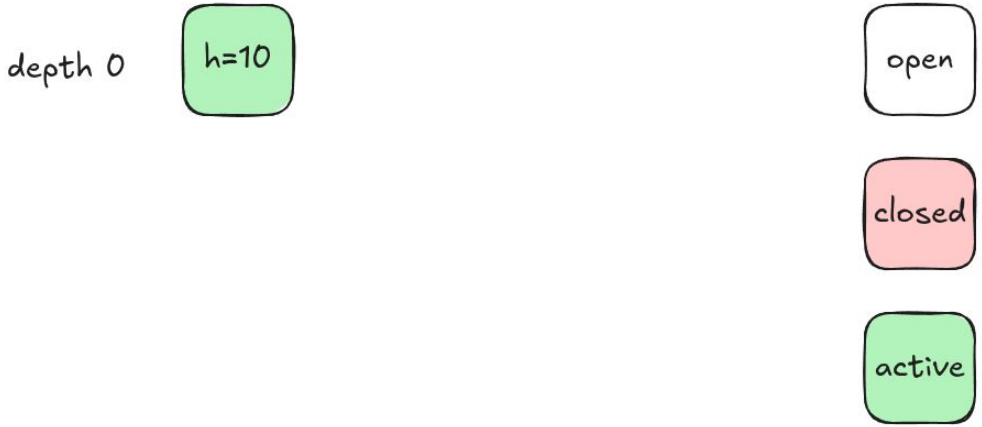
Algorithm	Cov.	Exp. (100k)	Gen. (100k)	Sec.
LAMA	878	0.42	4.24	0.06
TRI	576	0.85	14.15	0.04
RECT	470	5.08	40.10	1.64
GBFS	458	2.49	36.71	0.05
LAZY-GBFS	464	2.63	44.36	0.05
GBFS-PO	622	0.65	5.60	0.05
LAZY-GBFS-PO	682	0.56	4.67	0.04

Ok, But Why Does It Work Pretty Good?

- Compare to Random Walks to Escape Plateaus
  - Slope length random walk
- Compare to Greedy Policy Following
  - Slope length greedy h following
  - Bypassing global search order with useful fallback
- Compare to LAMA
  - Slope similar to boost
  - Multiple queues hoping to drive fairer comparison

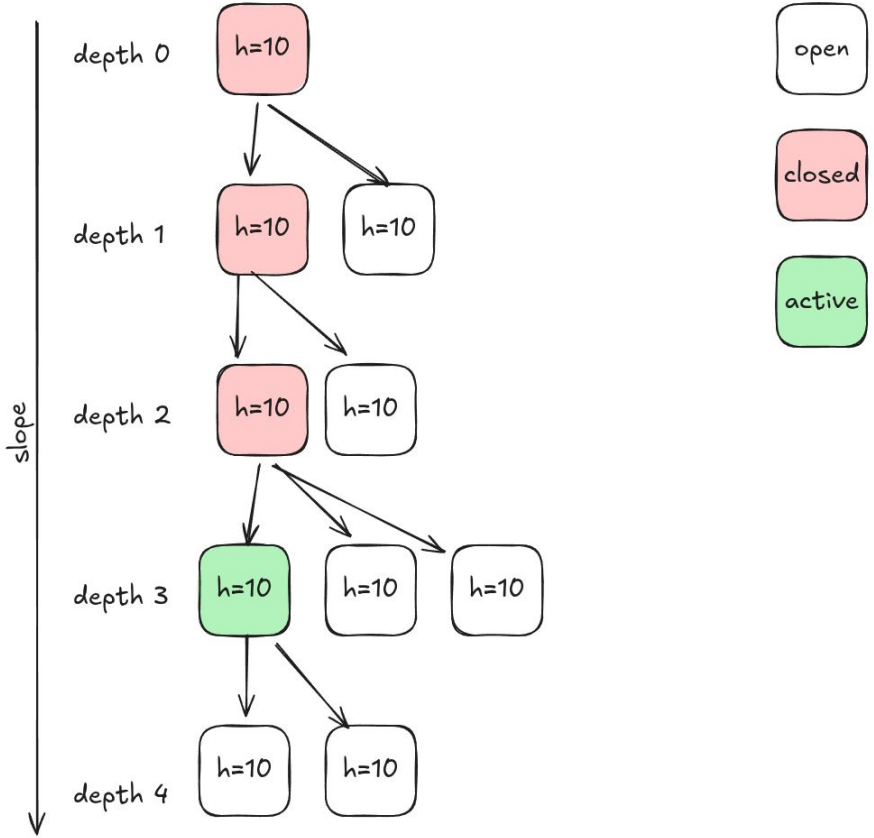
Triangle Search

Triangle search feels a lot like
Enforced hill climbing



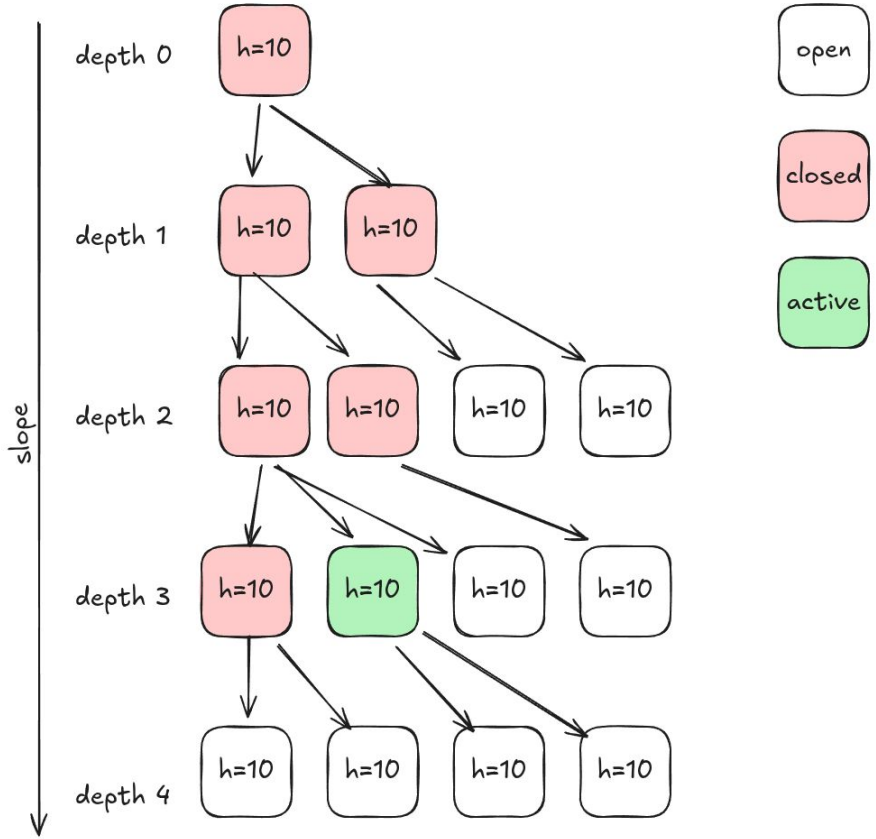
Ok, But Why Does It Work Pretty Good?

Random Walk



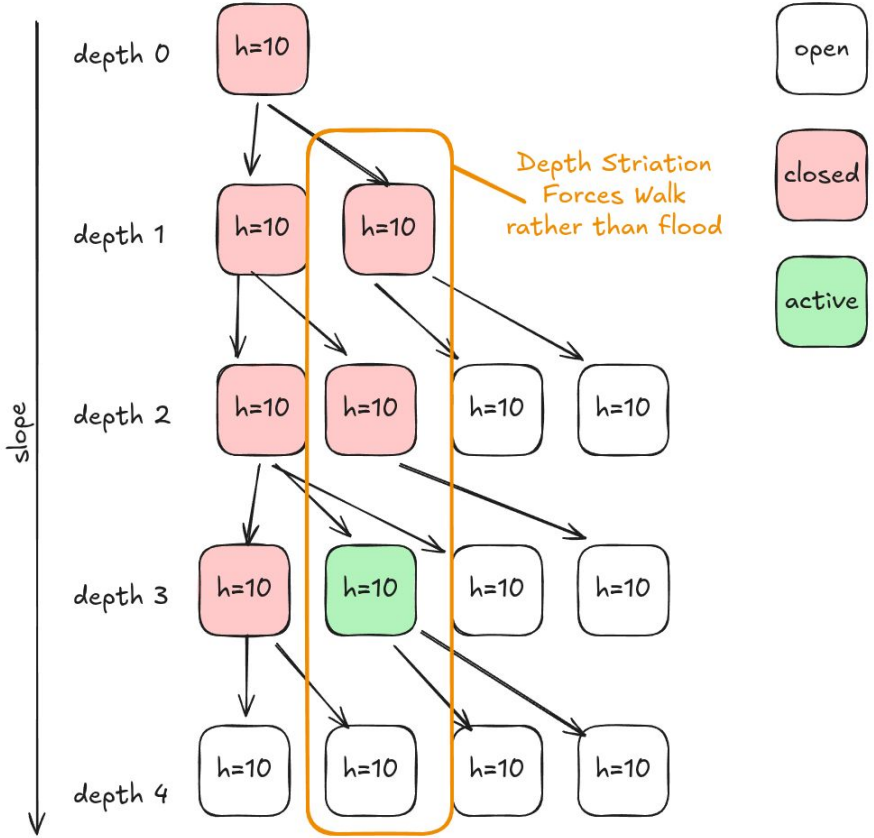
Ok, But Why Does It Work Pretty Good?

Random Walk



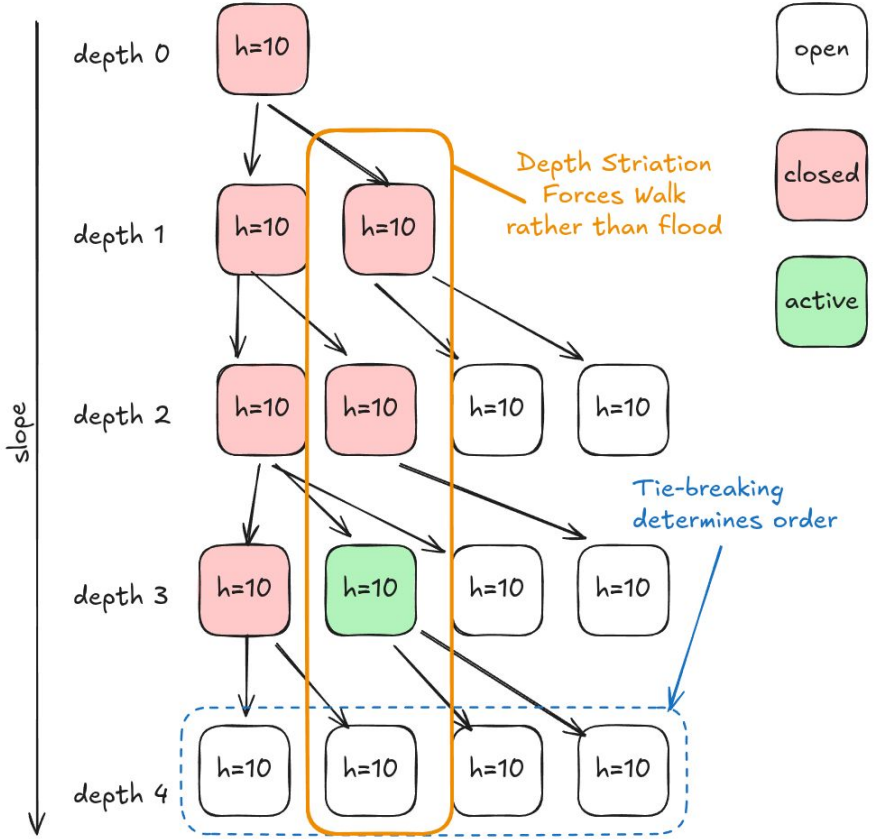
Ok, But Why Does It Work Pretty Good?

Random Walk



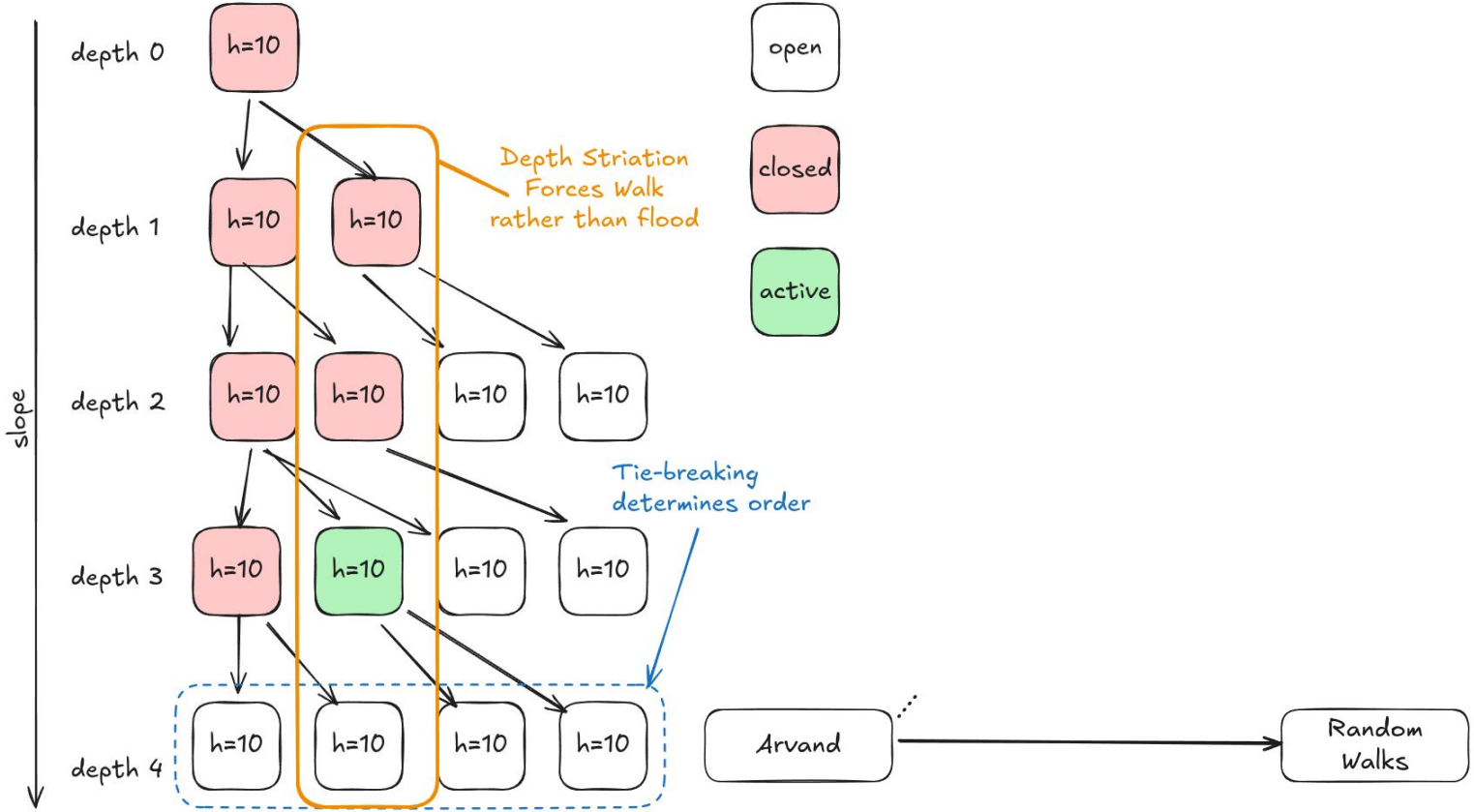
Ok, But Why Does It Work Pretty Good?

Random Walk



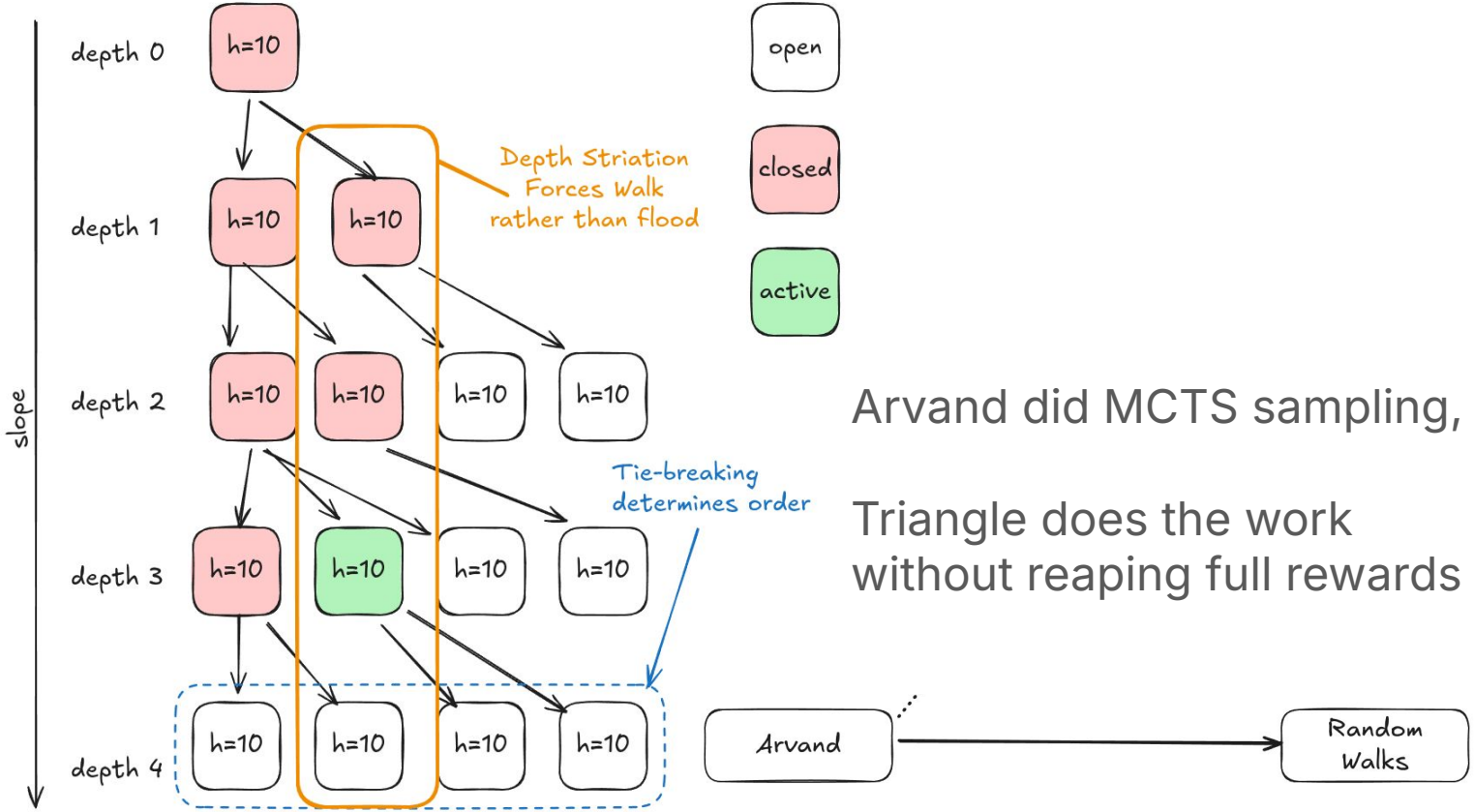
Ok, But Why Does It Work Pretty Good?

Random Walk



Ok, But Why Does It Work Pretty Good?

Random Walk

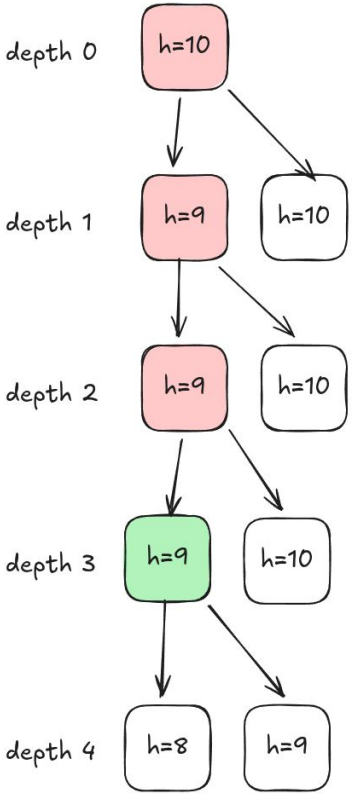


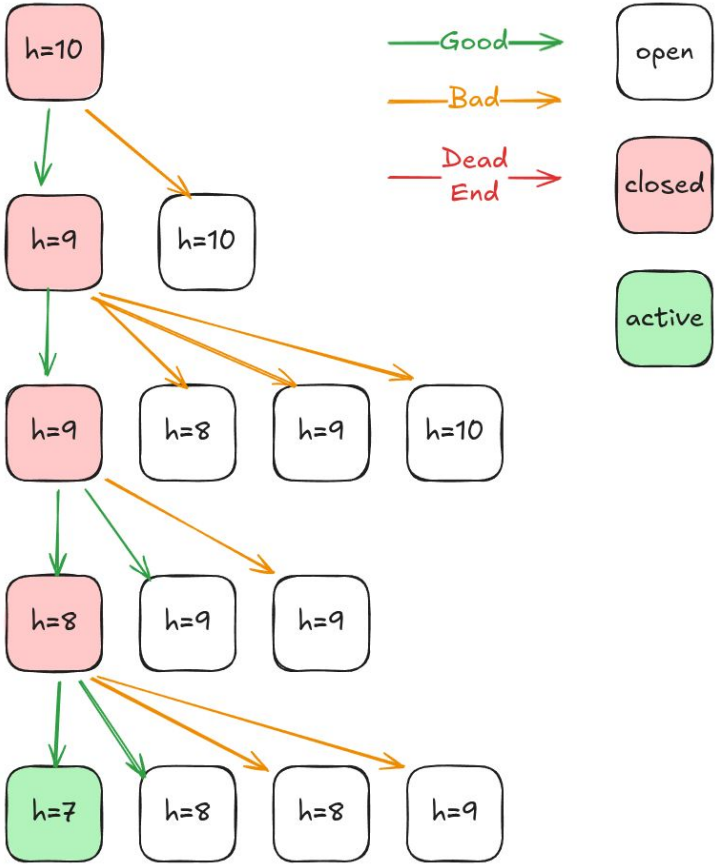
Arvand did MCTS sampling,

Triangle does the work without reaping full rewards

Ok, But Why Does It Work Pretty Good?

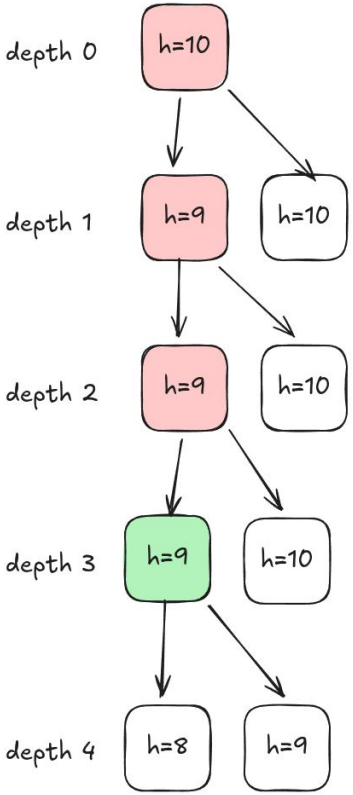
Policy Following

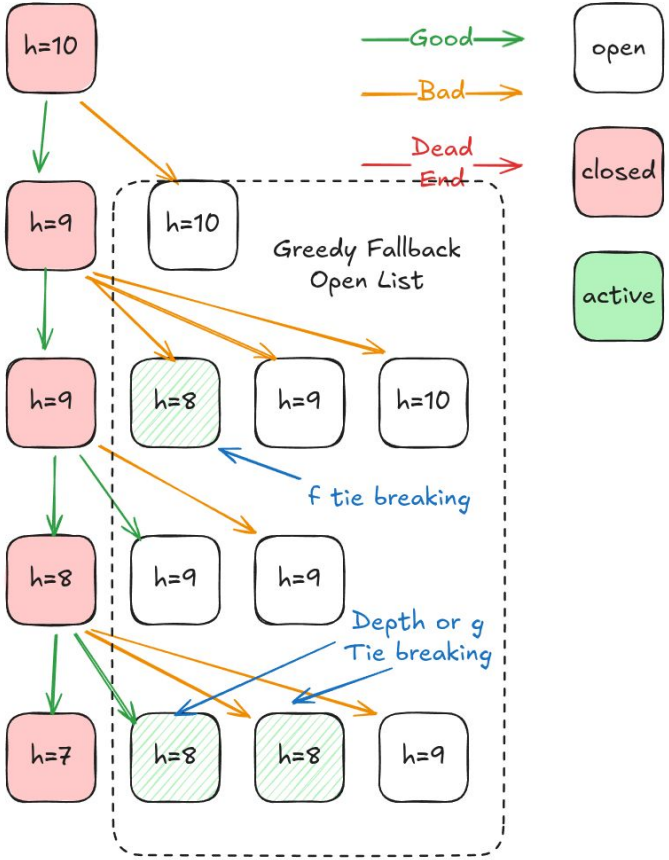




Ok, But Why Does It Work Pretty Good?

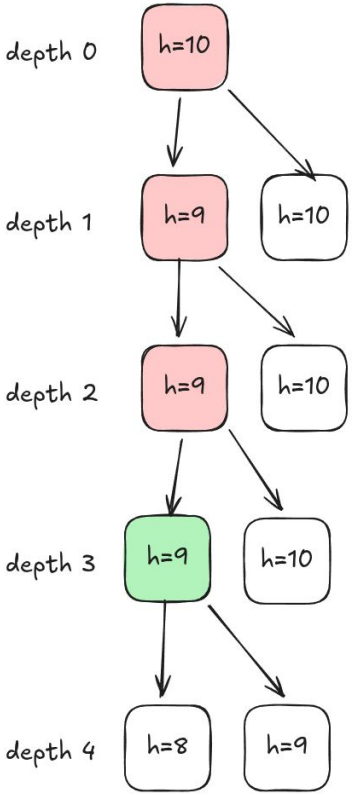
Policy Following

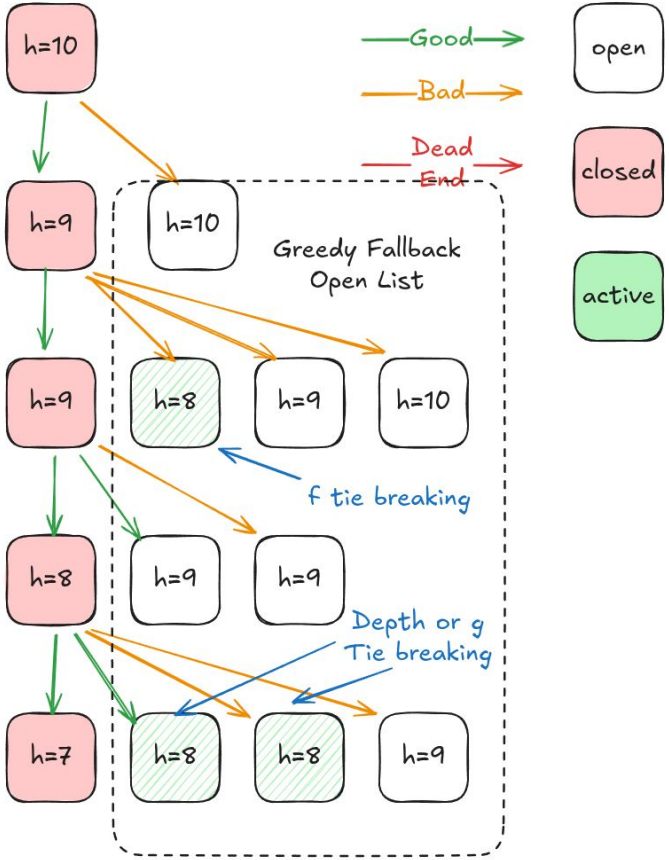




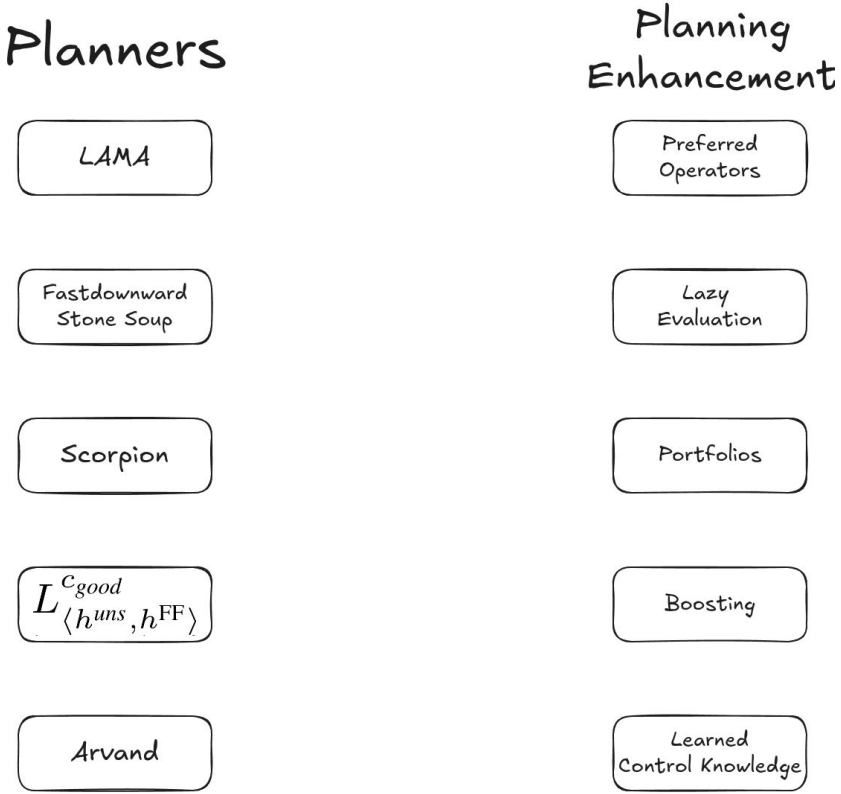
Ok, But Why Does It Work Pretty Good?

Policy Following

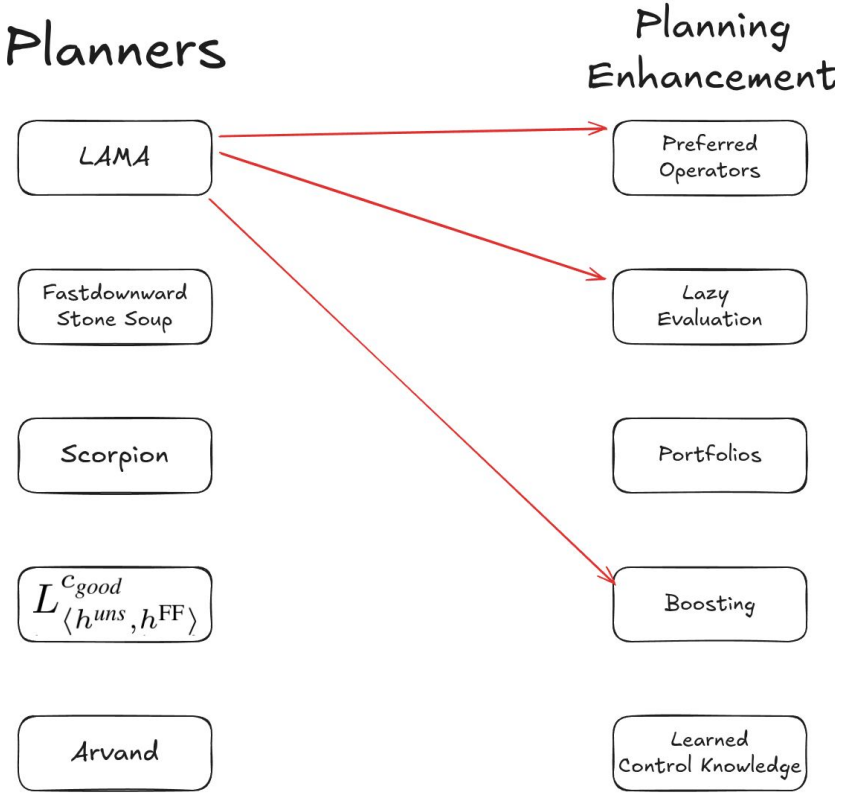




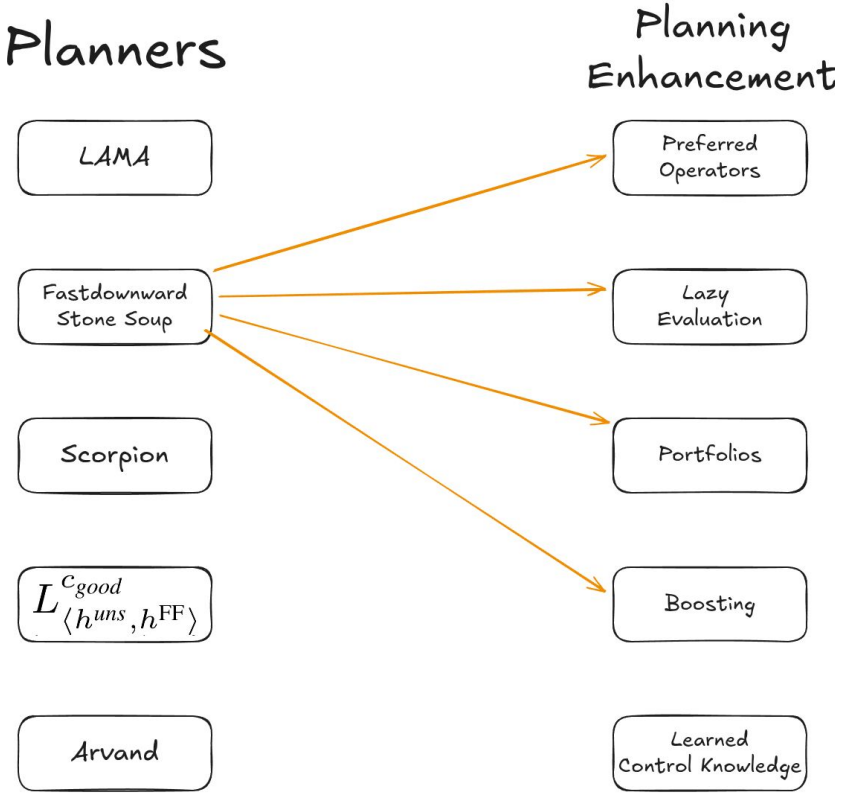
Planners Enhance Basic Search Techniques



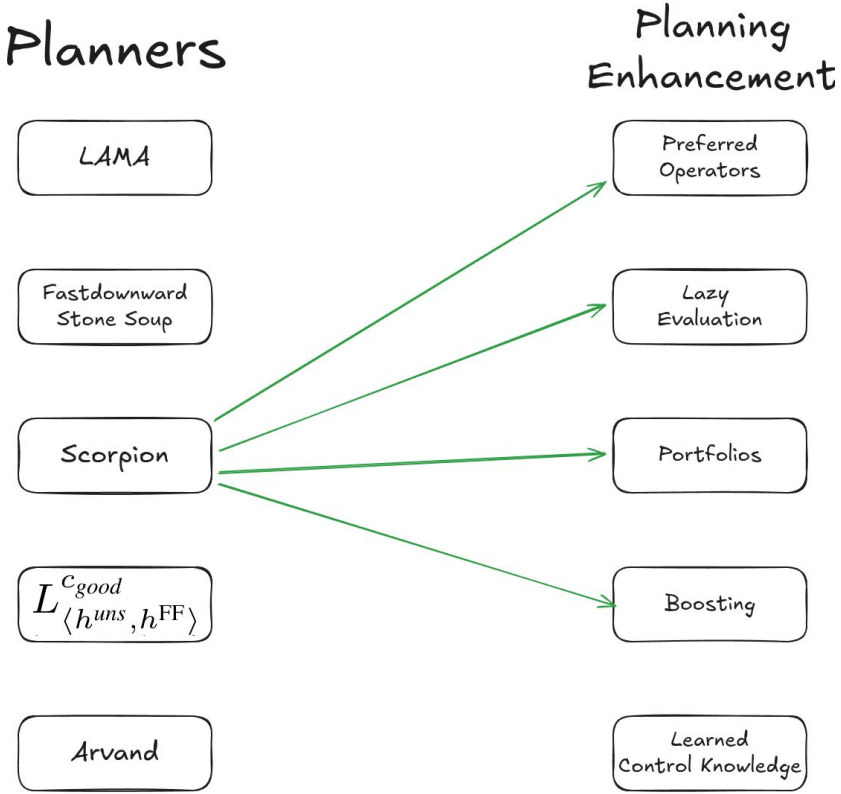
Planners Enhance Basic Search Techniques



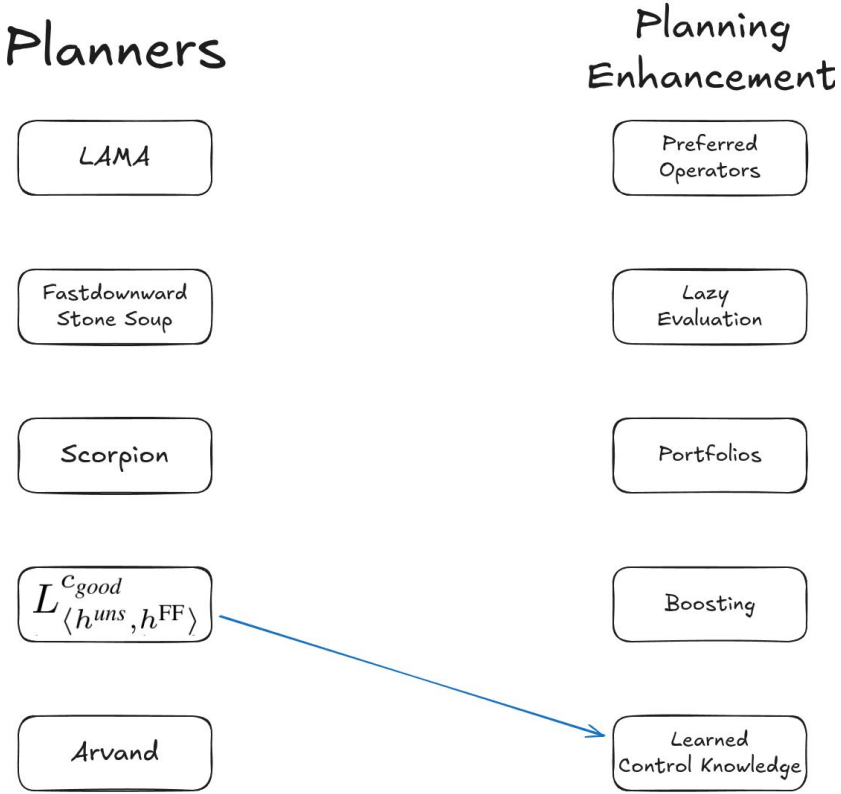
Planners Enhance Basic Search Techniques



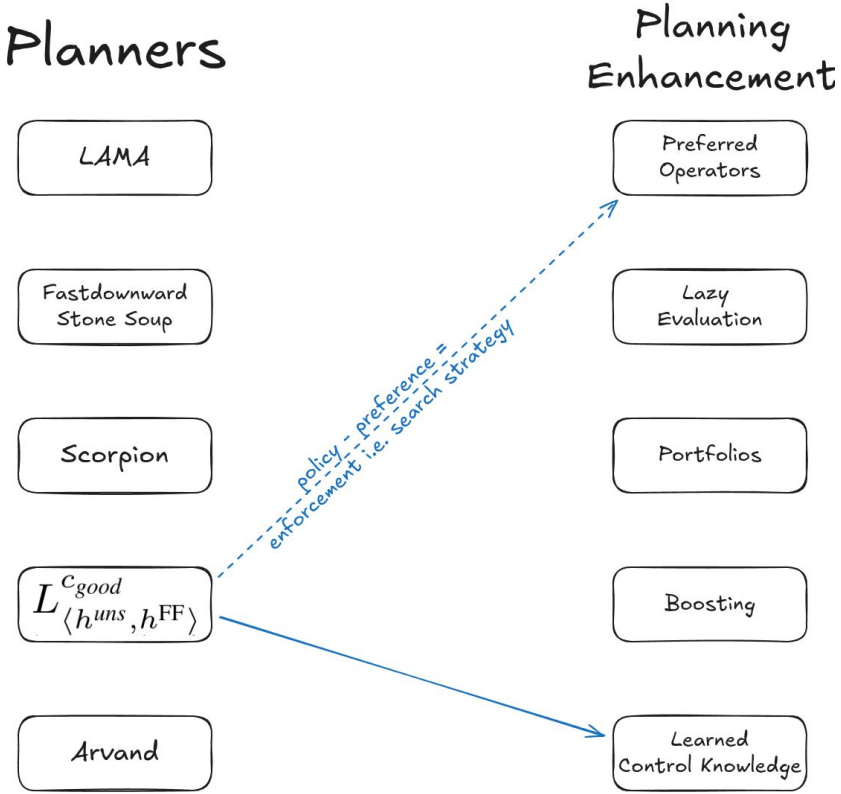
Planners Enhance Basic Search Techniques



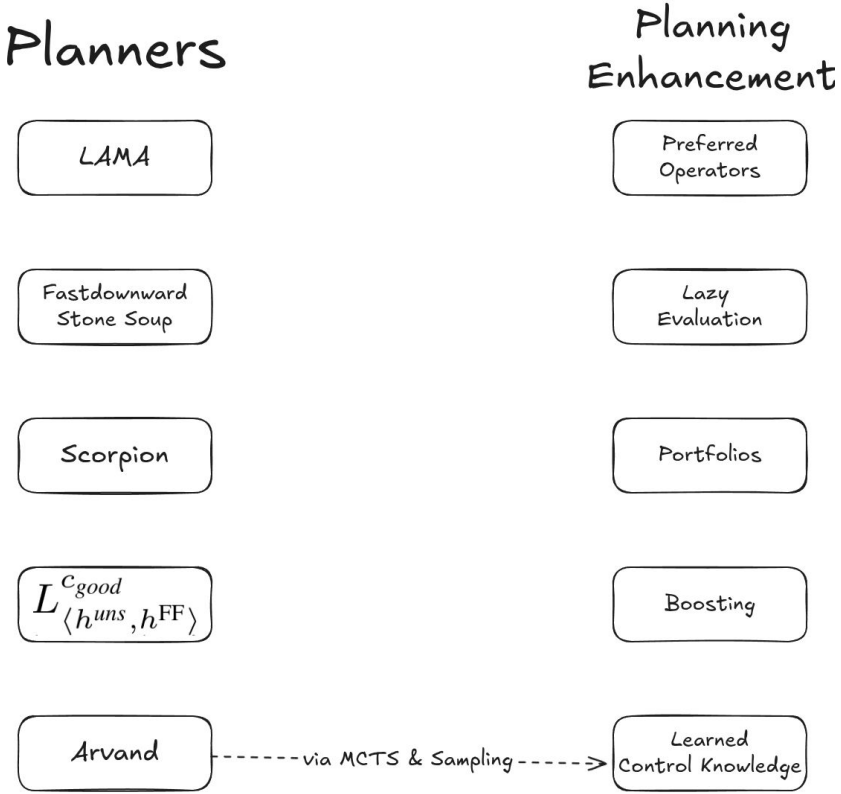
Planners Enhance Basic Search Techniques



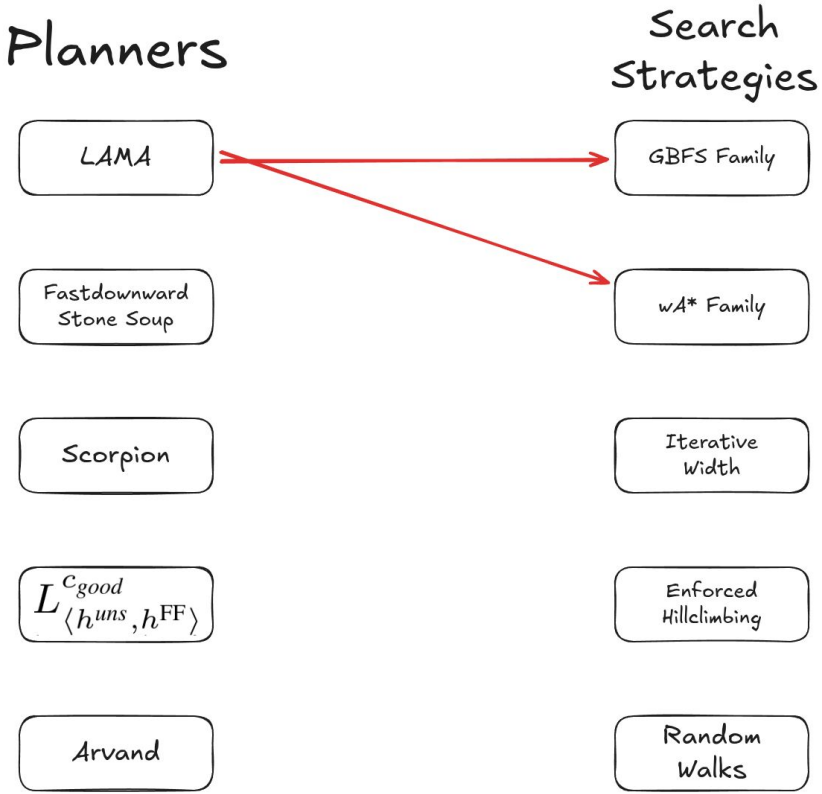
Planners Enhance Basic Search Techniques



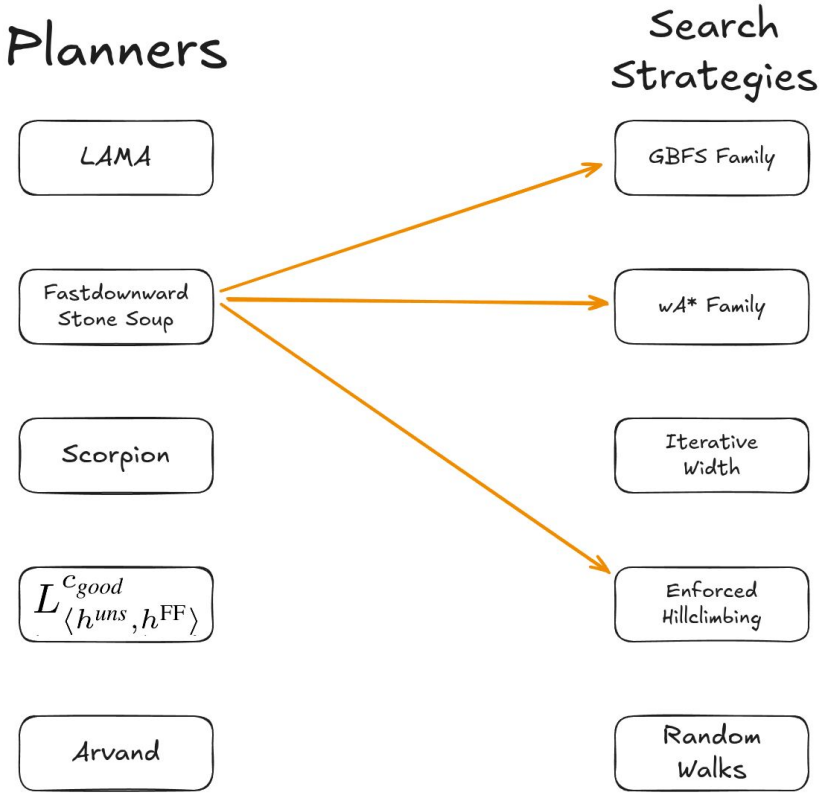
Planners Enhance Basic Search Techniques



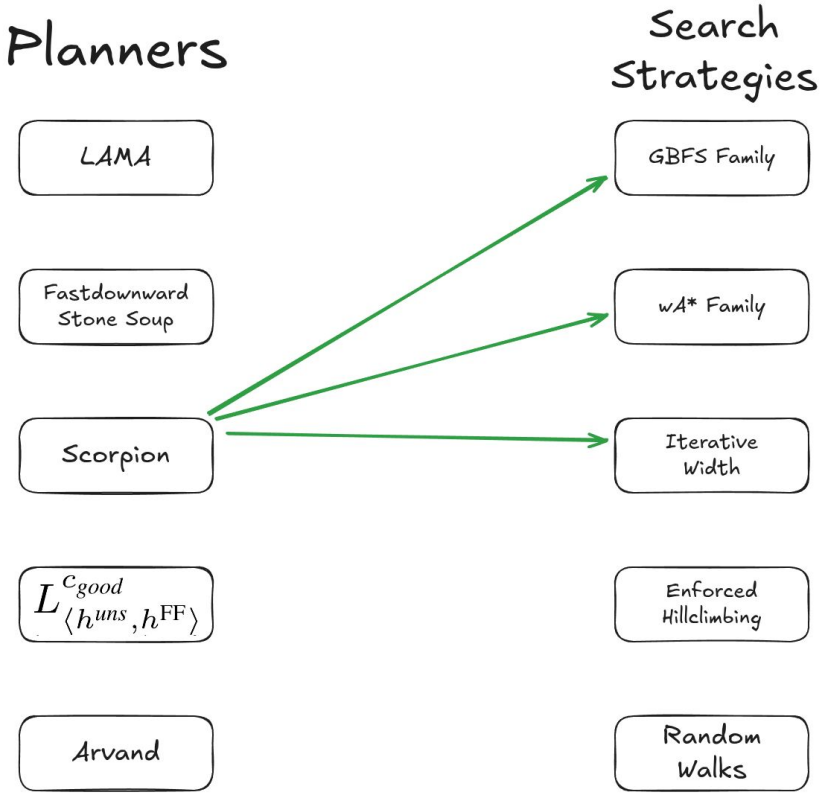
Planning Systems Use Search Under the Hood



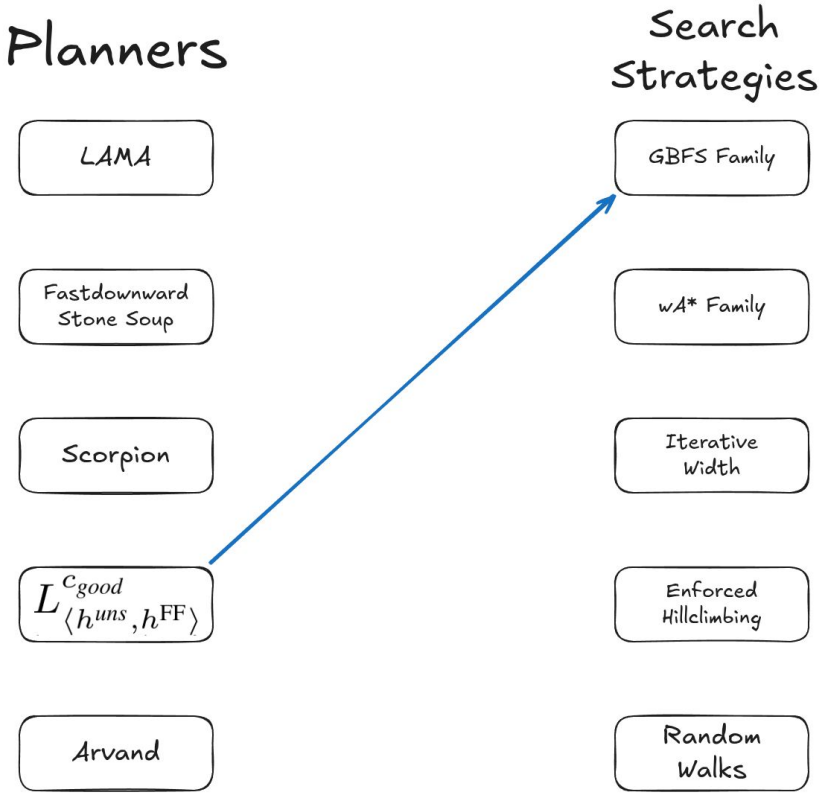
Planning Systems Use Search Under the Hood



Planning Systems Use Search Under the Hood



Planning Systems Use Search Under the Hood



Planning Systems Use Search Under the Hood