Bilevel MCTS for Amortized O(1) Node Selection in Classical Planning

Masataro Asai
MIT-IBM Watson AI Lab
IBM Research

Use arrow keys (← →) or n/p keys

If you found the pdf version of this document, you can find the orignal web presentation in https://asai-research-presentation.github.io/2026-aaai26-O1MCTS/
Created: 2025-12-28 Sun 15:36

1. Overview

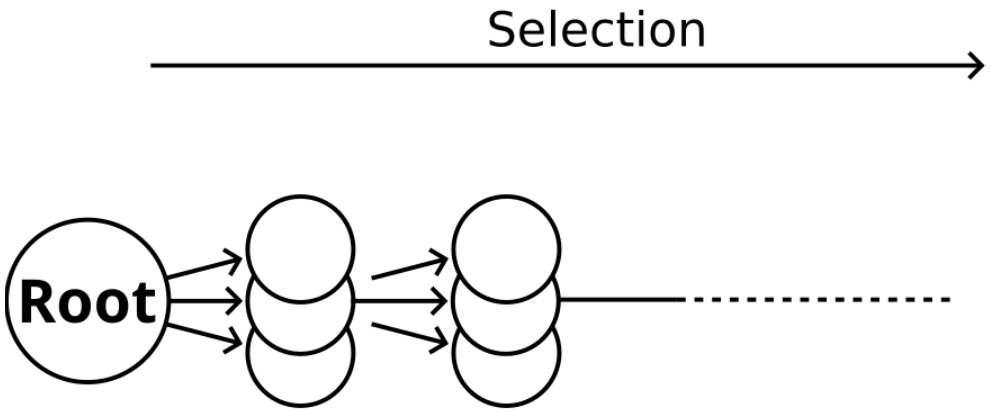
Common Weakness of Monte-Carlo Tree Search

Bilevel MCTS

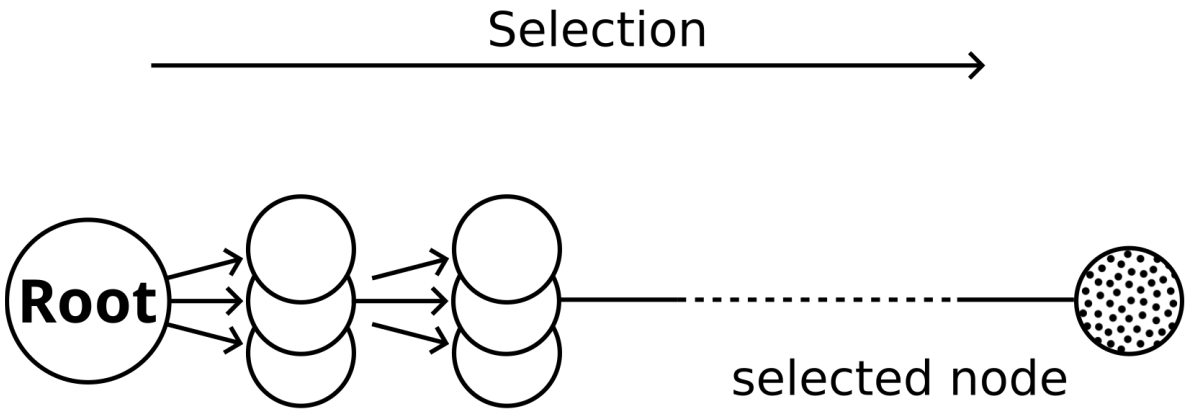
Tree Collapsing

Nebula planner: SoTA results (MCTS+LAMA+BFWS)

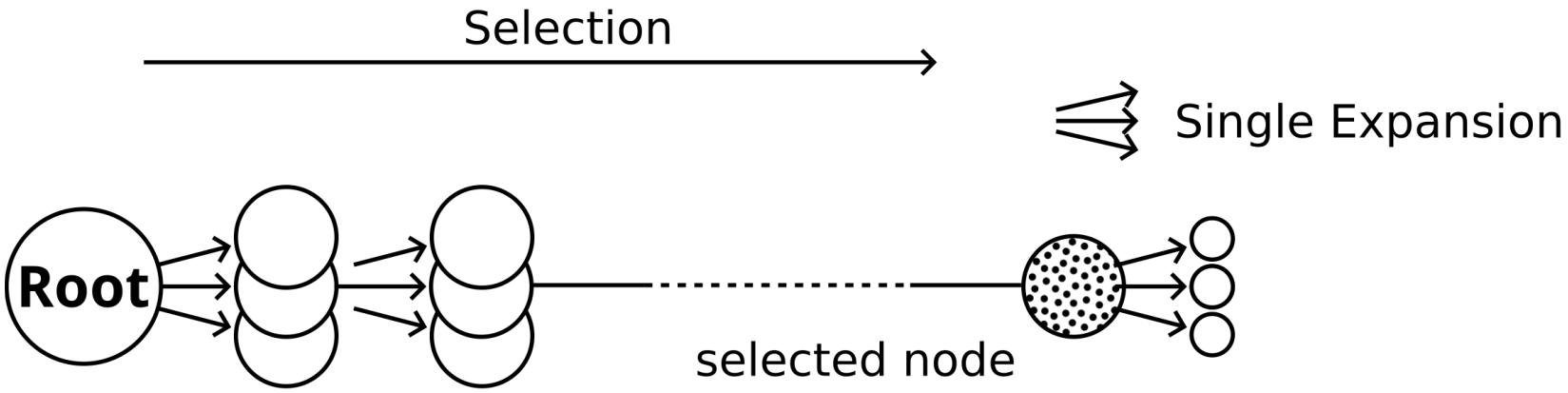
2. Monte Carlo Tree Search



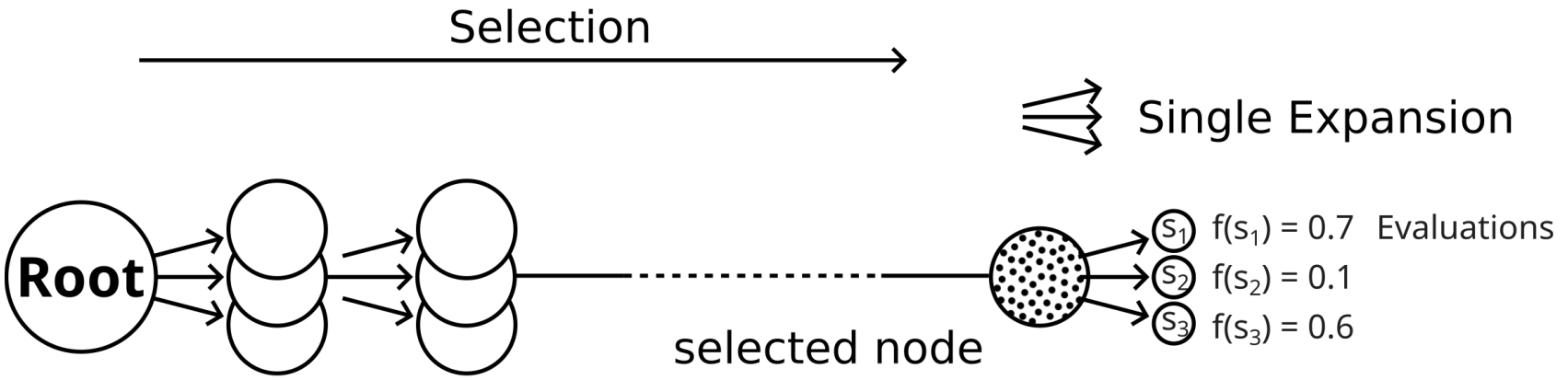
2.1. Monte Carlo Tree Search



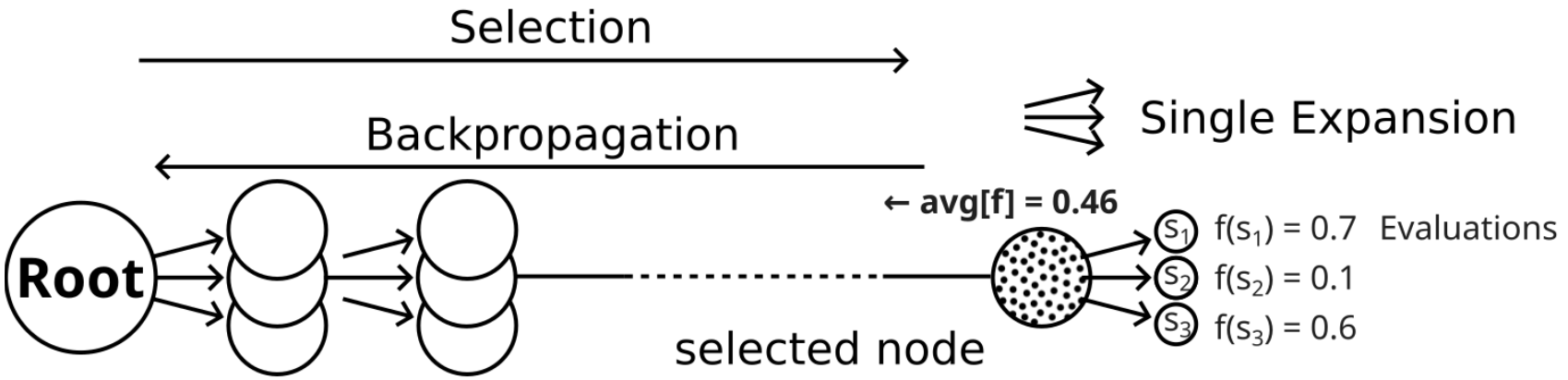
2.2. Monte Carlo Tree Search



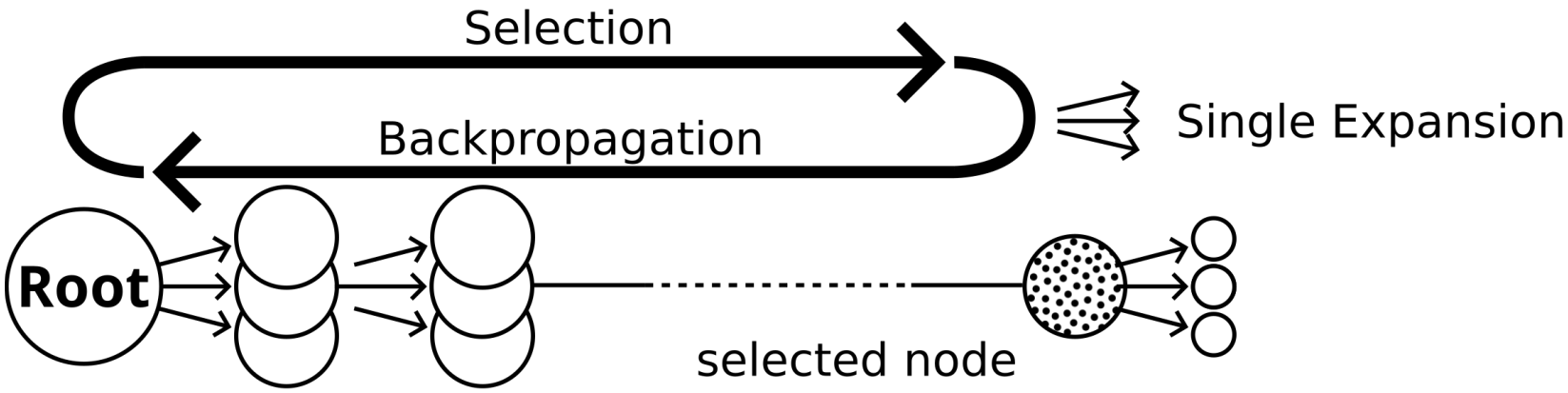
2.3. Monte Carlo Tree Search



2.4. Monte Carlo Tree Search

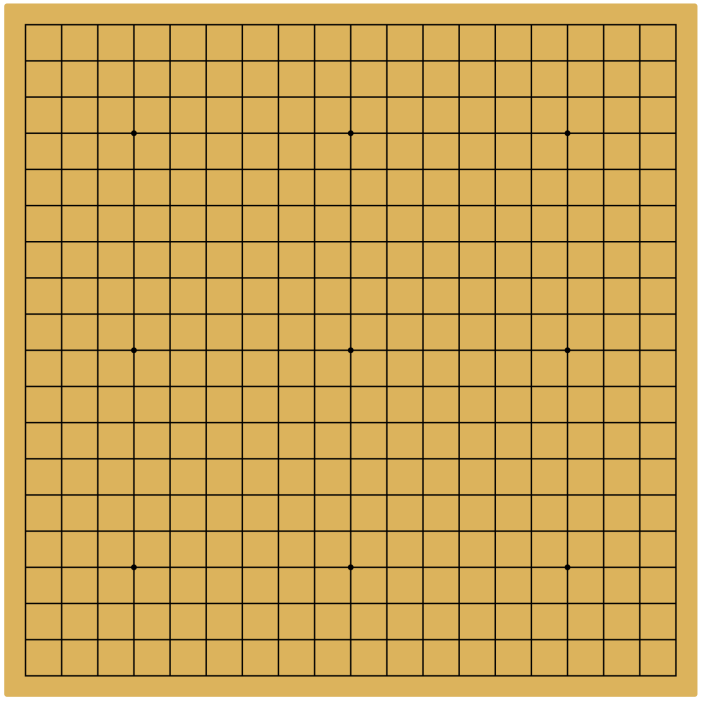


2.5. Monte Carlo Tree Search



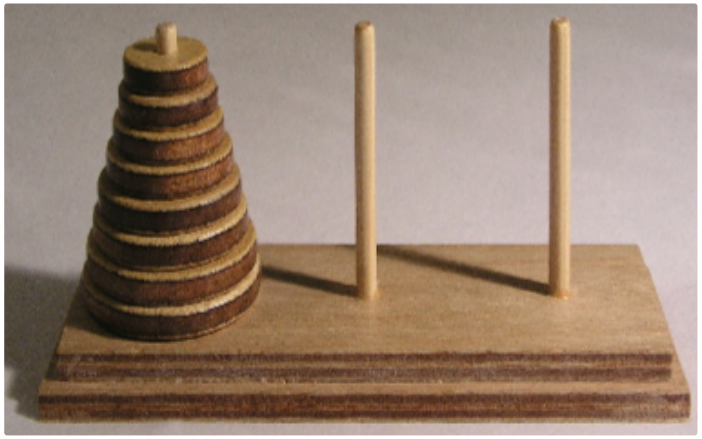
3. ***Weakness*** : needs a ***shallow search space***

Game of Go



Depth $d < 19^2 = 391$

Tower of Hanoi

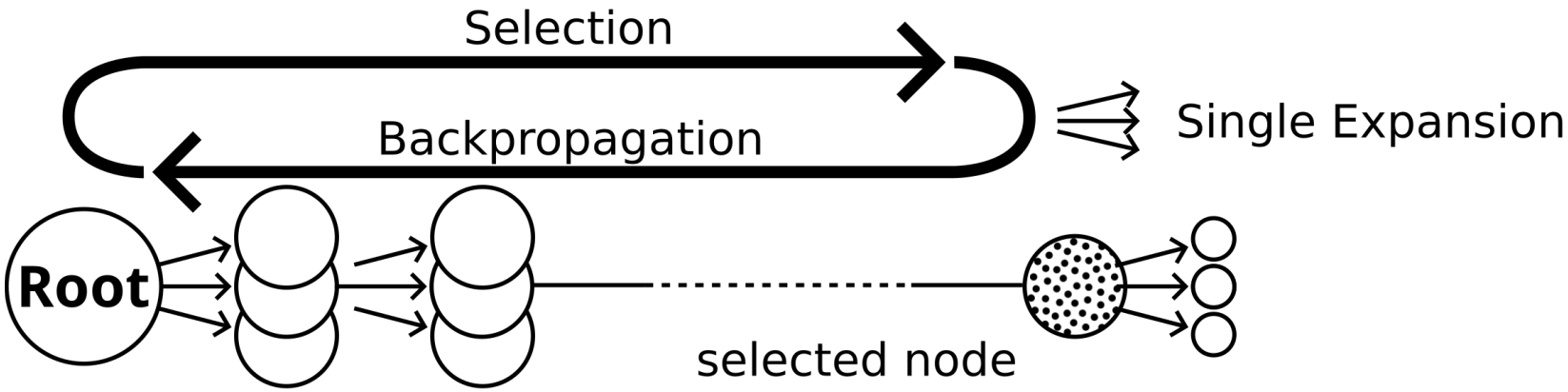


Depth $d = 2^k - 1$ (k disks)

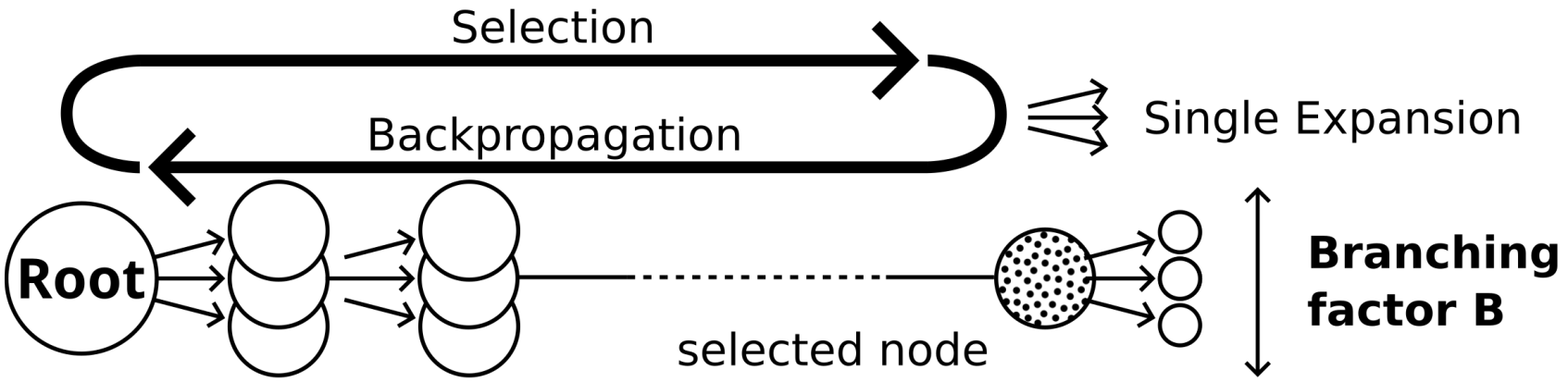
.

Single Player Planning often requires an *exponential* depth!

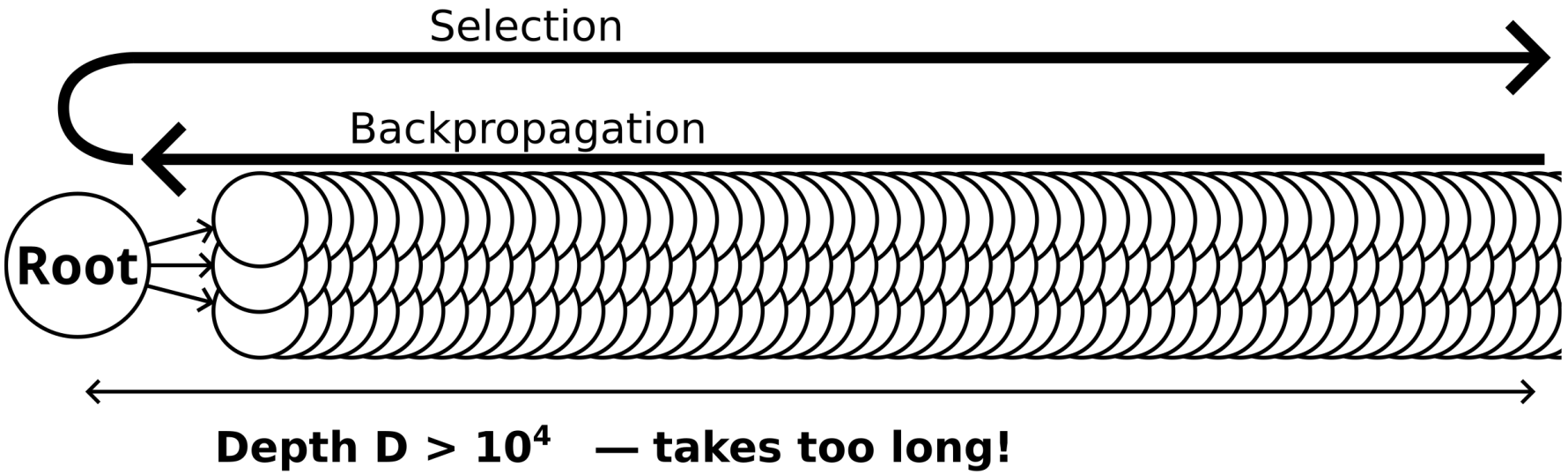
4. ***Weakness*** : Larger d → Slower MCTS



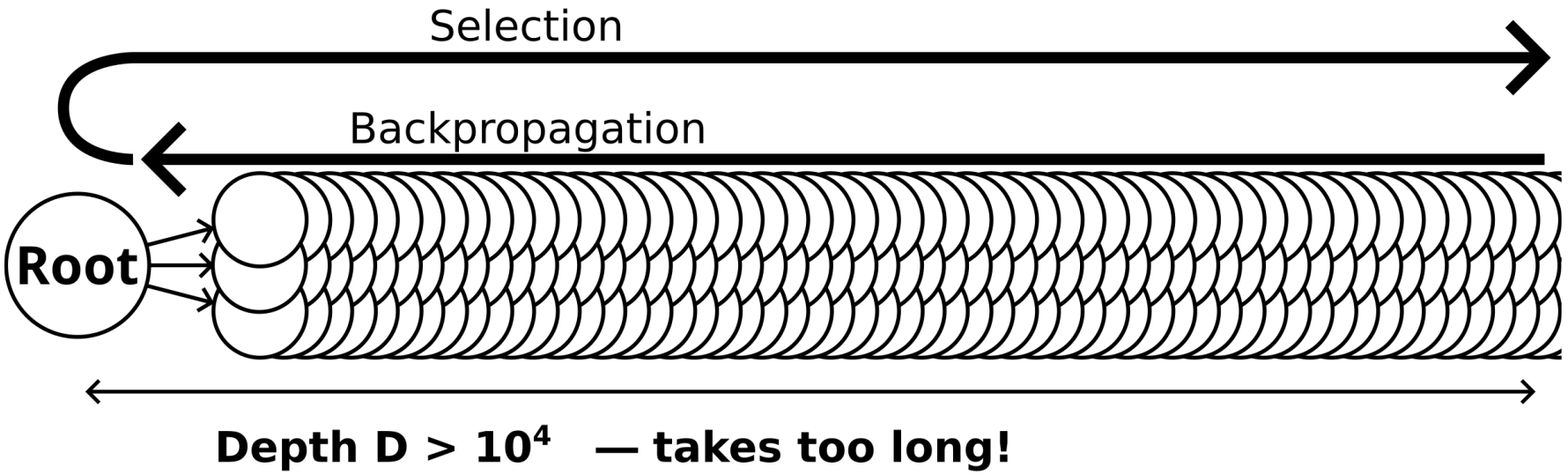
4.1. ***Weakness*** : Larger d → Slower MCTS



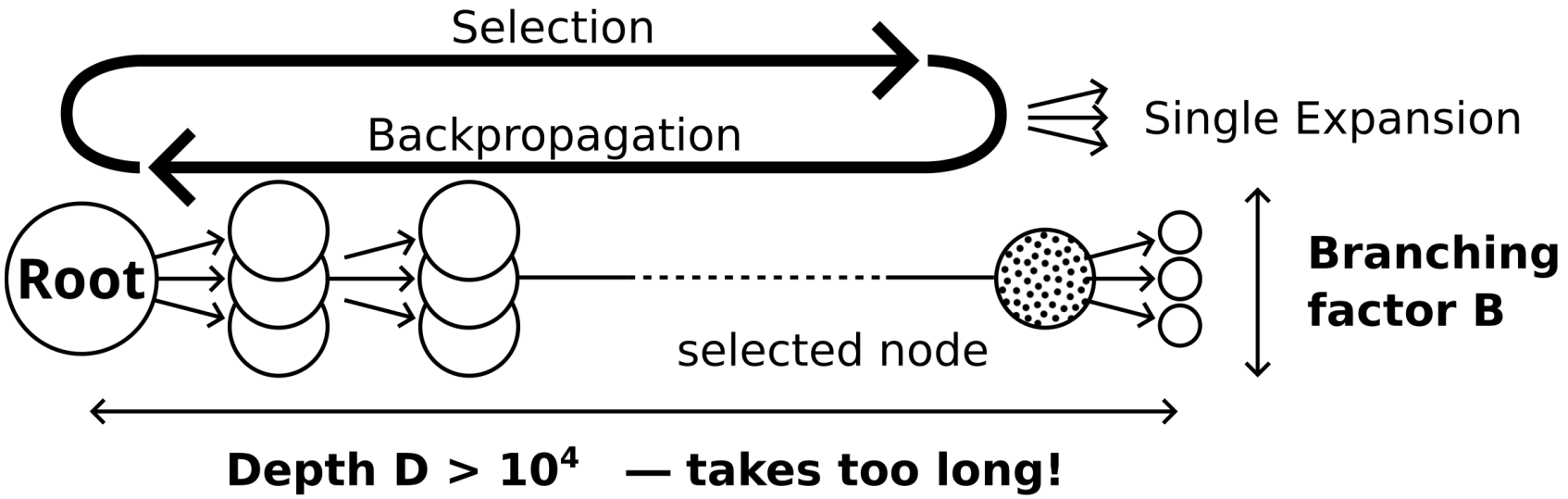
4.2. ***Weakness*** : Larger d → Slower MCTS



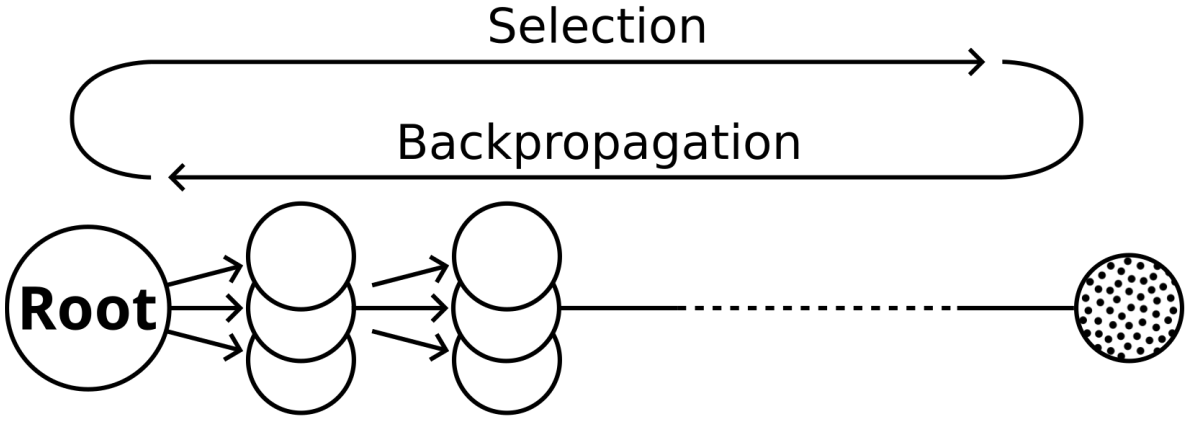
5. Proposed: ***Bilevel MCTS***



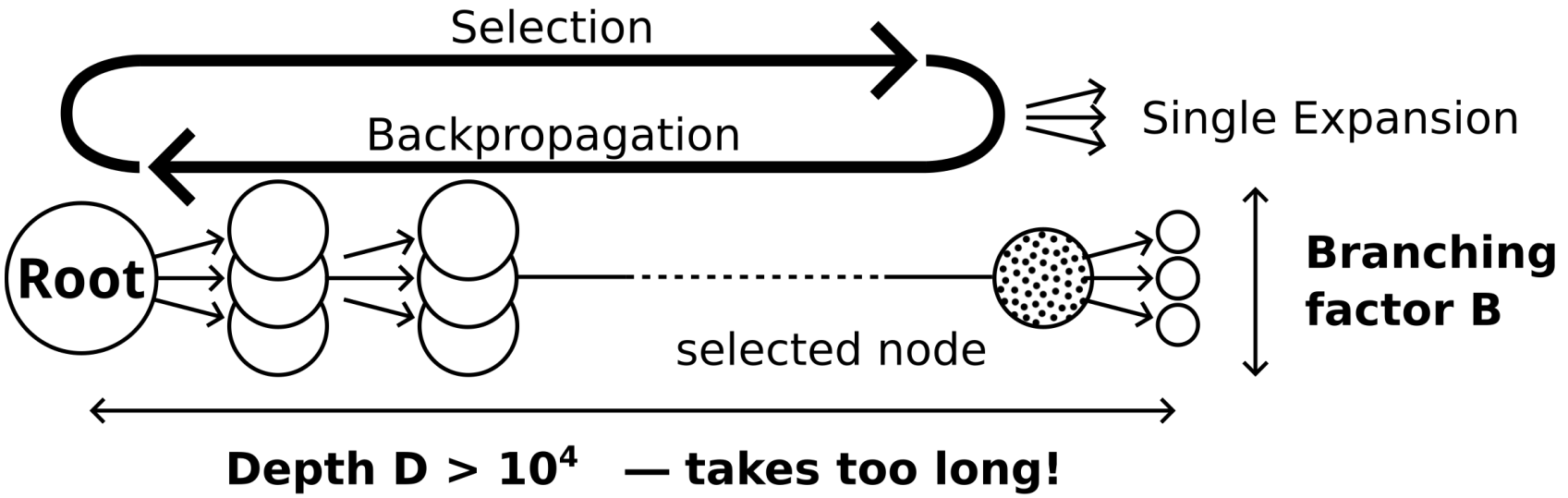
5.1. Proposed: ***Bilevel MCTS***

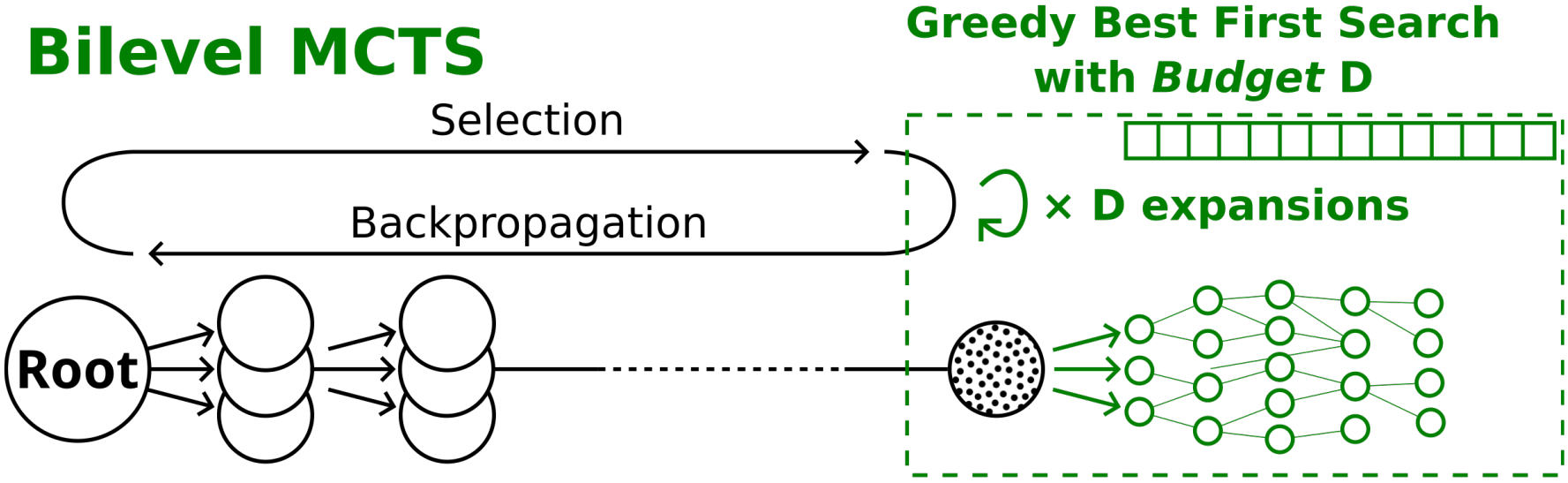


Bilevel MCTS

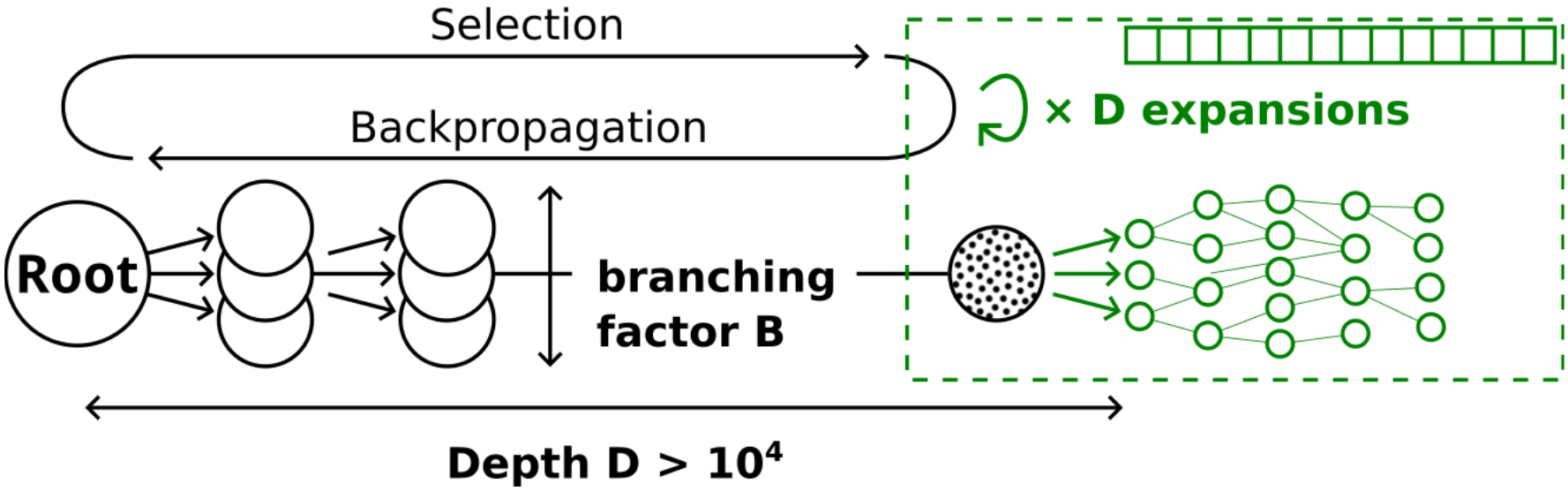


5.2. Proposed: ***Bilevel MCTS***

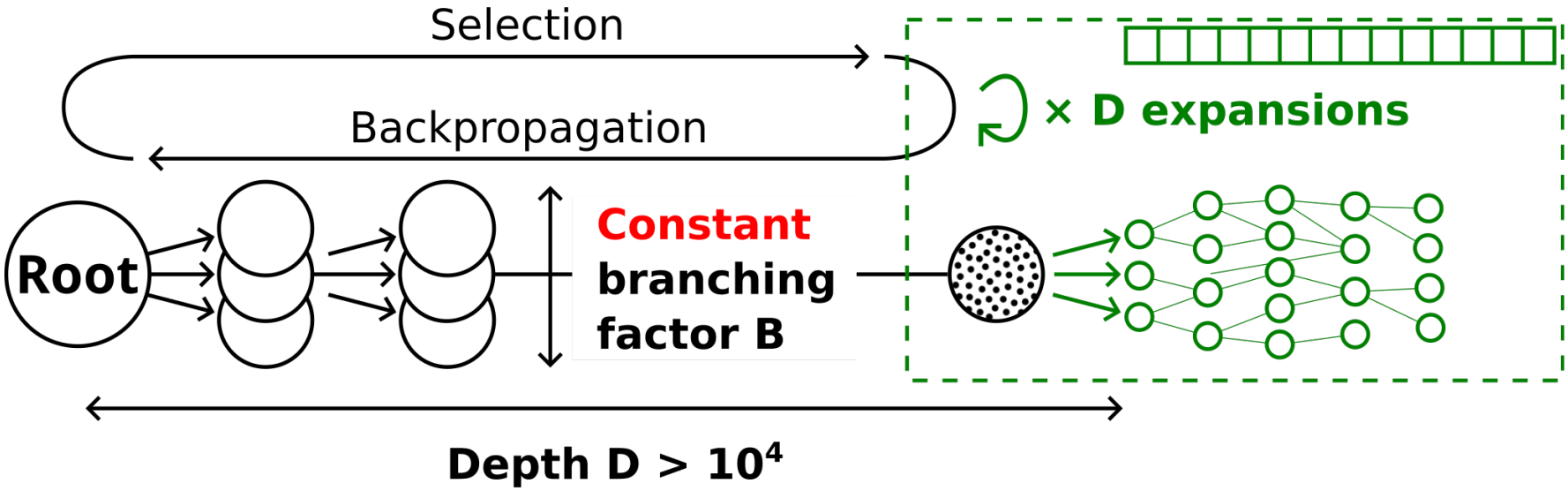




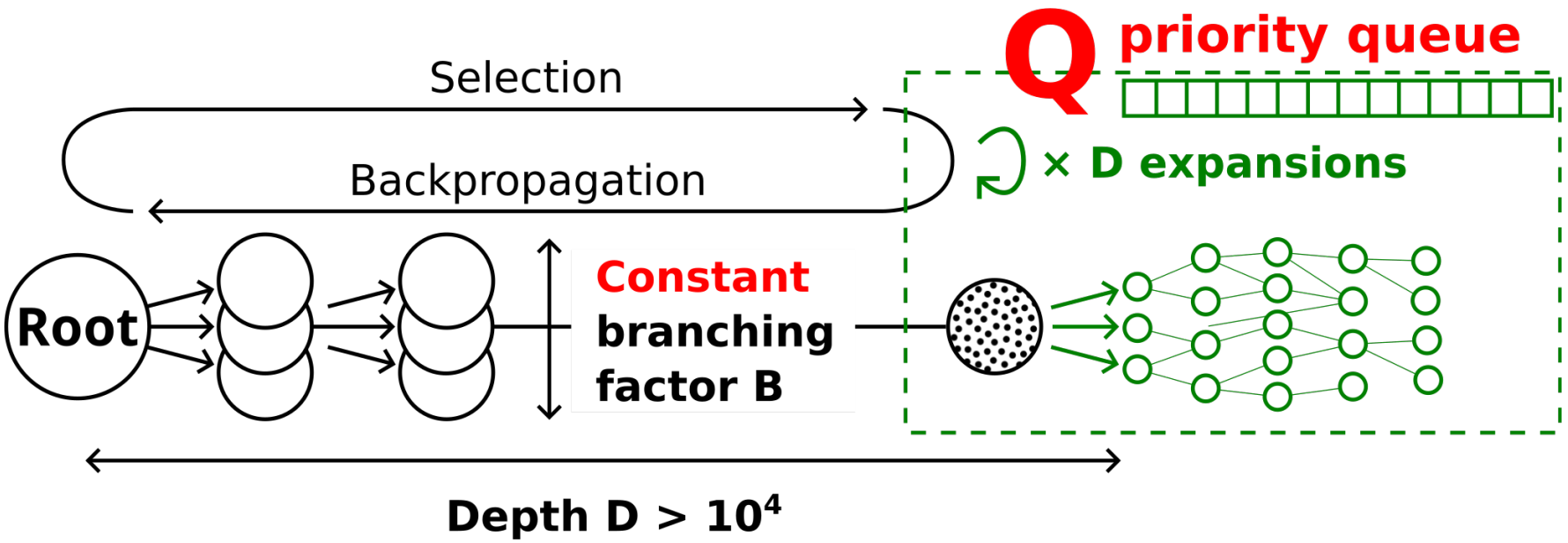
6. Theoretical Results



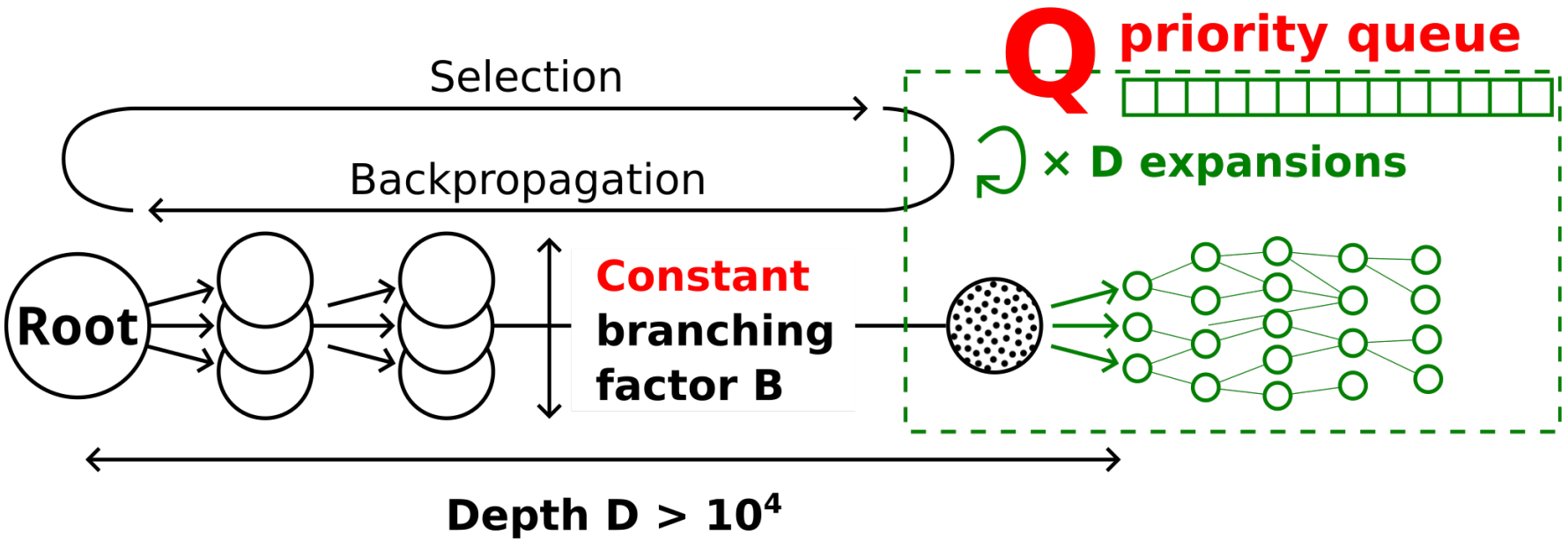
6.1. Theoretical Results



6.2. Theoretical Results



6.3. Theoretical Results



MCTS *runtime per node*

$O(BD) = O(\log N)$. N : number of nodes

Bilevel MCTS *amortized runtime per node* over D expansions

$O(\log \log N)$ if Q = heap

$O(1)$ if Q = array-based priority queue (Dial 1969, Burns 2012)

7. Evaluation on IPC 2018+2023

Instances solved in 5 min

	GBFS	Softmin	MCTS	***Bilevel***
cea/18	40	48.4	50.2	**53.8**
cea/23	30	**34.2**	25.8	30.8
cg/18	35	**59.2**	42.4	56.2
cg/23	36	39.4	25	**41.6**
ff/18	60	80.2	79.8	**87.4**
ff/23	51	55.4	23.8	**58.2**
total	252	316.8	247	**328**

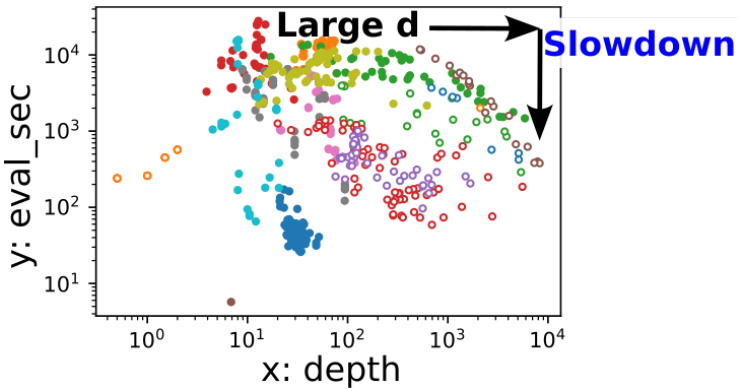
IPC Agile scores

	GBFS	Softmin	MCTS	***Bilevel***
cea/18	17.7	20.6	**23.8**	22.4
cea/23	23.1	**24.1**	21.5	23.8
cg/18	18.4	**27.2**	19.9	26.1
cg/23	23.6	24.8	16.4	**25.5**
ff/18	30.4	37.7	40.4	**43.0**
ff/23	31.0	**35.5**	19.0	32.9
total	144.2	169.9	141.0	**173.8**

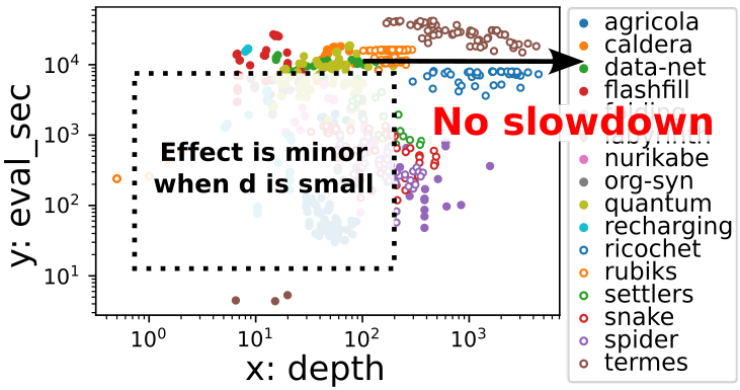
Softmin: Softmin-type-h (Kuroiwa and Beck 2022)

8. Expansion Speed [Nodes / sec]

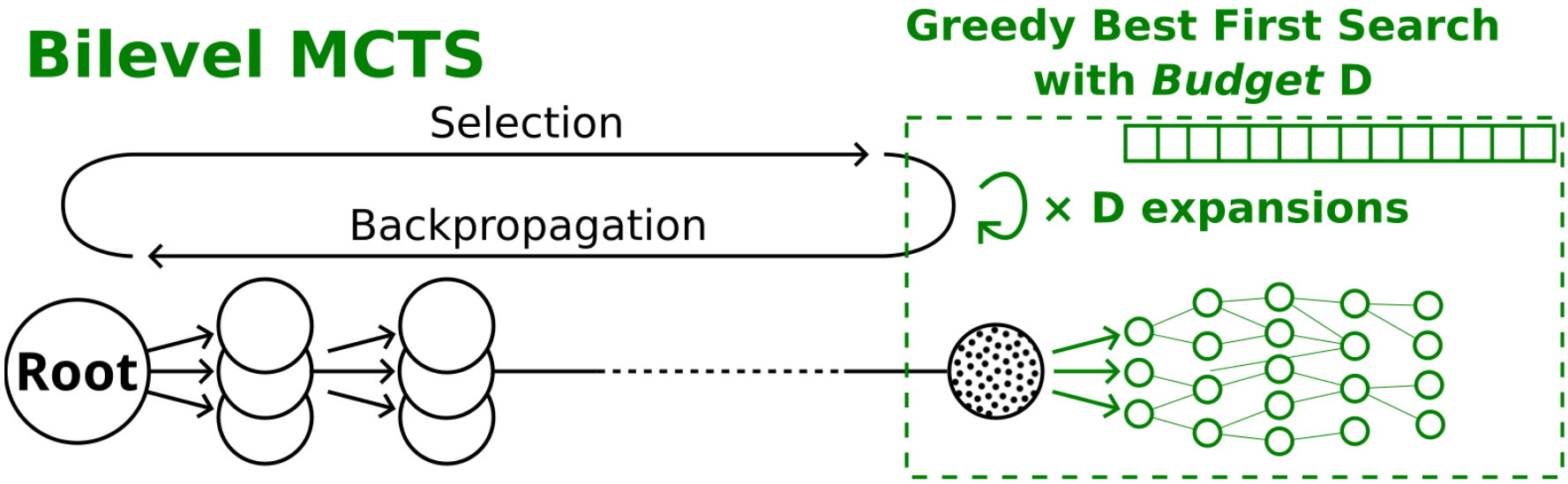
MCTS



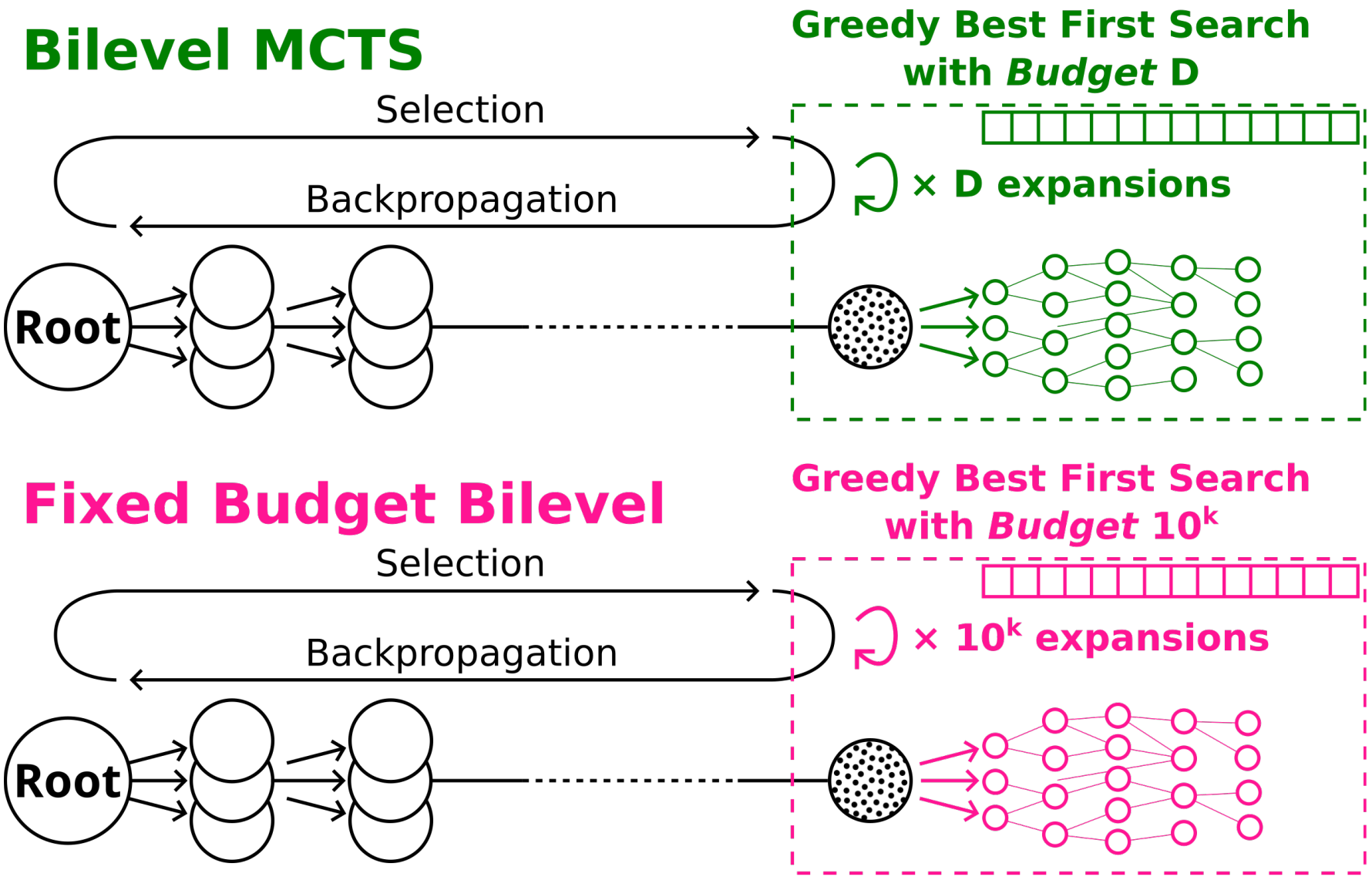
Bilevel MCTS



9. Bilevel MCTS uses a *Dynamic Depth Budget*



9.1. Bilevel MCTS uses a *Dynamic Depth Budget*



10. *Fixed Budget* $b = 10^k$ vs. *Depth Budget* $b = d$

Instances solved in 5 min

	b=1	*10*	*10^2*	*10^3*	*10^4*	*b=d*
cea/18	50.2	**57.6**	51.8	49	46.6	53.8
cea/23	25.8	32.4	**36.8**	33.8	33.6	30.8
cg/18	42.4	51.6	55.8	48.6	48.8	**56.2**
cg/23	25	34.2	42.2	**44**	42.2	41.6
ff/18	79.8	85.8	83.4	82.2	72.8	**87.4**
ff/23	23.8	45.4	56.8	**60**	58.6	58.2
total	247	↗307	↗326.8↘	317.6↘	302.6	**328**

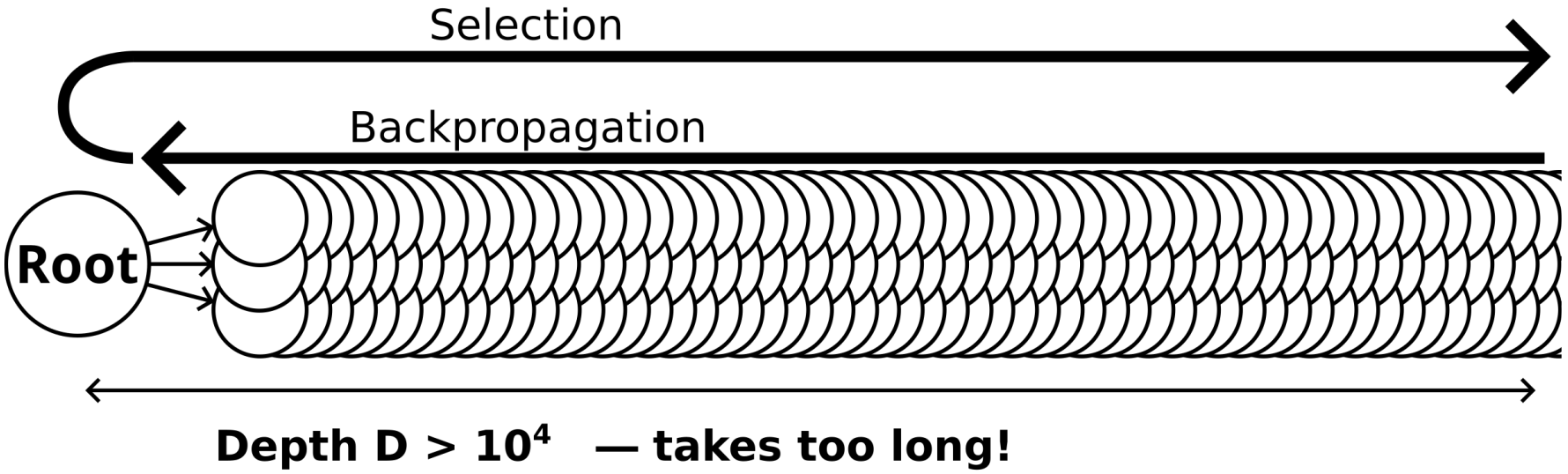
Fixed budget

Too many / few BFS

Dynamic budget

Handles various scenarios well

11. **If a large d slows down MCTS, *what if we reduce d ?* : → *Tree Collapsing***

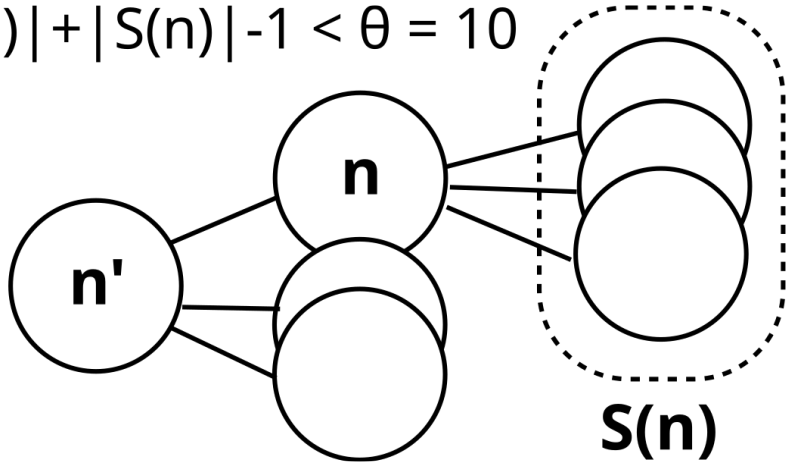


11.1. If a large d slows down MCTS, *what if we reduce d ?* : → *Tree Collapsing*

Hyperparameter: θ

When n has too few children …

$|S(n')| + |S(n)| - 1 < \theta = 10$



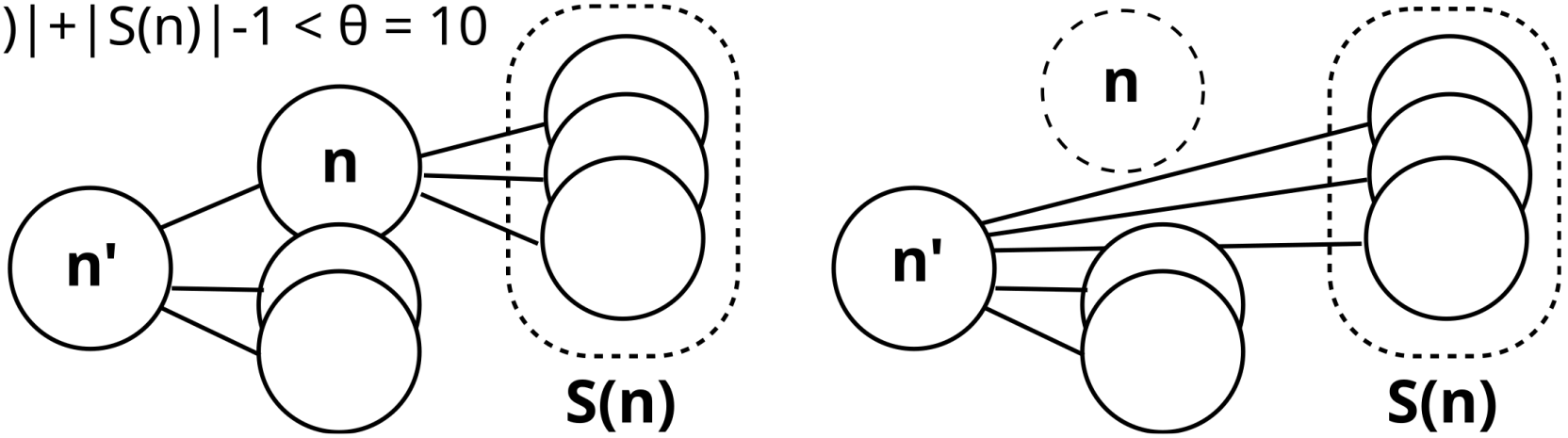
→

11.2. If a large d slows down MCTS, ***what if we reduce*** d ***?*** : → ***Tree Collapsing***

Hyperparameter: θ

When n has too few children … *adopt them to the grandparent n'*

$|S(n')| + |S(n)| - 1 < \theta = 10$



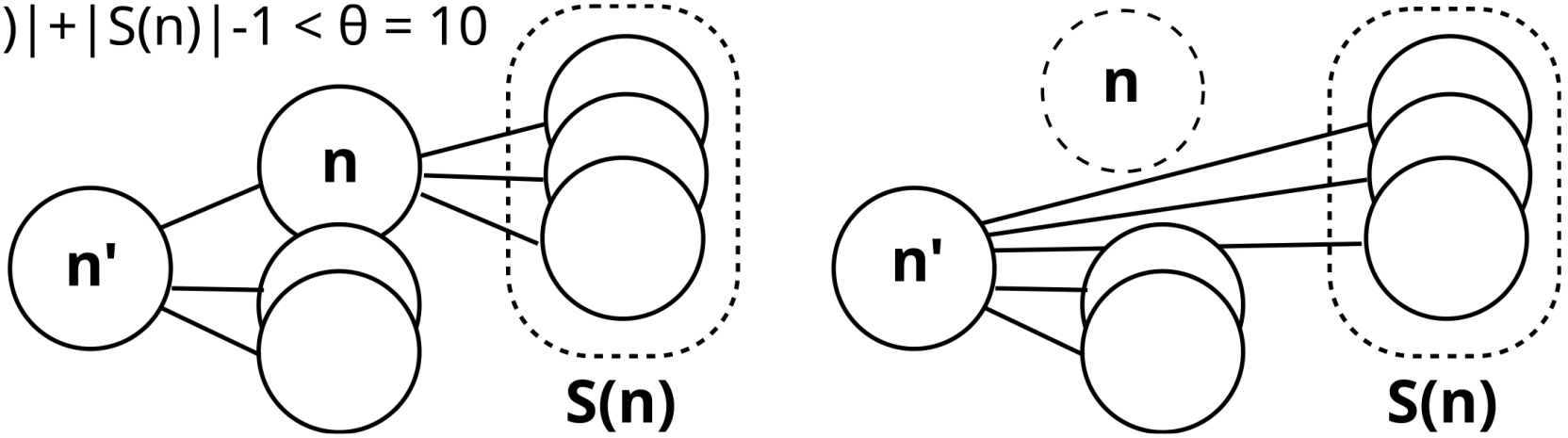
→

11.3. If a large d slows down MCTS, ***what if we reduce*** d ***?*** : → ***Tree Collapsing***

Hyperparameter: θ

When n has too few children … *adopt them to the grandparent n'*

$|S(n')| + |S(n)| - 1 < \theta = 10$



→ ***We also made*** θ ***dynamic*** ($\theta = d$)

11.4. If a large d slows down MCTS, *what if we reduce d ?* : → *Tree Collapsing*

Number of instances solved (5 min)

	no TC	$\theta = 10$	20	40	80	160	*Dynamic*
cea/18	53.8	56.4	**61.4**	58.6	58.8	57.8	61
cea/23	30.8	32	32.2	34.2	33	**34.4**	33.4
cg/18	56.2	58.4	59.6	**60.2**	59.8	56.2	57.6
cg/23	41.6	41.6	42	45.8	45.4	43.4	**46.4**
ff/18	87.4	87.2	88.6	90.4	90.6	**91.4**	90.2
ff/23	58.2	56.8	57.8	58	56.2	55	**58.4**
total	328	↗332.4	↗341.6	↗ **347.2**	↘343.8	↘338.2	**347**

Dynamic: Not always the best ($\theta = 40$) but is **robust**

12. N *ε* bula = Bilevel MCTS + LAMA + BFWS

N *ε* bula : ***3*** **best practices** in a single planner

Search algorithm : Bilevel MCTS + dynamic tree collapsing

Heuristics : LAMA (2010)'s boosted alt queue + h^{FF} + h^{LMcount}

State-based Exploration : Prioritize states with Novelty Metric (2017)

Codebase : Fast Downward

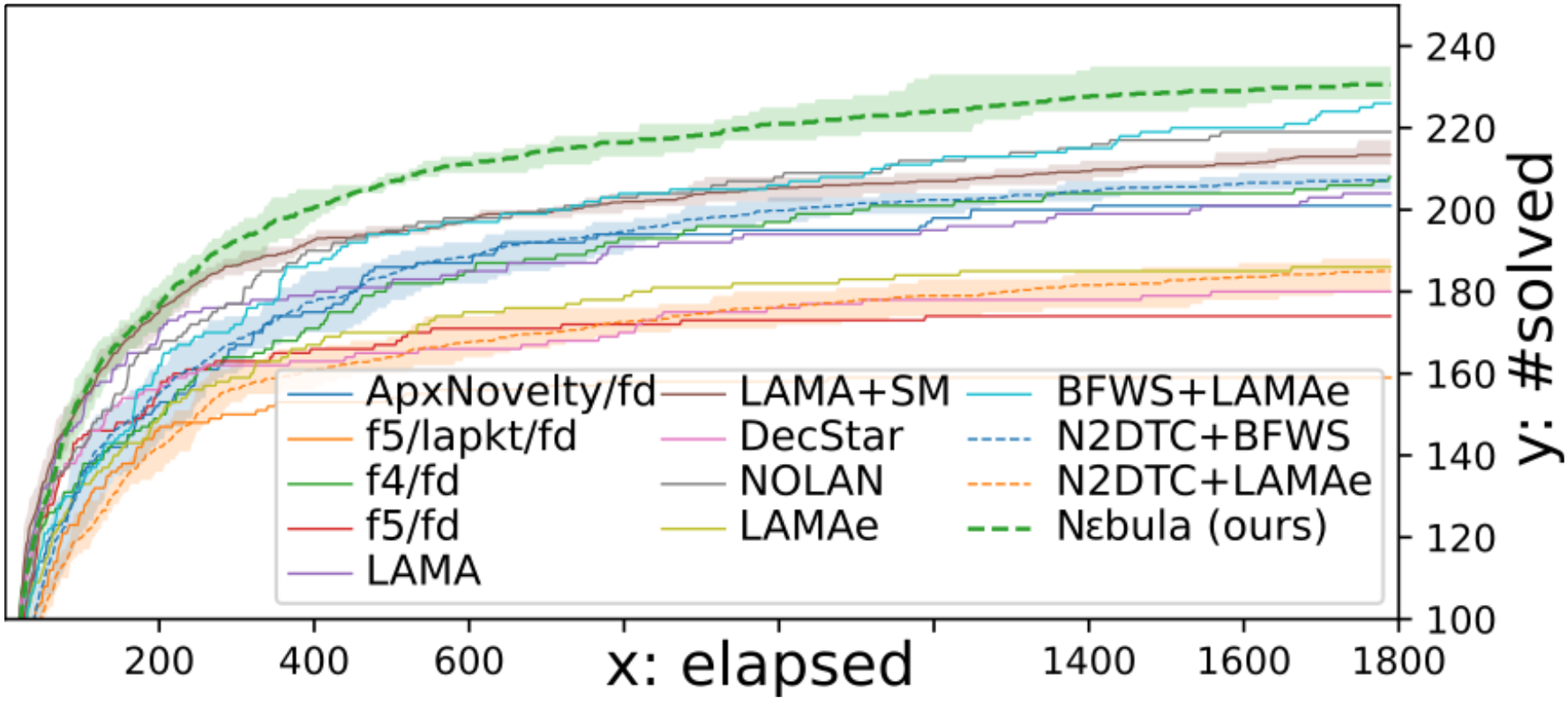
Eager search : Laziness hurts this MCTS so far

12.1. Results

IPC18+IPC23 with 5 min and 30 min limits	LAMA	SM-Type -LAMA	Dec Star	NO LAN	Nε bula
solved instances $(T = 300)$	177	186.4	162	177	**192.2**
IPC Agile score $(T = 300)$	95.4	99.3	93.7	92.8	**99.4**
solved instances $(T = 1800)$	204	213.4	180	219	**230.6**
IPC Agile score $(T = 1800)$	117.6	123.4	112.0	118.7	**126.8**

$$\text{IPC Agile score} = \sum_{i} \min\{1, \log t_i / \log T\}$$

12.2. Runtime Histogram



13. Conclusion

Recursive selection is a bottleneck in MCTS (causes $O(\log N)$)

- → Large depth d hurts
- → Bilevel MCTS: Amortize the bottleneck with a **budgeted BFS**
  - → $O(\log \log N)$ or $O(1)$
- → Tree Collapsing: Reduce the depth d

N ε bula : SotA by combining MCTS+LAMA+BFWS