ニューロシンボリック Classical Planning

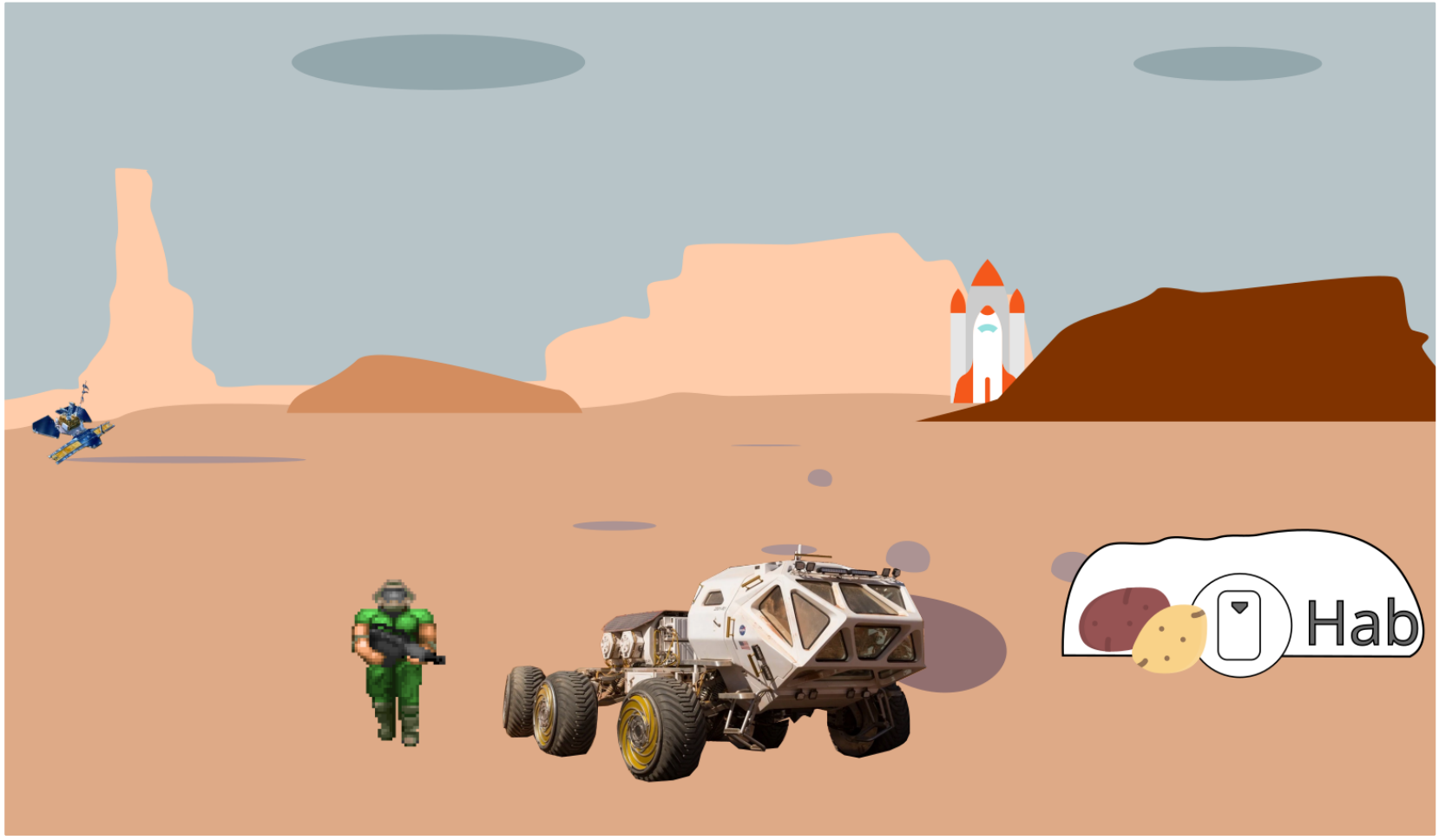
Masataro Asai
(MIT-IBM Watson AI Lab, IBM Research, Cambridge)

質問はZoomに書き込み

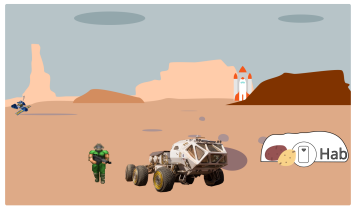
1. **Part 1:**

記号接地と
グラフィカルモデル

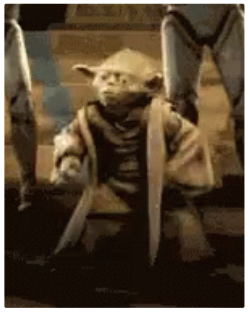
2. 火星に取り残された! うわ やっべ どうしよう?



3. どっちの「知性」がいいですか?



経験に基づく勘だけ で
どうにかなる?
"考えるな、感じろ" で
うまく行きますか?



Reflex (反射) agent

ちょっと **頭を使って** 臨機
応変に対応しますか?

Deliberative (熟考) agent

→ 私がやりたいのは **熟考エージェント**
しかも、 生のセンサー入力(画像)を直接使えるやつ

4. 古典プランナ (古典自動計画ソルバ): 8パズルをミリ秒単位で解ける.

記号的探索・推論システムは爆速

	6	8
7	3	2
5	1	4

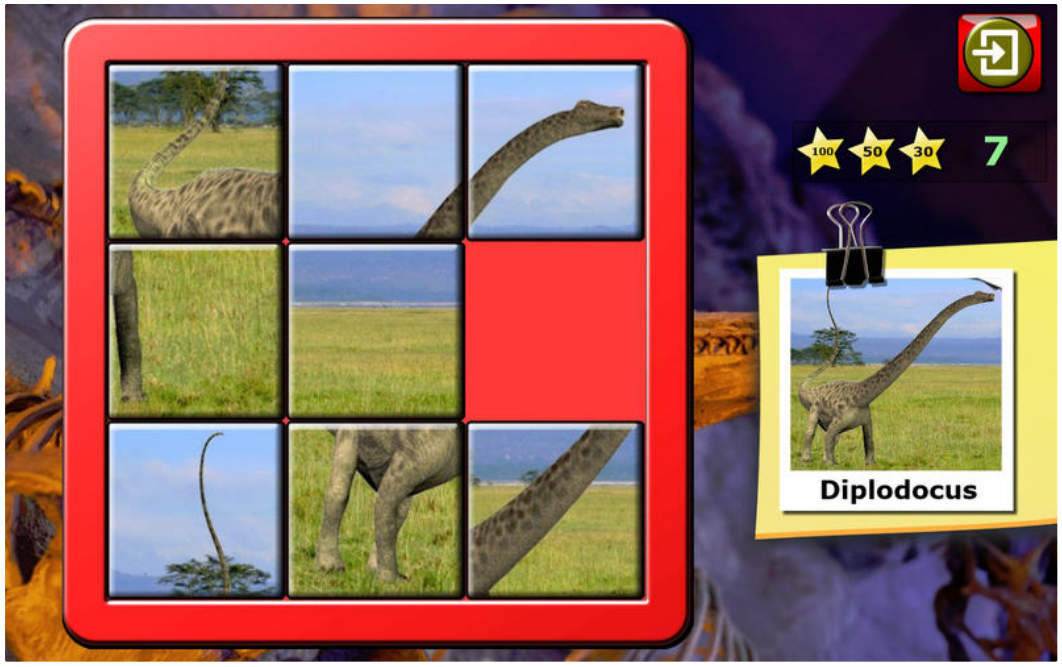
1	2	3
4	5	6
7	8	

*でも **PDDL** があるときにしか 適用不可能.*

```
(:action        move-up ...
 :precondition (and       (empty ?x ?y-old) ...)
 :effects      (and (not (empty ?x ?y-old))...))
```

PDDL : Planning Domain Description Language

5. 弱点: ***画像ベース入力*** になると適用不可能



なぜなら
プランナは
実行するのに
PDDL
モデルが
***必要*!**

Latplan (AAAI18, ICAPS19, IJCAI20, JAIR 22) はこの問題を解決
画像に対する事前知識なし (画像に対するアノテーションなど)
人に由来するラベル・記号は一切なし : "タイルが9つある", "動く"

6. Latplan の目標: *知識獲得ボトルネックの解消*

状態遷移の観測データ → **PDDL Model**


```
(:action          move-up
 :precondition
 (and       (empty ?x ?y-old) ..
 :effects
 (and (not (empty ?x ?y-old)).
```

Latplan が自動化すべき作業は **2つ** あり、これらは別のタスク:

1. 記号生成:

シンボル **= PDDL内で使われる語彙**

Types	Examples
オブジェクト	**panel**$_7$, **x**$_0$, **y**$_0$ …
述語	(**empty** ?x ?y)
命題	**p**$_{28}$ = (empty x_0 y_0)
アクション	(**slide-up** $panel_7$ x_0 y_1)

2. アクションモデル獲得:

状態遷移ルールをシンボルを用いて表現すること

When $Empty(x, y_{old}) \wedge \ldots$;

Then $\neg Empty(x, y_{old}) \wedge \ldots$

7. Latplan は PDDL を自動生成 (IJCAI20, JAIR22)

15パズル画像から自動生成したPDDL

```
(define (domain latent)
 (:predicates (z0) (z1) ... (z199))
 (:action a1 :parameters ()
  :precondition (and (not (z16)) (z36) ... ))
  :effect       (and (z60) (not (z87)) ... ))
 (:action a4 ...)
 ...
 (:action a398 ...))
(define (problem problem-2020-1-13-9-53-6)
  (:init (z5) ... (z199))
  (:goal (and (z3) (not (z7)) ... (z199))))
```

太字: 生成**されたシンボル**

7.1. Latplan は PDDL を自動生成 (IJCAI20, JAIR22)

15パズル画像から自動生成したPDDL

```
(define (domain latent)
 (:predicates (z0) (z1) ... (z199))
 (:action a1 :parameters ()
  :precondition (and (not (z16)) (z36) ... ))
  :effect       (and (z60) (not (z87)) ... ))
 (:action a4 ...)
 ...
 (:action a398 ...))
(define (problem problem-2020-1-13-9-53-6)
  (:init (z5) ... (z199))
  (:goal (and (z3) (not (z7)) ... (z199))))
```

太字: 生成**されたシンボル**

ソルバの返した
アクション列(プラン)

```
(a4)
(a37)
(a10)
(a7)
(a109)
...
(a15)
```

7.2. Latplan は PDDL を自動生成 (IJCAI20, JAIR22)

15パズル画像から自動生成したPDDL

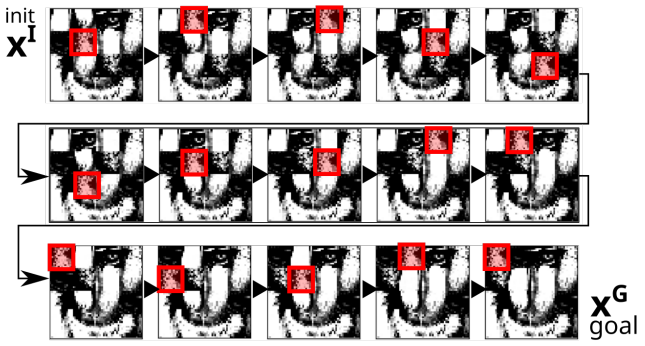
```
(define (domain latent)
 (:predicates (z0) (z1) ... (z199))
 (:action a1 :parameters ()
  :precondition (and (not (z16)) (z36) ... ))
  :effect       (and (z60) (not (z87)) ... ))
 (:action a4 ...)
 ...
 (:action a398 ...))
(define (problem problem-2020-1-13-9-53-6)
  (:init (z5) ... (z199))
  (:goal (and (z3) (not (z7)) ... (z199))))
```

太字: 生成**されたシンボル**

ソルバの返した
アクション列(プラン)

(a4)
(a37)
(a10)
(a7)
(a109)
...
(a15)

左の解をNNデコーダで可視化したもの



Red tile = movable tile↑

Latplan が生成・接地するシンボルは すべて **無名シンボル** (gensym)

人間の使う 自然言語シンボル は扱わない

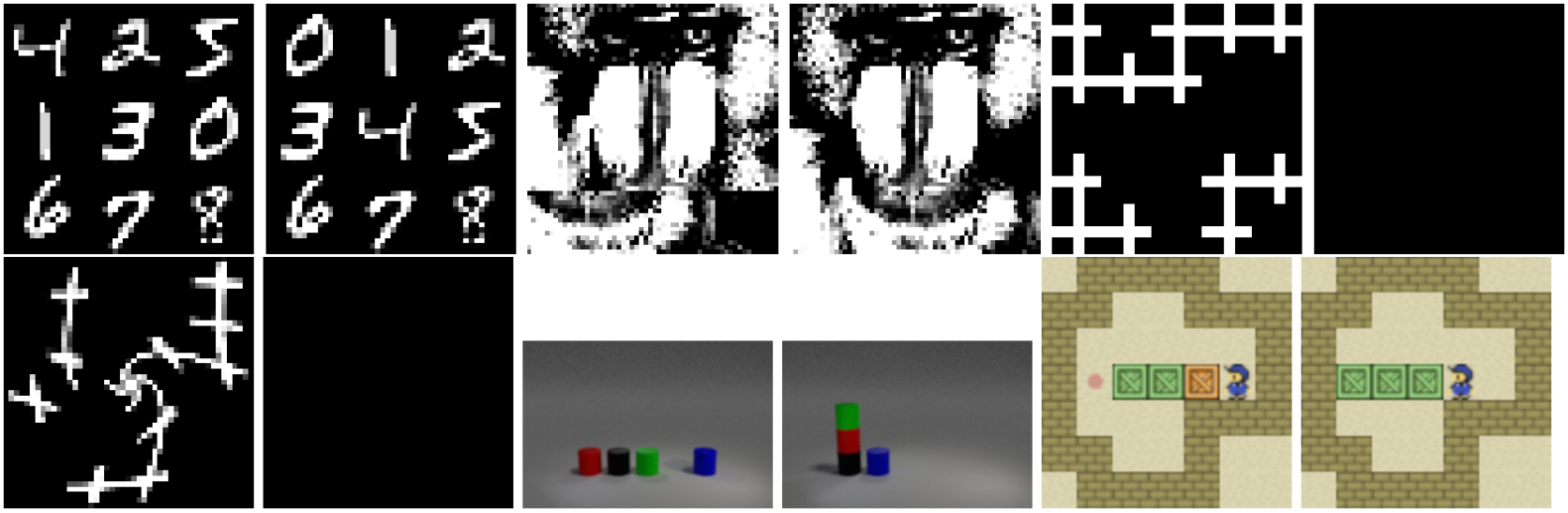
自然言語シンボルへの接地は **寄生的** 記号接地(*) とよばれる

理由: 人間の作った世界の離散化に依存しているから (例: **虹🌈は7色?**)

(*) Taddeo, Mariarosaria, and Luciano Floridi. "Solving the symbol grounding problem: a critical review of fifteen years of research." Journal of Experimental & Theoretical Artificial Intelligence 17.4 (2005): 419-445.

Created: 2024-09-06 Fri 14:46

8. Latplan が解けるパズル/プランニング問題の例



8-puzzle, 15-puzzle, Lights-Out, Twisted Lights-out, **Blocksworld**, Sokoban

8.1. 成功率 (1ドメインあたり40タスク)

Cube-Space AE (IJCAI20)

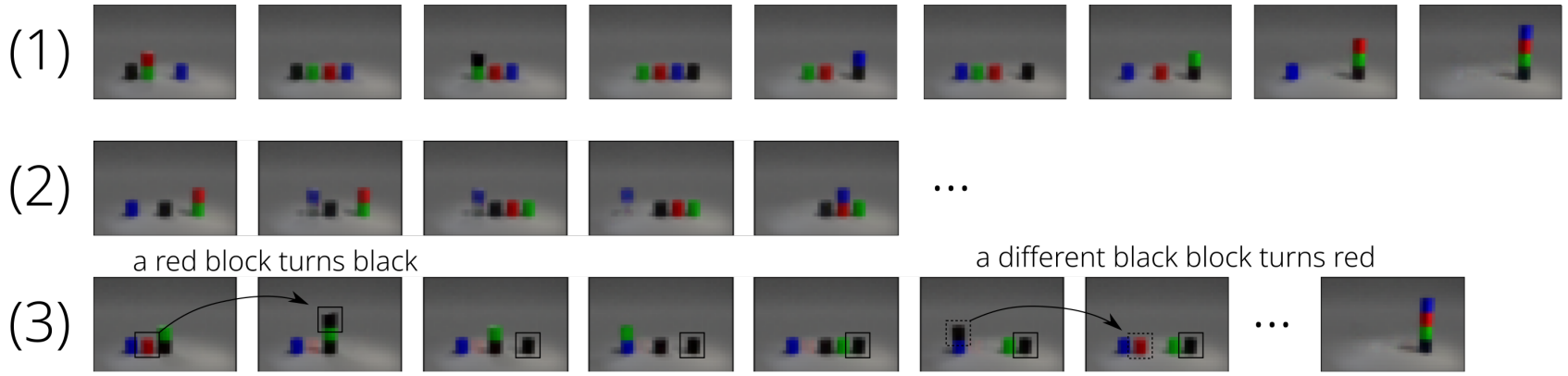
	found	valid	optimal
Blocks	1	1	-
LOut	40	40	35
Twisted	40	40	36
Mandrill	**38**	**38**	20
MNIST	39	39	5
Random MNIST	**39**	**39**	4
Sokoban	14	14	12
Total	211	211	112

Bidirectional Cube-Space AE (JAIR22)

	found	valid	optimal
Blocks	**33**	**32**	-
LOut	40	40	**40**
Twisted	40	40	**40**
Mandrill	25	23	**23**
MNIST	**40**	39	**6**
Random MNIST	36	34	**11**
Sokoban	**40**	**39**	**38**
Total	**254**	**247**	**158**

Significantly better & accurate

8.2. 成功例 / 失敗例


(1)
(2)
...
a red block turns black
a different black block turns red
(3)
...

9. このシステムの肝: 生成モデリング + ***ちゃんと古典プランニングのことを勉強すること***

Latplan は PDDLと互換性のあるアクションモデルを画像から学習する.

「PDDLと互換性がある」ってどういう意味？

アクション: $a = \mathrm{Eat}(\mathrm{pizza}🍕)$

前提条件: $\mathrm{pre}(a) = \neg\mathrm{full}() \wedge \mathrm{have}(\mathrm{pizza}🍕)$ – 空腹かつピザがある

効果: **削除効果:** $\mathrm{del}(a) = \{\mathrm{have}(\mathrm{pizza}🍕)\}$ – ピザはなくなって

効果: **追加効果:** $\mathrm{add}(a) = \{\mathrm{full}()\}$ – 満腹になる

状態遷移ルール: $z_{t+1} = (z_t \setminus \mathrm{del}(a)\ ) \cup \mathrm{add}(a)$

効果 $\mathrm{del}(a), \mathrm{add}(a)$ は z_t **に依存しない.**

よくある世界モデル は **そこをわかってない！** (自動計画の人がいない)

よくある世界モデル: $z_{t+1} =$ なぞのニューラルネット(z_t, a)

PDDLに変換できない → ***高速なソルバを使えない***

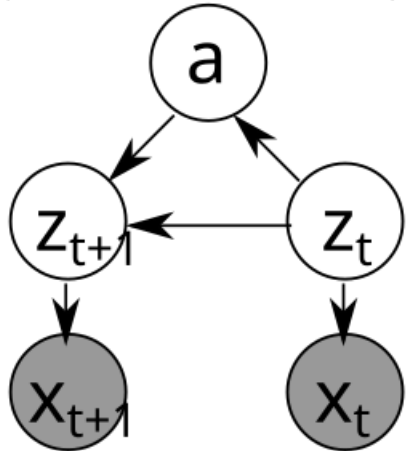
10. 高速ソルバのためにPDDL記号モデルを学習

→ 言語の制約を ***NNモデルに埋め込まないといけない***

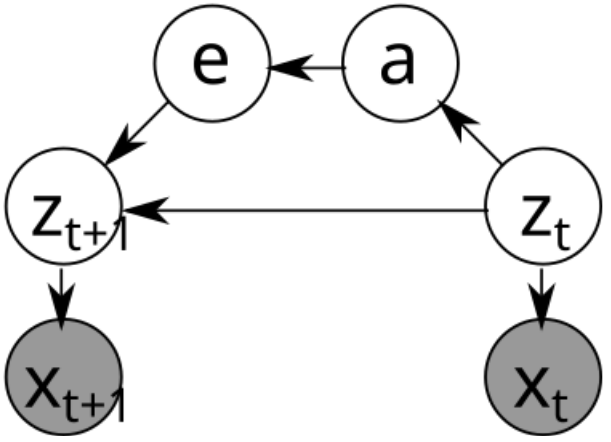
PDDLでは、アクションの効果と状態 z_t は独立している.

→ z_t と独立な変数 **e** を導入する必要あり。

よくある世界モデル
(PDDLと互換性なし)

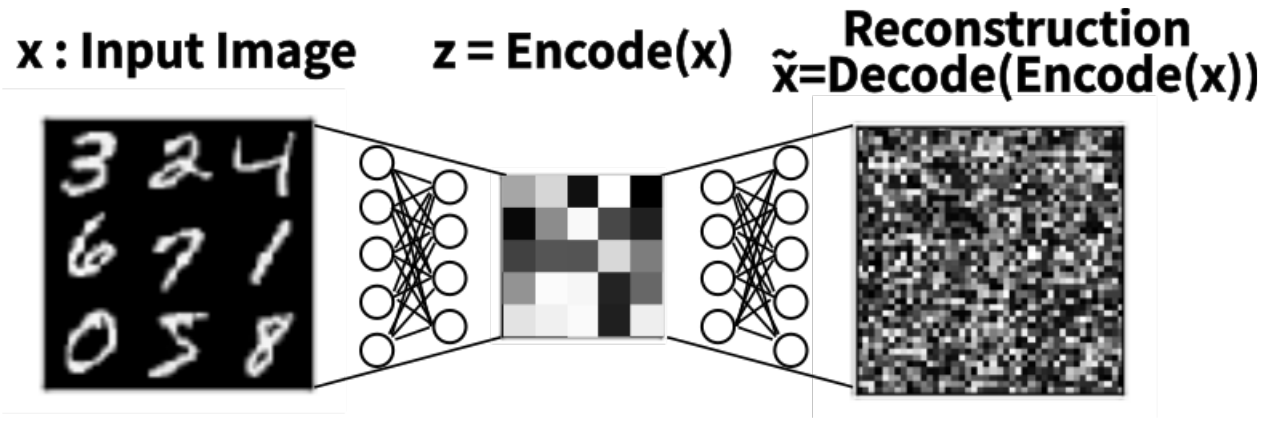


Cube-Space AE
(学習後PDDLに変換可能)

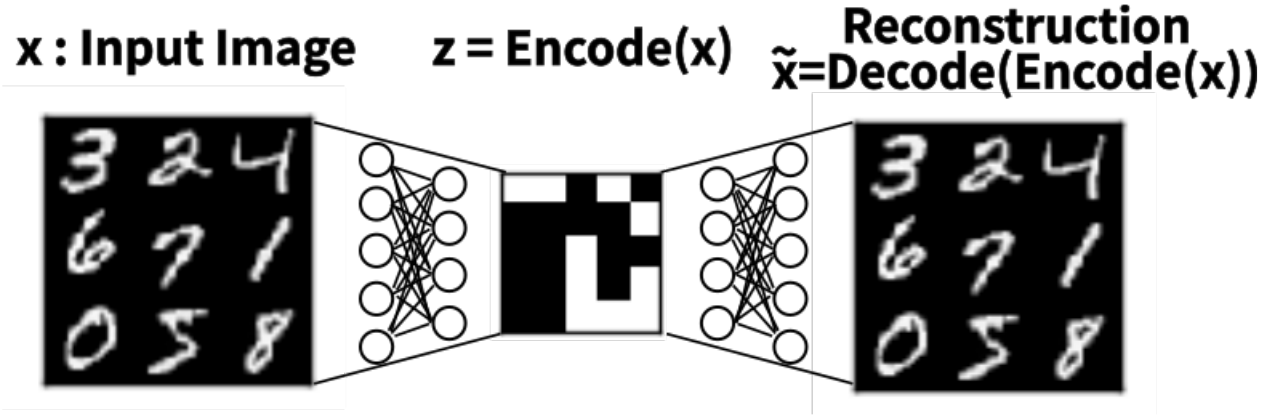


これを **離散的隠れ変数** (Gumbel-Softmax) + **変分学習** で訓練。
グラフィカルモデルが数学的根拠を与える

11. 離散表現学習: Gumbel-Softmax VAE ほか



11.1. 離散表現学習: Gumbel-Softmax VAE ほか

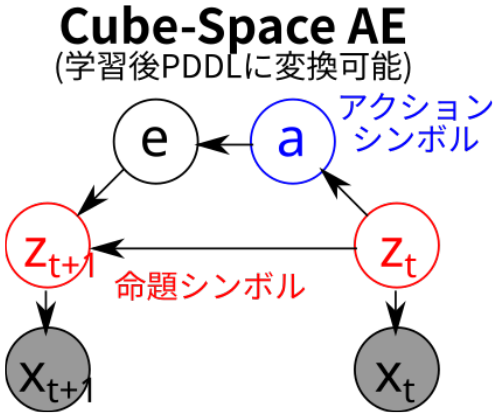


2017年ぐらい以降、多数の手法が考案されている
(REBAR, RELAX, VQVAE, SQVAE, …)

ニューロシンボリックをするなら要チェック!

12. シンボル は一種類だけではない

命題シンボル (handempty, …) vs. ***アクションシンボル*** (pickup,…).



これらは **別の名前空間に存在する**.

自然言語シンボルも同様:

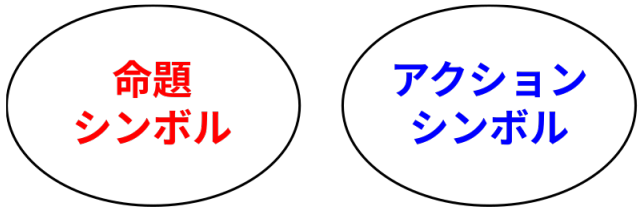
I ***like*** an apple : 動詞

Thanks for the ***like*** on Facebook! : 名詞

Common Lisp (Lisp-2) vs. Scheme (Lisp-1)

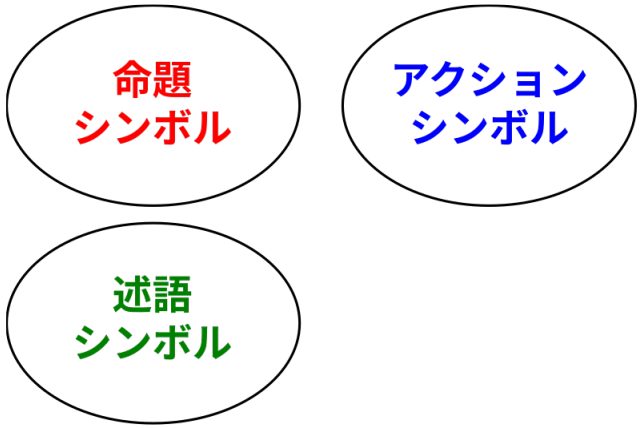
→ **別の名前空間** にあるシンボルは
別の言語制約が必要。

13. PDDL: 6つの名前空間, 6つのメカニズム



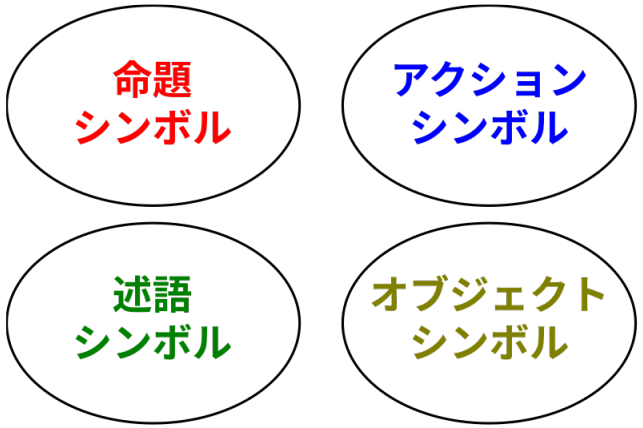

```
(define (domain           )
 (:predicates (handempty)          ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem          )
  (:objects      ...)
  (:init ...)
  (:goal ...))
```

13.1. PDDL: 6つの名前空間, 6つのメカニズム




```
(define (domain           )
 (:predicates (handempty) (on ?x ?y)  ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem          )
  (:objects      ...)
  (:init ...)
  (:goal ...))
```

13.2. PDDL: 6つの名前空間, 6つのメカニズム




```
(define (domain           )
 (:predicates (handempty) (on ?x ?y)  ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem          )
  (:objects A B C ...)
  (:init ...)
  (:goal ...))
```

13.3. PDDL: 6つの名前空間, 6つのメカニズム


```
(define (domain              )
 (:predicates (handempty) (on ?x ?y)  ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem world-1986)
  (:objects A B C ...)
  (:init ...)
  (:goal ...))
```

問題シンボル: 「マリオのステージを表すシンボル」→ 別の状態空間を生成

Stage 1-1, Stage 1-2, …

13.4. PDDL: 6つの名前空間, 6つのメカニズム


```
(define (domain blocksworld)
 (:predicates (handempty) (on ?x ?y)  ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem world-1986 )
  (:objects A B C ...)
  (:init ...)
  (:goal ...))
```

問題シンボル: 「マリオのステージを表すシンボル」 → 別の状態空間を生成

Stage 1-1, Stage 1-2, …

ドメインシンボル: 「別のゲームを表すシンボル」 → 別のアクションを生成

マリオ、ゼルダ、ドンキーコング、F-Zero …

13.5. PDDL: 6つの名前空間, 6つのメカニズム


```
(define (domain blocksworld)
 (:predicates (handempty) (on ?x ?y)  ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem world-1986)
  (:objects  A B C ...)
  (:init ...)
  (:goal ...))
```

自律エージェントは、これら全てを
人に頼らず生成できなくてはいけない

問題シンボル: 「マリオのステージを表すシンボル」 → 別の状態空間を生成

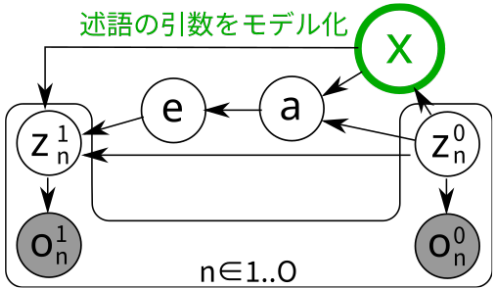
Stage 1-1, Stage 1-2, …

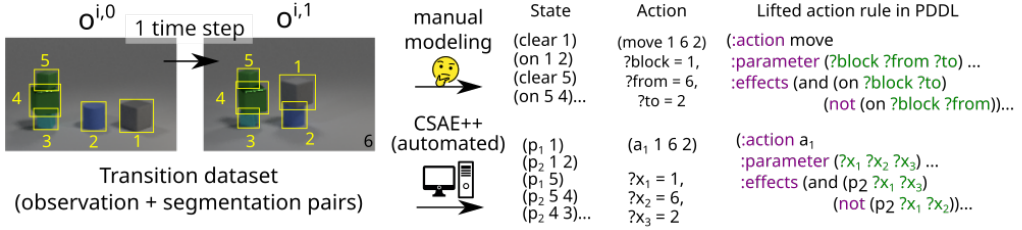
ドメインシンボル: 「別のゲームを表すシンボル」 → 別のアクションを生成

マリオ、ゼルダ、ドンキーコング、F-Zero …

14. 実際 Latplan を述語論理に拡張可 (ICLR2021 workshop)

Latplan/FOSAE++：画像内の異なるオブジェクトに使えるアクションを学習





追加制約: Successor State Axiom (非束縛変数の禁止 [Reiter, 1991])

アクション: $a = \mathrm{Eat}(\mathrm{pizza}🍕)$ ：引数 $\mathrm{pizza}🍕$

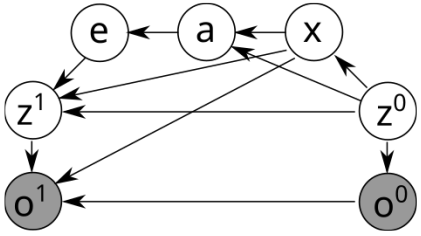
効果: *発表(フロム, アーマードコアの新作)*

↑ $\mathrm{pizza}🍕$ を使わないので禁止

新たな制約が **述語シンボル** のグラウンディングを可能にした

15. Extending the generative model (unpublished)

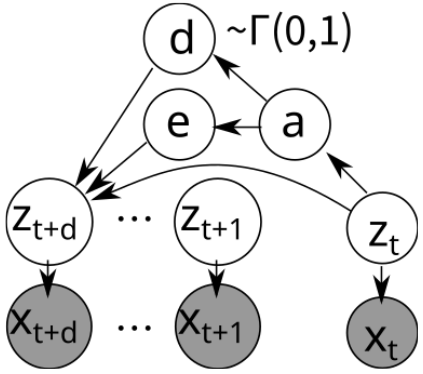
Lifted actions,
Parameters x



Predicates, too.

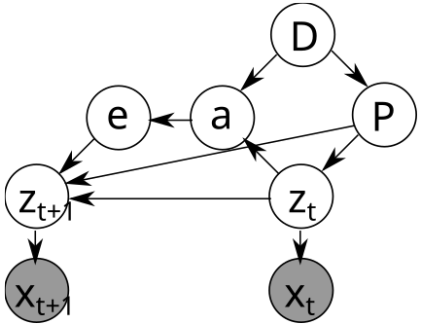
Compatible with First Order Logic.

Temporal actions,
Action durations d



Compatible with LTL.

Domain symbol D,
Problem symbol P

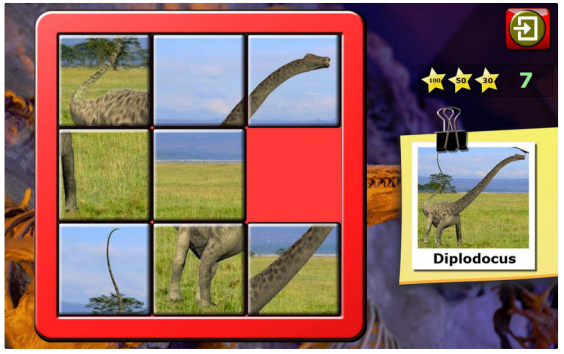


Allows mixed-domain training?

16. Part 1: Conclusion

Goal: 理性的な実世界エージェント

Latplan で記号接地



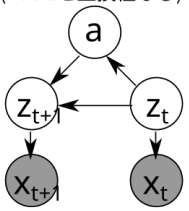
異なるシンボルは **別々の制約が必要**


```
(define (domain blocksworld)
 (:predicates (handempty) (on ?x ?y)  ...)
 (:action pick-up
  :precondition ...
  :effect       ...))
(define (problem world-1986)
  (:objects  A B C ...)
  (:init ...)
  (:goal ...))
```

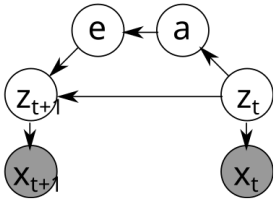
自律エージェントは、これら全てを
人に頼らず生成できなくてはいけない

生成モデルによって **制約を精緻に記述可能**

よくある世界モデル
(PDDLと互換性なし)



Cube-Space AE
(学習後PDDLに変換可能)



記号接地するなら **対象言語を *ちゃんと* 理解して *正しく* モデル化しよう!**

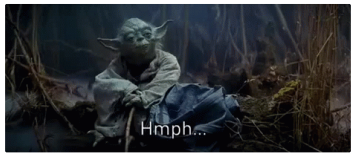
質問: 2つまで

17. Part 2:

理性的な実世界エージェントの アーキテクチャ を考える

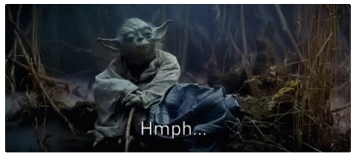
→ **表現学習以外** の プランニング+機械学習

18.　理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

18.1. 理性的な実世界エージェントのアーキテクチャ

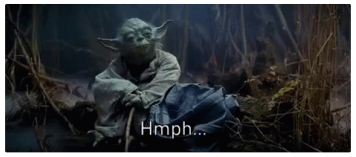


ノイズだらけの実世界観測データ

記号的表現 **& ソルバによる**高速求解

above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
preconditions: handempty, clear(A) ...
effects: ...

18.2. 理性的な実世界エージェントのアーキテクチャ



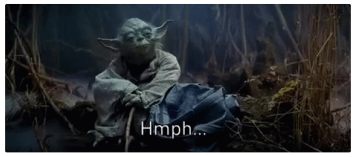
ノイズだらけの実世界観測データ

記号接地 ⇕ **(グラフィカルモデルによる**
教師なし 離散表現 学習**)**

記号的表現 **& ソルバによる**高速求解

above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
preconditions: handempty, clear(A) ...
effects: ...

18.3. 理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

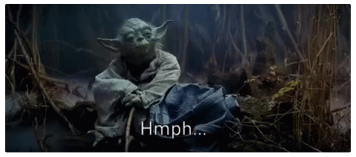
記号接地 ⇕ **(グラフィカルモデルによる**
教師なし 離散表現 学習**)**

記号的表現 **& ソルバによる**高速求解

```
above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
  preconditions: handempty, clear(A) ...
  effects:  ...
```

探索ヒューリスティクス
h(s), V(s), π(a), Q(s, a)

18.4. 理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

記号接地 ⇕ **(グラフィカルモデルによる**
教師なし 離散表現 学習**)**

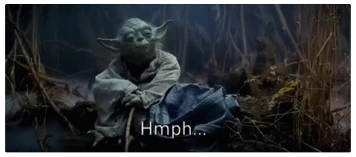
記号的表現 **& ソルバによる**高速求解

```
above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
  preconditions: handempty, clear(A) ...
  effects:  ...
```

探索ヒューリスティクス
h(s), V(s), π(a), Q(s, a)

探索アルゴリズム
A*, GBFS, MCTS

18.5. 理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

記号接地 ⇕ **(グラフィカルモデルによる**
教師なし 離散表現 学習)

記号的表現 **& ソルバによる**高速求解

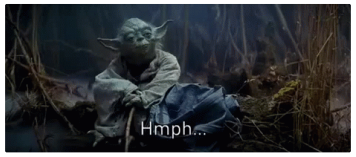
```
above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
  preconditions: handempty, clear(A) ...
  effects:  ...
```

オフライン学習
(Reinforcement Learning ⇕
Un/Semi/Supervised Learning)

探索ヒューリスティクス
h(s), V(s), π(a), Q(s, a)

探索アルゴリズム
A*, GBFS, MCTS

18.6. 理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

記号接地 ⇕ (グラフィカルモデルによる 教師なし 離散表現 学習)

記号的表現 & ソルバによる高速求解

above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
  preconditions: handempty, clear(A) ...
  effects: ...

オフライン学習 (Reinforcement Learning ⇕ Un/Semi/Supervised Learning)

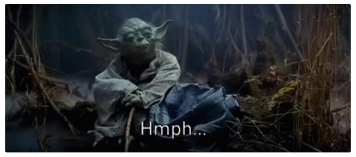
⇕

オンライン学習/枝刈り (Multi-Armed Bandit)

探索ヒューリスティクス
h(s), V(s), π(a), Q(s, a)

探索アルゴリズム
A*, GBFS, MCTS

18.7. 理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

記号接地 ⇕ **(グラフィカルモデルによる 教師なし 離散表現 学習)**

記号的表現 & ソルバによる高速求解

above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
preconditions: handempty, clear(A) ...
effects: ...

オフライン学習 (Reinforcement Learning ⇕ Un/Semi/Supervised Learning)

⇕

オンライン学習/枝刈り (Multi-Armed Bandit)

探索ヒューリスティクス
h(s), V(s), π(a), Q(s, a)

探索アルゴリズム
A*, GBFS, MCTS

19. 探索ヒューリスティクスの学習:

RL (ICAPS 2022)

SL (IJCAI 2024)

20. ICAPS22

Reinforcement Learning for Classical Planning: Viewing Heuristics as Dense Reward Generators

Clement Gehring, Masataro Asai(*),
Rohan Chitnis, Tom Silver,
Leslie Kaelbling, Shirin Sohrabi, Michael Katz

21. Value function learning

SGD Loss function:

$$\frac{1}{|B|}\sum_{s\in B}\frac{1}{2}|V(s)-\mathbb{E}_{a\in A}[Q(a,s)]|^2$$

B : batches in the experience replay

Research Questions:

Q: RLだけで 古典プランニングを解けるか? (A: **不可**)

Q: RLだけで プランニングを高速化できるか? (A: **不可**)

Q: プランニングの道具と適切に組み合わせれば? ***(A: 可)***

Lessons learned: **古典プランニングを甘く見るな**

22. Why Classical Plannning is difficult for RL?

Rewards in Classical Planning:

$$r(s, a, s') = \begin{cases} 1 & \text{if } s' \text{ is a goal,} \\ 0 & \text{otherwise.} \end{cases}$$

This is extremely sparse.

Any non-goal states are **equally** ***worthless***

No guidance for RL until a goal,

which is **difficult to find** in the first place!

23. Potential-Based *Reward Shaping* (PBRS) for Sparse Rewards

Given a **potential function** $\phi(s)$, ***redefine rewards*** as:

$$\hat{r}(s, a, s') = r(s, a, s') + \gamma\phi(s') - \phi(s).$$

Important: PBRS **preserves** the optimal value function

$$V_\gamma^*(s) = \hat{V}_\gamma^*(s) + \phi(s).$$

= learning the residual from $\phi(s)$

Ng, Harada, Russell. "Policy invariance under reward transformations: Theory and application to reward shaping." ICML. Vol. 99. 1999.
Created: 2024-09-06 Fri 14:46

24. Heuristic Function $h(s)$ in classical planning

Distance estimate

$h(s) = 5 \quad \longleftrightarrow \quad s - s_1 - s_2 - s_3 - s_4 - \text{goal}$ (おおよそ)

Many implementations: h^{add} , h^{FF} , h^{CEA} , ...
(Bonet&Geffner 01, Hoffmann 01, Helmert 08) もっと新しいのもある

Heuristics satisfies the PBRS requirements

Let's use $h(s)$ for PBRS!

25. Issue 1: *Discounting*

RL : *discounted rewards* ↔ Planning : ***undiscounted costs***

Our solution:

Training : Convert $h(s) \rightarrow h_\gamma(s)$: ***discounting*** $h(s)$ for PBRS

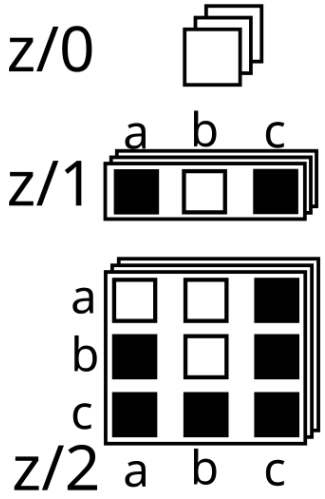
$$\phi(s) = h_\gamma(s) = \sum_{t=1}^{h(s)} \gamma^t \cdot 1 = \frac{1-\gamma^{h(s)}}{1-\gamma}.$$

Planning : Convert $V_\gamma(s) \rightarrow V(s)$: ***undiscounting*** a learned $V_\gamma(s)$

$$V_\gamma(s) = \frac{1-\gamma^{V(s)}}{1-\gamma} \Leftrightarrow V(s) = \log_\gamma(1-(1-\gamma)V_\gamma(s)).$$

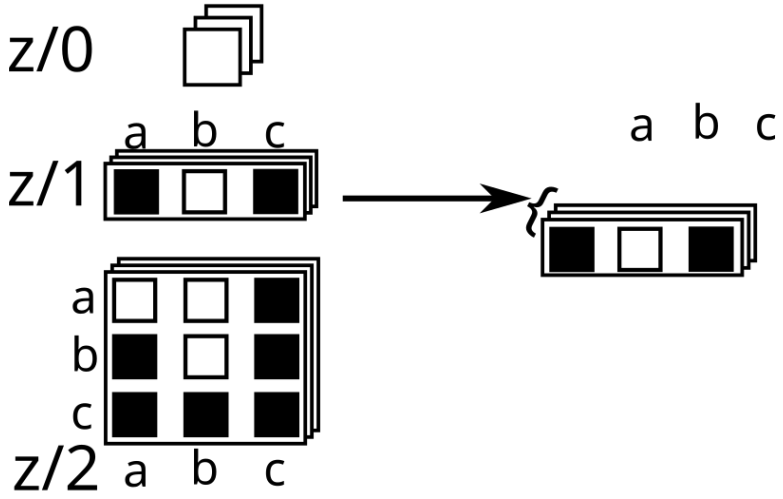
26. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



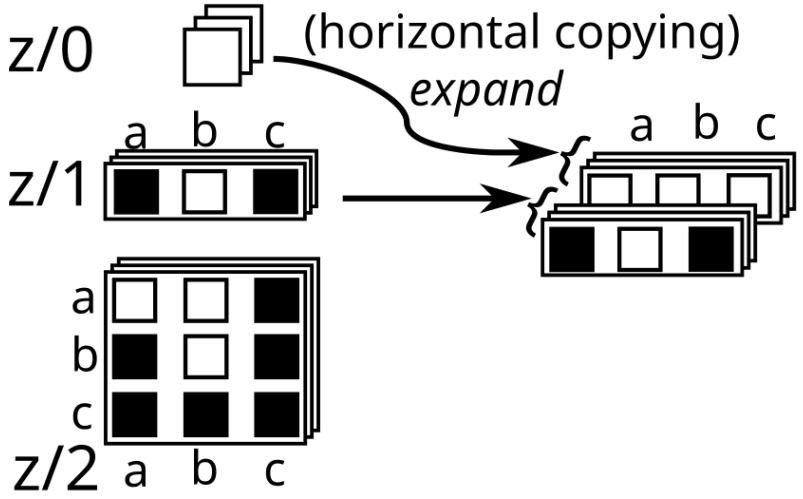
26.1. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



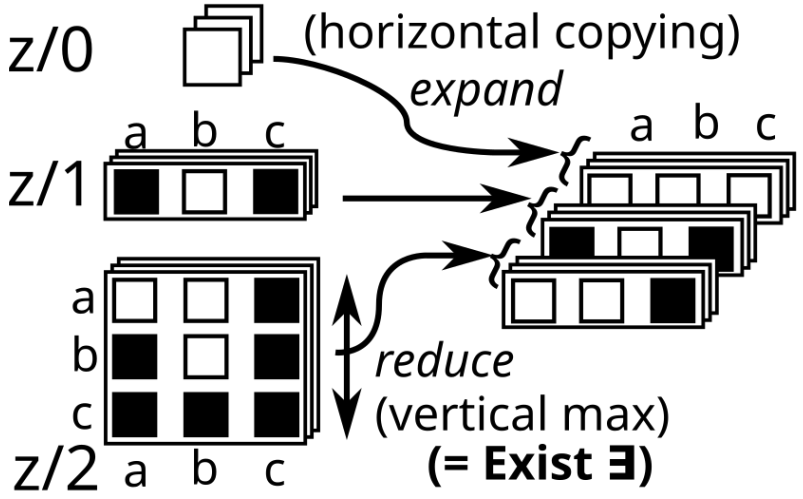
26.2. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



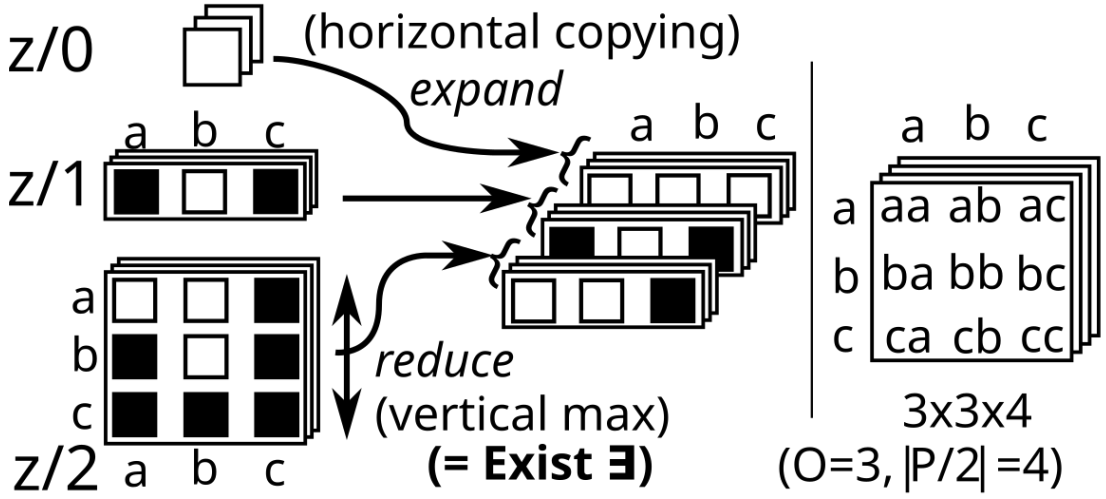
26.3. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



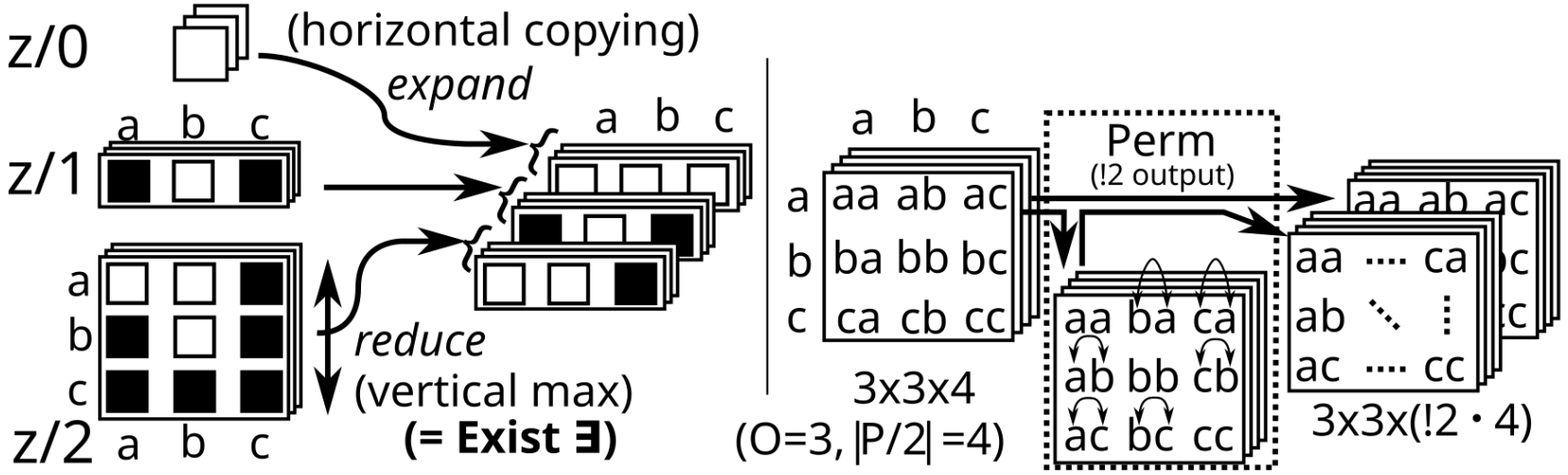
26.4. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



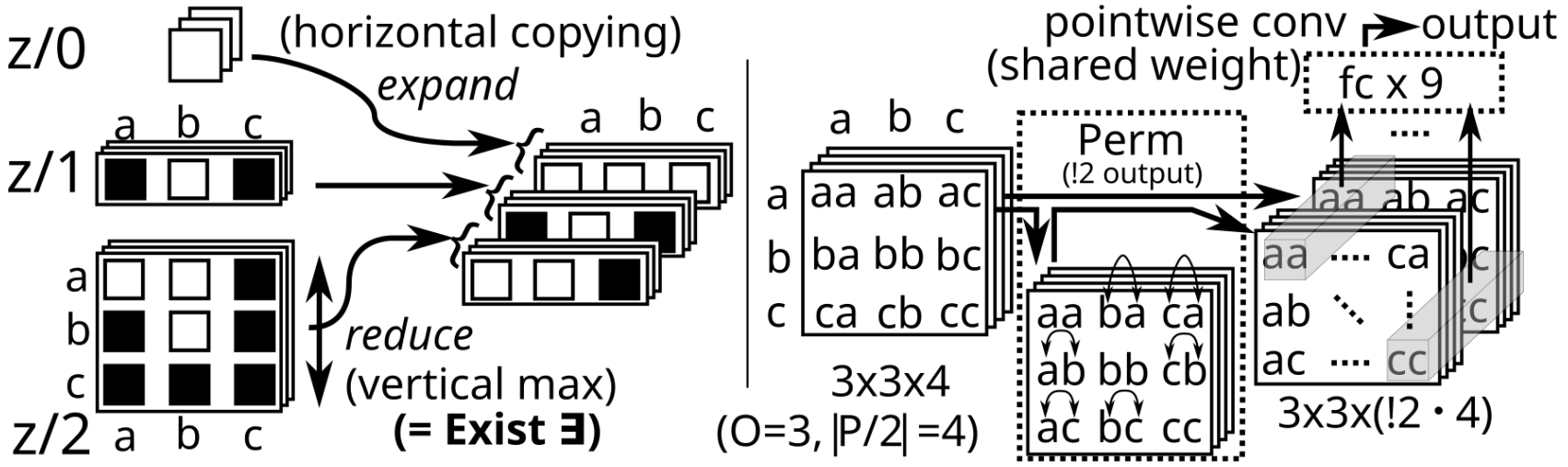
26.5. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



26.6. Issue 2: Handling *First-Order Logic Input*

Our solution: **Neural Logic Machine (NLM)** [Dong et. al. ICLR 2019]



Size & Permutation invariant to FOL binary inputs

Quite useful for various FOL tasks

ILP [Dong et. al. ICLR 2019]

Regression (our work)

27. Results

Skipped

28. ICAPS 2022 : Conclusion

Sparse Rewards

$$r(s) = \begin{cases} 1 & \text{if } s \text{ is a goal,} \\ 0 & \text{otherwise.} \end{cases}$$

ゴールにたどり着くのが
とても難しい

Potential-Based Reward Shaping

$$V^*_\gamma(s) = \hat{V}^*_\gamma(s) + \phi(s)$$

$\phi(s) =$ 距離の見積もり $h(s)$

Sparse Reward を解決

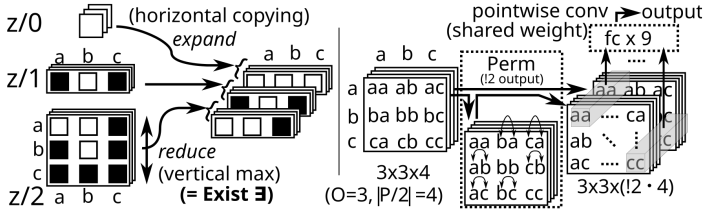
Discounting

$$h_\gamma(s) = \frac{1 - \gamma^{h(s)}}{1 - \gamma}$$

$$V(s) = \log_\gamma(1 - (1 - \gamma)V_\gamma(s))$$

これがないと
$h(s) = \infty$ のとき死ぬ

First-Order Logic Input



異なるサイズの問題に汎化

Q: RLだけで 古典プランニングを解けるか? (A: **不可**)
Q: RLだけで プランニングを高速化できるか? (A: **不可**)
Q: プランニングの道具と適切に組み合わせれば? ***(A: 可)***
Lessons learned: ***古典プランニングを甘く見るな***

29. IJCAI24

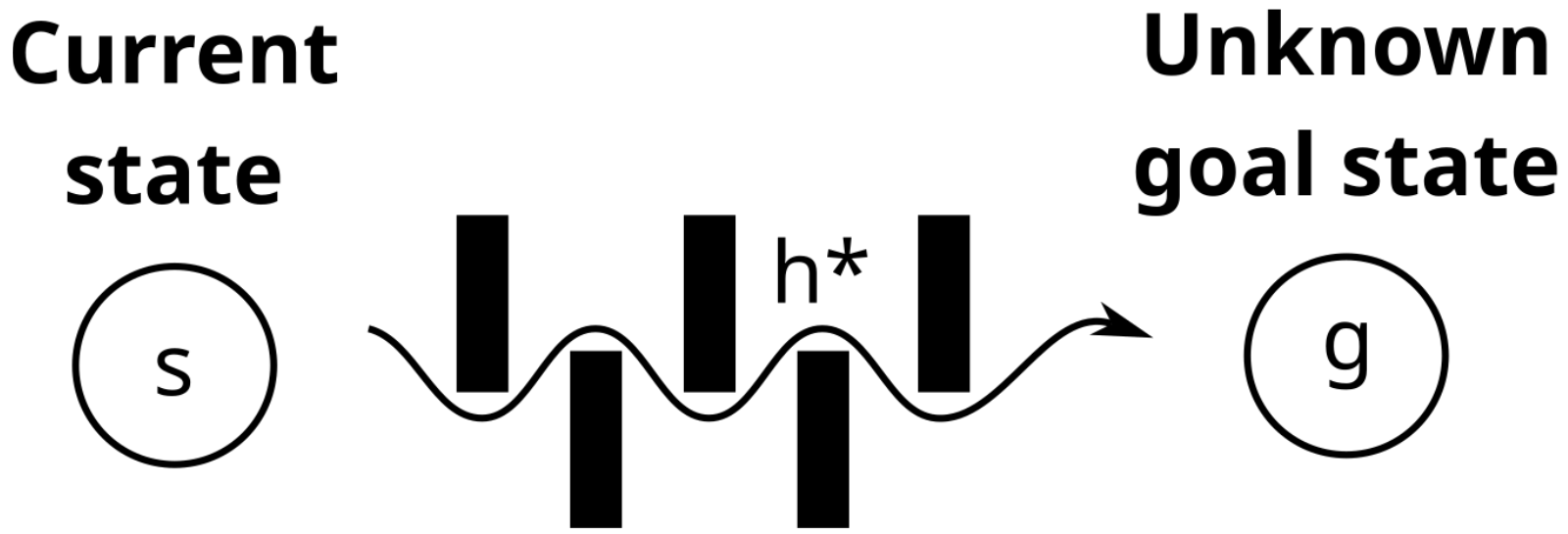
On Using Admissible Bounds for Learning Forward Search Heuristics

Carlos Núñez-Molina*, Masataro Asai*,
Pablo Mesejo, Juan Fernández-Olivares

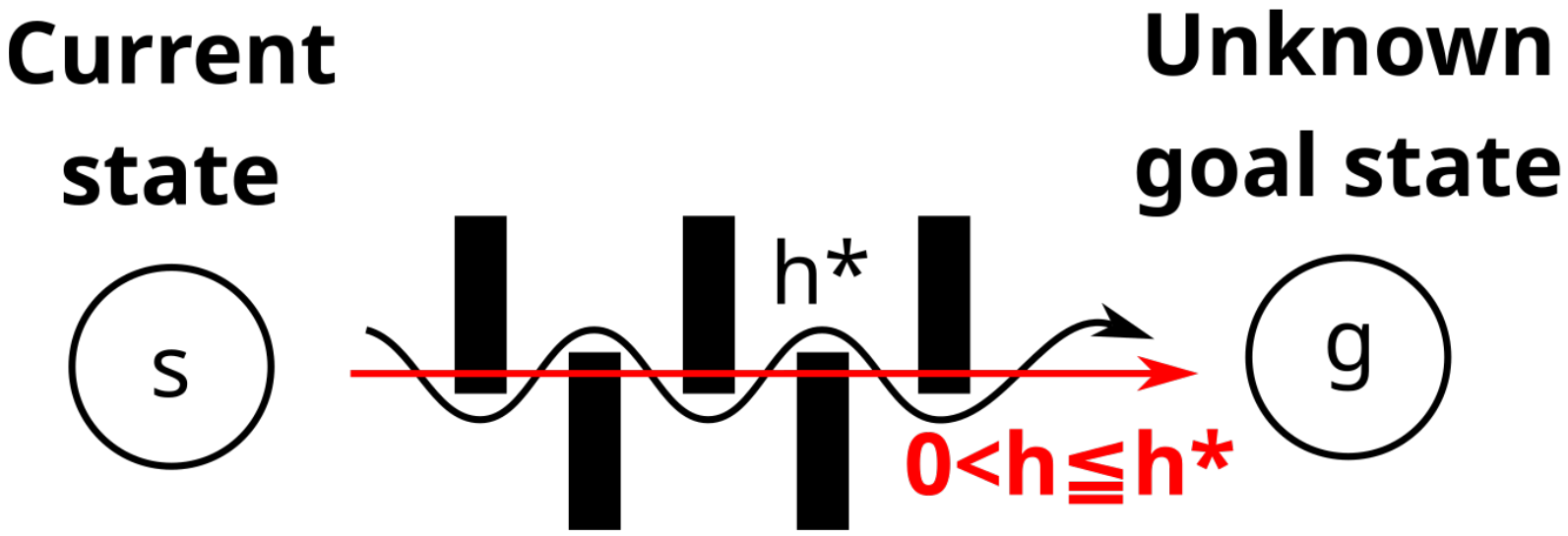
RL 嫌いなんで ... 教師あり学習!

30. Task: Learn the shortest path cost

30.1. Task: Learn the shortest path cost



30.2. Task: Learn the shortest path cost



We know a certain lower bound of h*:

Admissible heuristics h

obtained by solving a **relaxed problem**

Q: Can we exploit the *admissibility* of heuristics in training?

31. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

31.1. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

31.2. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

$-\log p(x)$

31.3. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

$-\log p(x)$

31.4. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

$$-\log p(x)$$

31.5. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

$$-\log p(x) = -\log \mathcal{N}(\mu, \sigma)$$

31.6. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

of a Gaussian distribution

$$-\log p(x) = -\log \mathcal{N}(\mu, \sigma)$$

31.7. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

of a Gaussian distribution

$$-\log p(x) = -\log \mathcal{N}(\mu, \sigma)$$

$$= \frac{(x-\mu)^2}{2\sigma^2} + \log \sqrt{2\pi\sigma^2}.$$

31.8. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

of a Gaussian distribution

$$-\log p(x) = -\log \mathcal{N}(\mu, \sigma)$$

$$= \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2}.$$

which contains the **square error**

31.9. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

of a Gaussian distribution

$$-\log p(x) = -\log \mathcal{N}(\mu, \sigma)$$

$$= \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2}.$$

which contains the **square error**

It is a special case because

it sets an arbitrary fixed σ=$\frac{1}{\sqrt{2}}$

31.10. Stop using (Mean) Square Errors!

x : data point (optimal solution cost, x=h*)

μ : prediction (from **s**, **g**)

$(x-\mu)^2$ **: Square error**

because it is just a special case of

Negative Log Likelihood (NLL)

of a Gaussian distribution

$$-\log p(x) = -\log \mathcal{N}(\mu, \sigma) = \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2}.$$

which contains the **square error**

It is a special case because

it sets an arbitrary fixed σ=$\frac{1}{\sqrt{2}}$

$$-\log \mathcal{N}(\mu, \frac{1}{\sqrt{2}}) = \frac{(x-\mu)^2}{2\cdot\frac{1}{2}} + \log\sqrt{2\pi\cdot\frac{1}{2}}$$

$$= (x-\mu)^2 + \text{Const.}$$

Choosing a loss function
⇔ Choosing a distribution

Choosing a loss function
⇔ Choosing a distribution

Shoutout: don't "innovate" a new loss!

Innovate a **distribution** instead!

Choosing a loss function
⇔ Choosing a distribution

Shoutout: don't "innovate" a new loss!

Innovate a **distribution** instead!

(Mean) Squared error ⇔ **Gaussian** NLL

$$-\log \mathcal{N}(\mu, \sigma) = \frac{(x-\mu)^2}{2\sigma^2} + \log \sqrt{2\pi\sigma^2}$$

Choosing a loss function
⇔ Choosing a distribution

Shoutout: don't "innovate" a new loss!

Innovate a **distribution** instead!

(Mean) Squared error ⇔ **Gaussian** NLL

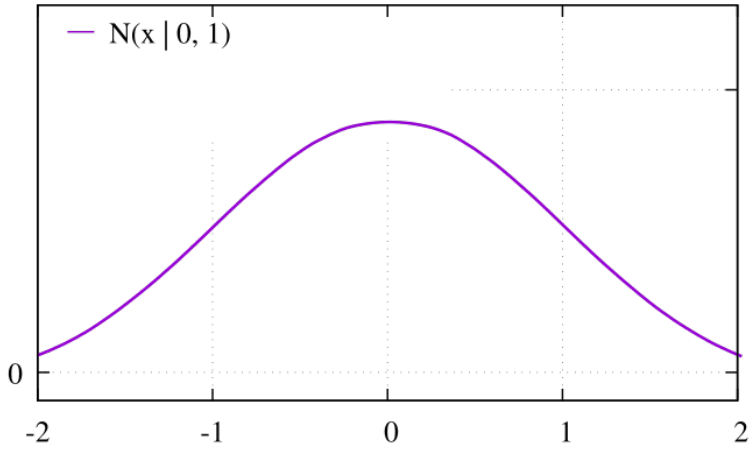
$$-\log \mathcal{N}(\mu, \sigma) = \frac{(x-\mu)^2}{2\sigma^2} + \log \sqrt{2\pi\sigma^2}$$

(Mean) Absolute error ⇔ **Laplacian** NLL

$$-\log \mathcal{L}(\mu, b) = \frac{|x-\mu|}{b} + \log 2b$$

33. Thinking with Distributions

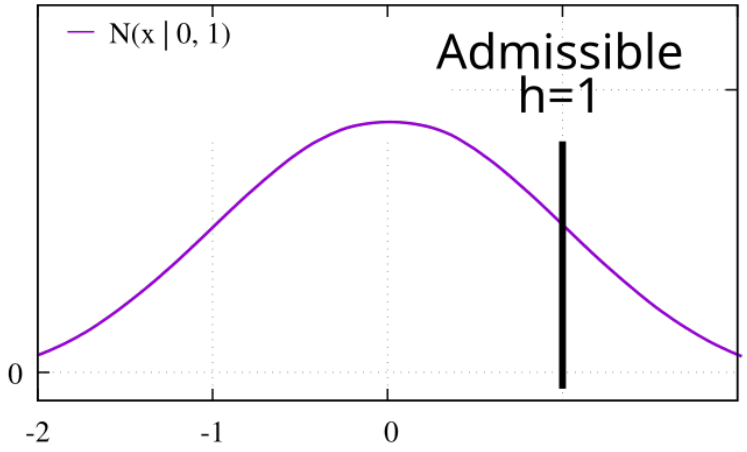
Is $\mathcal{N}(\mu, \sigma)$ the correct distribution for h^* ?



N(μ,σ) assigns non-zero probability to *all values* x ∈ R

33.1. Thinking with Distributions

Is $\mathcal{N}(\mu, \sigma)$ the correct distribution for h^* ?

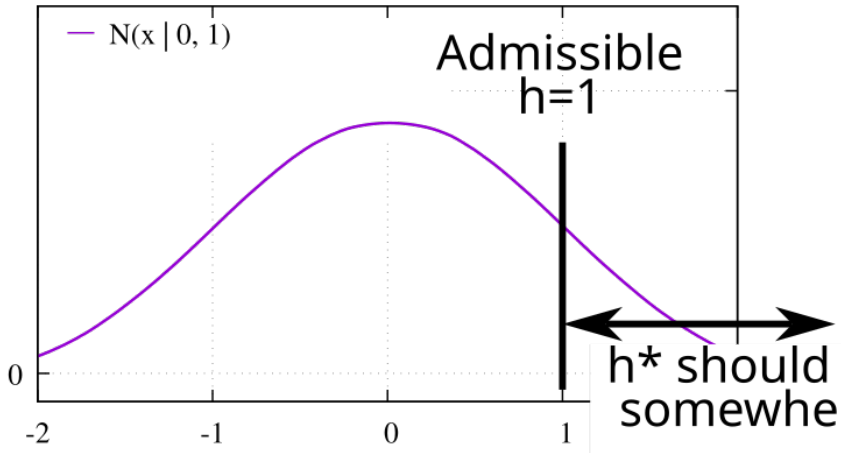


N(μ,σ) assigns
non-zero probability
to *all values* x ∈ R

- We know **h* ≧ 0**
- We know **h* ≧ h : admissible heuristics**

33.2. Thinking with Distributions

Is $\mathcal{N}(\mu, \sigma)$ the correct distribution for h^* ?

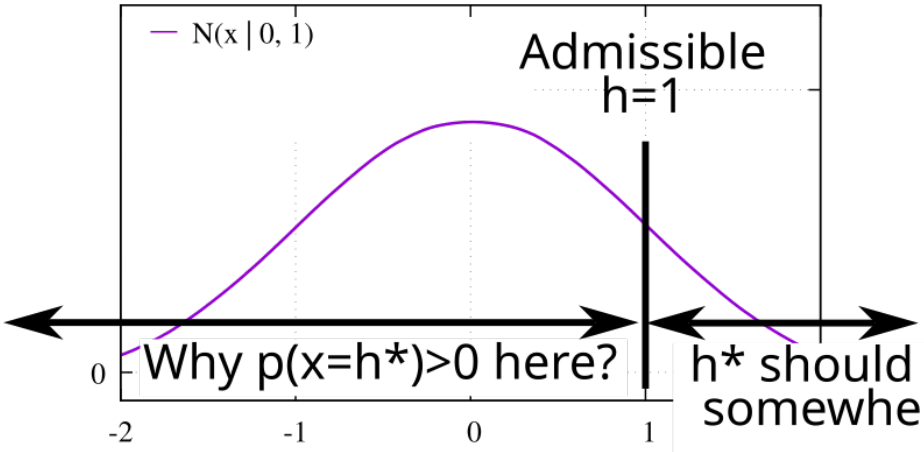


N(μ,σ) assigns
non-zero probability
to *all values* x ∈ R

- We know **h* ≧ 0**
- We know **h* ≧ h : admissible heuristics**

33.3. Thinking with Distributions

Is $\mathcal{N}(\mu, \sigma)$ the correct distribution for h^* ?

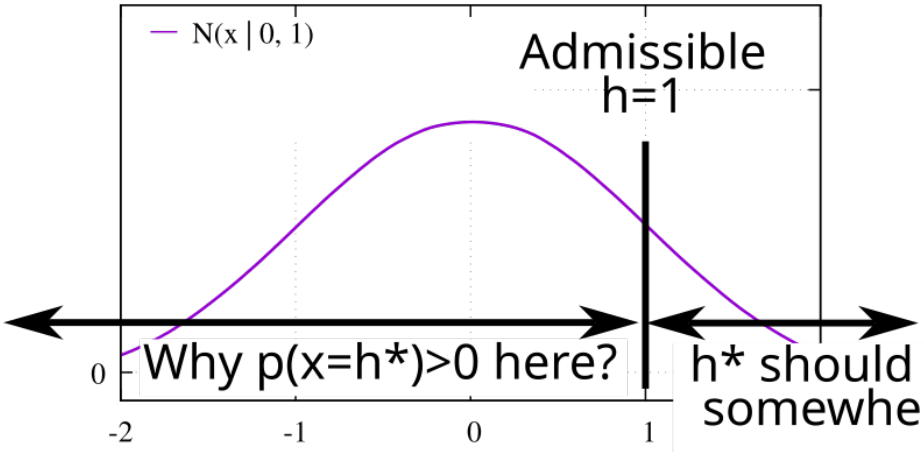


N(μ,σ) assigns
non-zero probability
to *all values* x ∈ R

- We know **h* ≧ 0**
- We know **h* ≧ h : admissible heuristics**
- **Why** do we assign **non-zero probability** to h*<h?

33.4. Thinking with Distributions

Is $\mathcal{N}(\mu, \sigma)$ the correct distribution for h^* ?

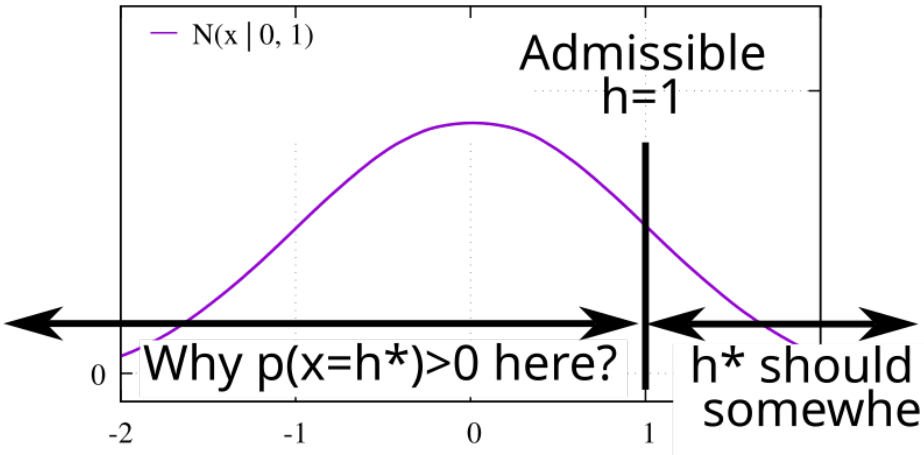


N(μ,σ) assigns
non-zero probability
to *all values* x ∈ R

- We know **h* ≧ 0**
- We know **h* ≧ h : admissible heuristics**
- **Why** do we assign **non-zero probability** to h*<h?
- i.e. N(μ,σ) **ignores** our **expert knowledge** on h*

33.5. Thinking with Distributions

Is $\mathcal{N}(\mu, \sigma)$ the correct distribution for h^* ?



N(μ,σ) assigns
non-zero probability
to *all values* x ∈ R

- We know **h* ≧ 0**
- We know **h* ≧ h : admissible heuristics**
- **Why** do we assign **non-zero probability** to h*<h?
- i.e. N(μ,σ) **ignores** our **expert knowledge** on h*
- It is **not** the **correct** distribution

34. What is the correct distribution?

Follow the ***Maximum Entropy Principle*** (Jaynes 1957):

*Choose the distribution with the **largest entropy***

*among those that satisfy the **expert knowledge / constraint**.*

↪ We know a lower bound $h \leq h^*$!

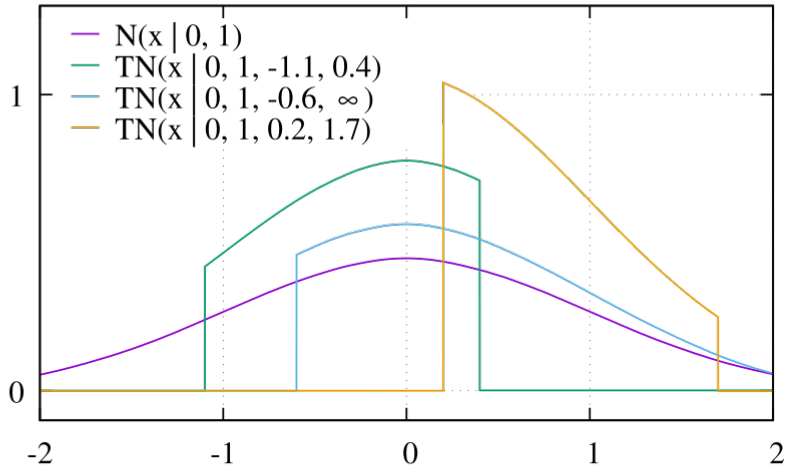
This is our expert knowledge!

35. What is the max-ent distribution for our assumption?

Truncated-Gaussian distribution TN(μ, σ, l, u): Dawson, Wragg 1973

Max-ent distribution under

- mean μ,
- variance σ^2,
- lower/upper bound l, u
  (u = ∞ is permissible)
  (l = LMcut heuristics)

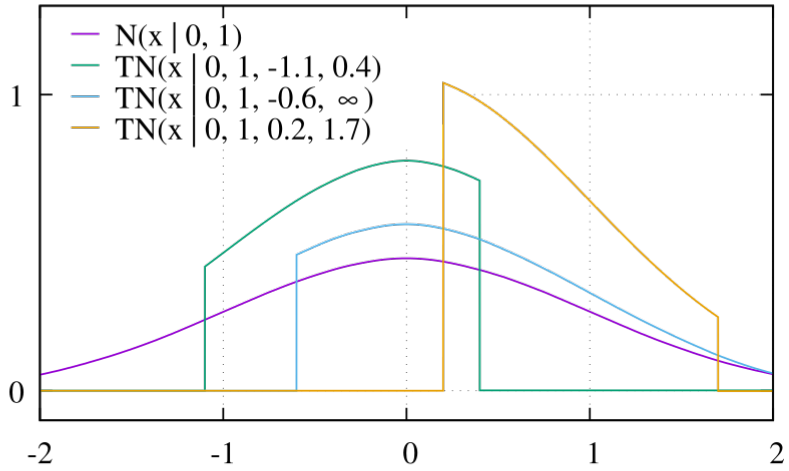


35.1. What is the max-ent distribution for our assumption?

Truncated-Gaussian distribution TN(μ, σ, l, u): Dawson, Wragg 1973

Max-ent distribution under

- mean μ,
- variance σ^2,
- lower/upper bound l, u
  (u = ∞ is permissible)
  (l = LMcut heuristics)



NLL Loss:

$$-\log \mathcal{TN}(x|\mu,\sigma,l,u) = \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2} + \log(\Phi(\tfrac{u-\mu}{\sigma}) - \Phi(\tfrac{l-\mu}{\sigma})).$$

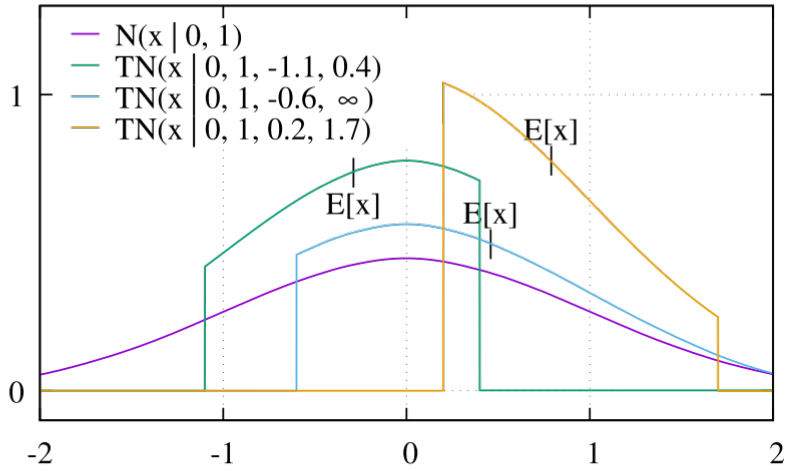
l=LMcut is different across the dataset

35.2. What is the max-ent distribution for our assumption?

Truncated-Gaussian distribution TN(μ, σ, l, u): Dawson, Wragg 1973

Max-ent distribution under

- mean μ,
- variance σ^2,
- lower/upper bound l, u
  (u = ∞ is permissible)
  (l = LMcut heuristics)



NLL Loss:

$$-\log \mathcal{TN}(x|\mu,\sigma,l,u) = \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2} + \log(\Phi(\tfrac{u-\mu}{\sigma}) - \Phi(\tfrac{l-\mu}{\sigma})).$$

l=LMcut is different across the dataset

We use E[x] as the heuristics for GBFS. l ≦ E[x] ≦ u

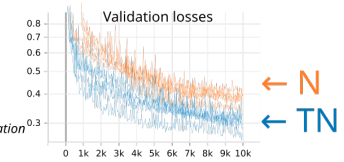
given by a different formula (omitted due to space)

35.3. Results

5. Results

Data generation details

- On 4 domains, we generated train, validation, test problems separately
- Train: 456-1536 instances, val/test: 132-384 instances
- *The test instances are larger than the training instances.* → *Evaluate problem-size generalization*
- h* : A*+LMcut on Fast Downward under 5min/8GB (train/val) or 30min/8GB (test)



TN training: faster & more accurate

h^{FF} vs. none: μ = NN output + h^{FF} vs. μ= NN output
learn vs. fixed: σ = NN output vs. σ= $\frac{1}{2}$
+clip: Clip μ with lmcut when computing the MSE.

Training details:

- 40k steps
- AdamW batch-size 256 decay 10^{-2}
- gradient clip 0.1, lr=10^{-2}
- 0.5-2 hours / training

Findings:

- **N vs TN: TN outperforms N**
  - N often completely fails esp. w/o h^{ff}
- **Residual learning (h^{FF})** is effective
- **Learning σ** is crucial for TN
- **N+clip vs TN: Clipping** is not enough

				learn/h^{FF}		learn/none		fixed/h^{FF}		fixed/none	
domain	metric	h^{FF}	h^{LMcut}	$\mathcal{N}$	$\mathcal{TN}$	$\mathcal{N}$	$\mathcal{TN}$	$\mathcal{N}$	$\mathcal{TN}$	$\mathcal{N}$	$\mathcal{TN}$
blocks	MSE	22.8	25.06	.76±.1	**.65±.1**	3.26±.6	**2.71±.4**	.83±.1	**.66±.1**	2.97±.9	**2.44±.3**
	+clip			.76±.2		2.91±.4		.83±.2		2.74±.6	
ferry	MSE	9.77	11.10	3.73±.7	**3.45±.8**	141.05±29.4	**8.63±2.7**	**2.98±1.4**	3.85±.9	118.59±10.4	**9.58±1.5**
	+clip			3.72±.6		10.44±28.4		**2.98±1.1**		10.50±9.6	
gripper	MSE	9.93	15.82	3.65±.9	3.70±.9	68.12±16.0	**5.65±1.3**	3.69±.9	3.72±.9	68.22±16.1	**11.97±2.2**
	+clip			3.65±.7		13.37±15.2		3.69±.8		13.38±14.5	
visitall	MSE	13.9	36.4	7.67±.4	**5.30±.6**	25.31±7.9	**9.70±1.6**	6.49±.6	6.62±.9	21.71±2.6	**14.11±1.0**
	+clip			7.60±.4		18.79±7.3		**6.35±.6**		16.38±2.3	

Test metrics for NLM (smaller the better). Each number represents the mean±std of 5 random seeds. For each configuration, we performed 10^4 training steps, saving the checkpoints with the best validation MSE metric.

TN searches faster

- Instances: 100 instances subsampled from test
  - ↪ **Larger than training**
- Pyerplan GBFS with early-stopping (added)
- Limited to 10k node evaluations
- Compared **success ratio** and **avg. evaluations**

Findings:

- **TN outperforms N, N+clip**
- **TN also outperforms h^{FF}**
- Better to **learn σ** and **learn the residual from h^{FF}**
  - ↪ TN's heuristics E[x] depends on σ

		learn/h^{FF} (proposed)			fixed/none (baseline)		
domain	h^{FF}	$\mathcal{N}$	$\mathcal{N}$+clip	$\mathcal{TN}$	$\mathcal{N}$	$\mathcal{N}$+clip	$\mathcal{TN}$
Ratio of solved instances under 10^4 evaluations (higher the better)							
blocks	.13	.84±.19	.85±.19	**.88±.14**	**.79±.29**	.50±.35	.55±.33
ferry	.82	.91±.19	.91±.19	**.98±.05**	.01±.01	.57±.10	**.58±.13**
gripper	.96	**1**	**1**	**1**	0	.92±.12	**1**
visitall	.86	.97±.07	**.98±.06**	**.98±.05**	.82±.33	**1**	**1**
Average node evaluations (smaller the better)							
blocks	9309	2690±2128	2681±2121	**2060±1607**	**4118±2663**	6268±2675	5903±2685
ferry	5152	3216±1964	3117±1967	**2477±1093**	9933±92	6675±582	**6475±725**
gripper	3918	1642±139	1643±141	**1637±492**	10000±0	2941±1513	**1709±658**
visitall	3321	2156±1451	2148±1511	**1683±1290**	3384±3448	**591±216**	612±363

- Model-agnostic: tested NLM, GNN, linear regression
- Comparable performance to rank-based approach

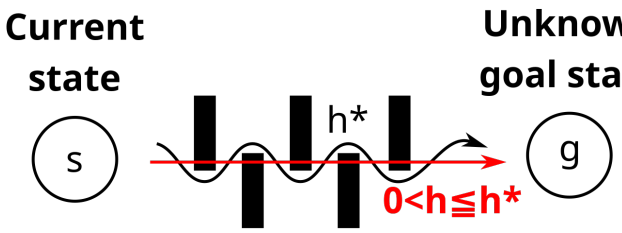
notes:

- Chrestein et. al. uses HGN → we compared our HGN models vs Chrestein et. al.
- Our HGN models is not our best model (it performs worse than NLM models.)

		learn/h^{FF}			(Chrestien *et al.*, 2023)
domain	h^{FF}	$\mathcal{N}$	$\mathcal{N}$+clip	$\mathcal{TN}$	L^{GBFS}
Ratio of solved instances under 10^4 evaluations (higher the better)					
blocks	.13	.70±.30	**.72±.29**	.48±.26	.24
ferry	**.82**	.01±.02	.01±.02	.02±.06	.61
gripper	**.96**	.36±.14	.37±.15	.39±.13	–
visitall	.86	**.99±.03**	.97±.04	.97±.04	0
Average node evaluations (smaller the better)					
blocks	9309	3984±2675	**3906±2649**	5844±2088	8282.6
ferry	5152	9916±132	9915±134	9834±424	**5103.5**
gripper	**3918**	7078±971	7008±1058	6949±1020	–
visitall	3321	1512±1192	1555±1149	**1472±1182**	10000

36. IJCAI 2024 : Conclusion

We learn h^*



We know a certain lower bound of h*:
Admissible heuristics h
obtained by solving a **relaxed problem**

Square error = Gaussian

Choosing a loss function ⇔ Choosing a distribution

Shoutout: don't "innovate" a new loss!
Innovate a **distribution** instead!

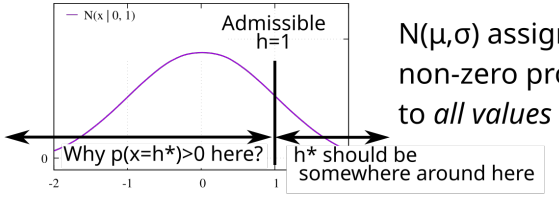
(Mean) Squared error ⇔ **Gaussian** NLL

$$-\log \mathcal{N}(\mu,\sigma) = \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2}$$

(Mean) Absolute error ⇔ **Laplacian** NLL

$$-\log \mathcal{L}(\mu,b) = \frac{|x-\mu|}{b} + \log 2b$$

Gaussian is a lie



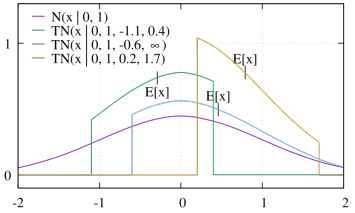
N(μ,σ) assigns non-zero probability to *all values* x ∈ R

- We know **h* ≧ 0**
- We know **h* ≧ h : admissible heuristics**
- **Why** do we assign **non-zero probability** to h*<h?
- i.e. N(μ,σ) **ignores** our **expert knowledge** on h*
- It is **not** the **correct** distribution

Max-ent: Truncated Gaussian

Truncated-Gaussian distribution TN(μ, σ, l, u): Dawson, Wragg 1973

Max-ent distribution under
- mean μ,
- variance σ^2,
- lower/upper bound l, u
  (u = ∞ is permissible)
  (l = LMcut heuristics)



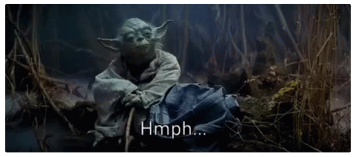
NLL Loss:
$$-\log \mathcal{TN}(x|\mu,\sigma,l,u) = \frac{(x-\mu)^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2} + \log(\Phi(\tfrac{u-\mu}{\sigma}) - \Phi(\tfrac{l-\mu}{\sigma})).$$

l=LMcut is different across the dataset

We use E[x] as the heuristics for GBFS. l ≦ E[x] ≦ u
given by a different formula (omitted due to space)

Take-home message: **Whenever you see a square error, doubt it!**

37. 理性的な実世界エージェントのアーキテクチャ



ノイズだらけの実世界観測データ

記号接地 ⇕ **(グラフィカルモデルによる 教師なし 離散表現 学習)**

記号的表現 & ソルバによる高速求解

```
above(A,B): on(A,C), above(C, B)
above(A,B): on(A,B)
move(A, B):
  preconditions: handempty, clear(A) ...
  effects:  ...
```

オフライン学習 (Reinforcement Learning ⇕ Un/Semi/Supervised Learning) ⇕ **オンライン学習/枝刈り (Multi-Armed Bandit)**

探索ヒューリスティクス
h(s), V(s), π(a), Q(s, a)

探索アルゴリズム
A*, GBFS, MCTS

38. 探索アルゴリズムの改善:

MCTS + Gaussian Bandit (ECAI 2024)

MCTS + Power Bandit (SoCS 2024 extended abstract)

39. ECAI 2024

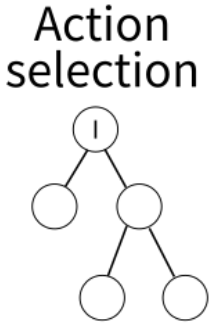
Scale-Adaptive Balancing of Exploration and Exploitation in Classical Planning

Stephen Wissow*, Masataro Asai*

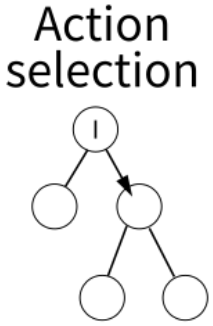
University of New Hampshire

MIT-IBM Watson AI Lab

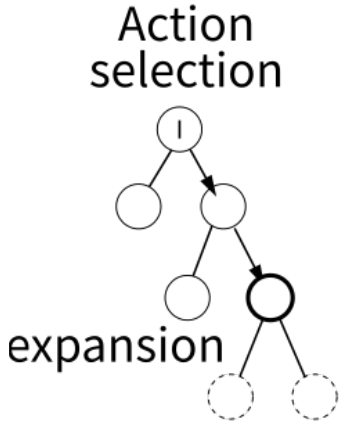
40. Monte Carlo Tree Search (General version)



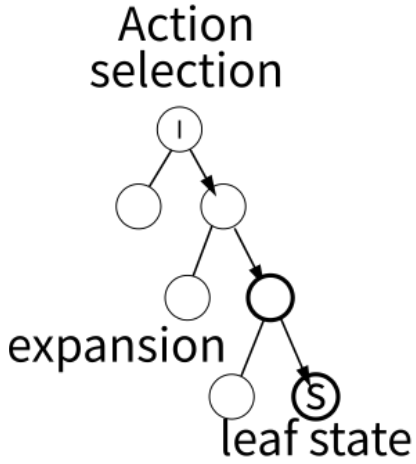
40.1. Monte Carlo Tree Search (General version)



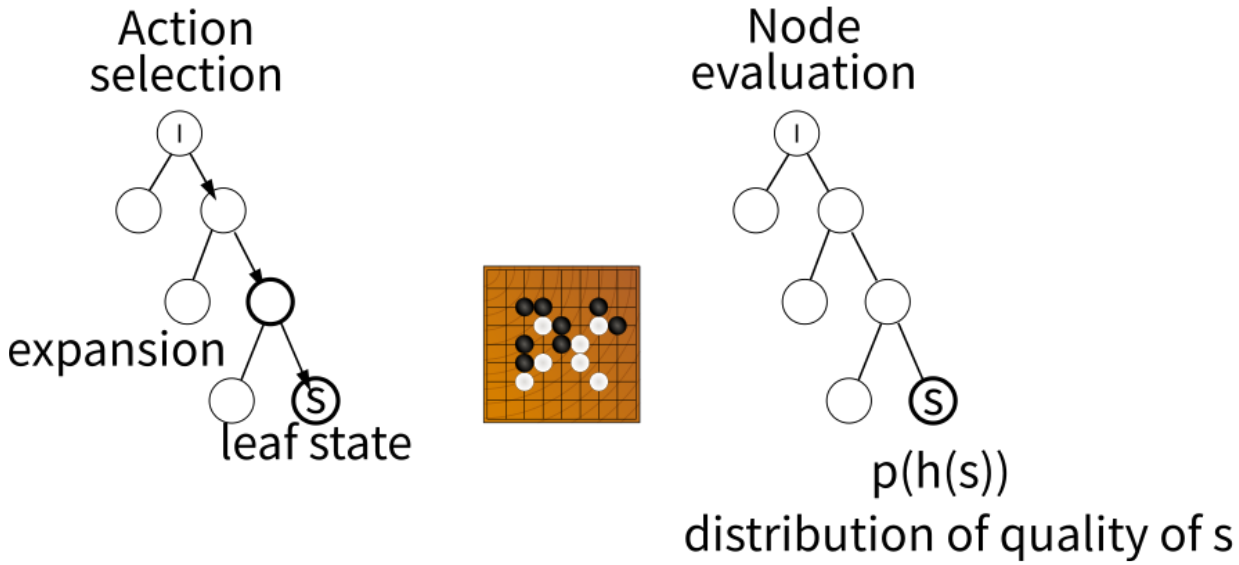
40.2. Monte Carlo Tree Search (General version)



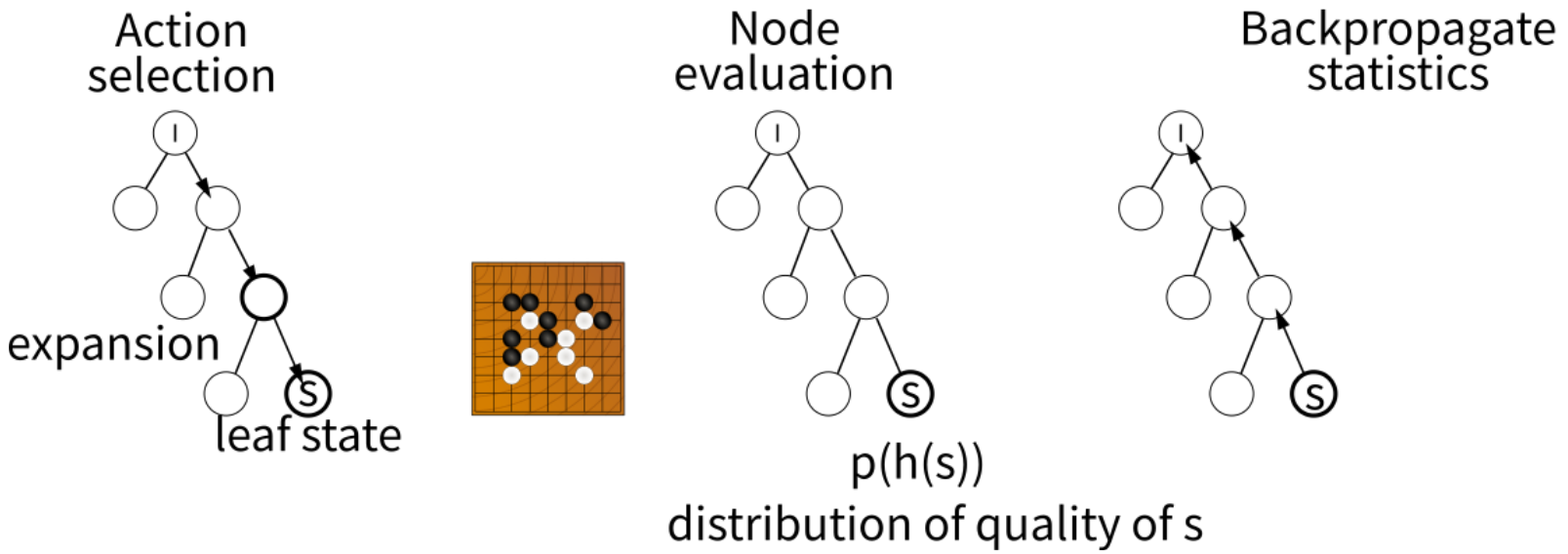
40.3. Monte Carlo Tree Search (General version)



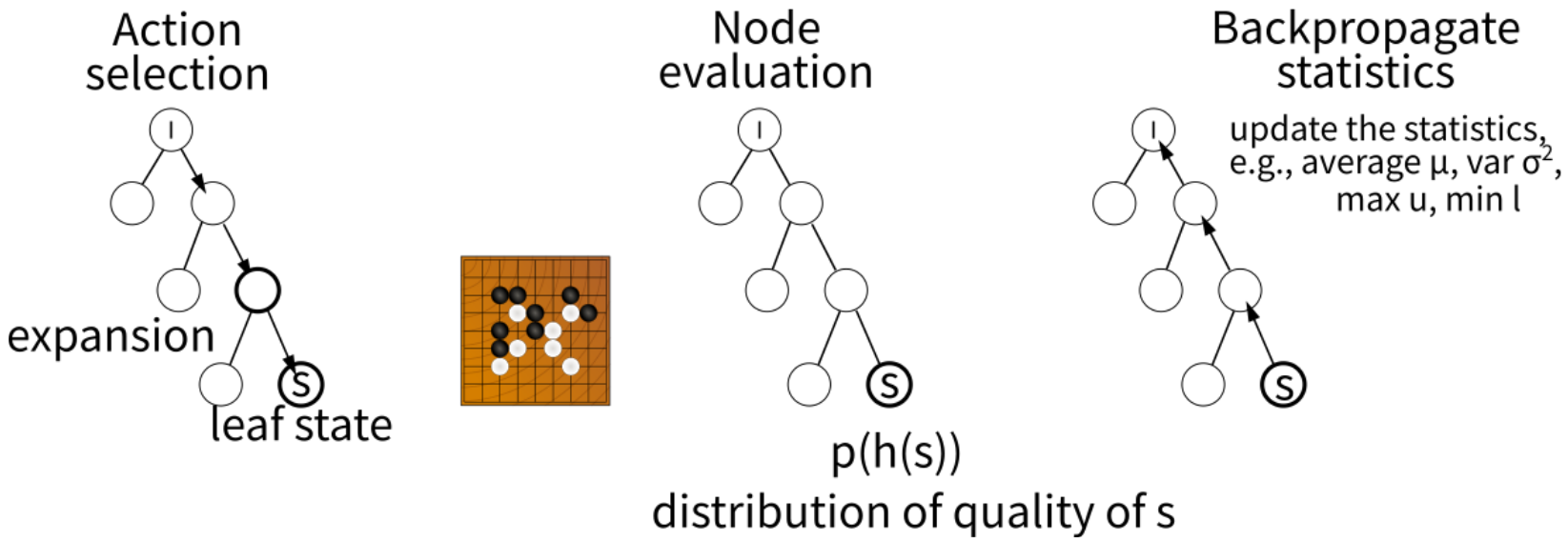
40.4. Monte Carlo Tree Search (General version)



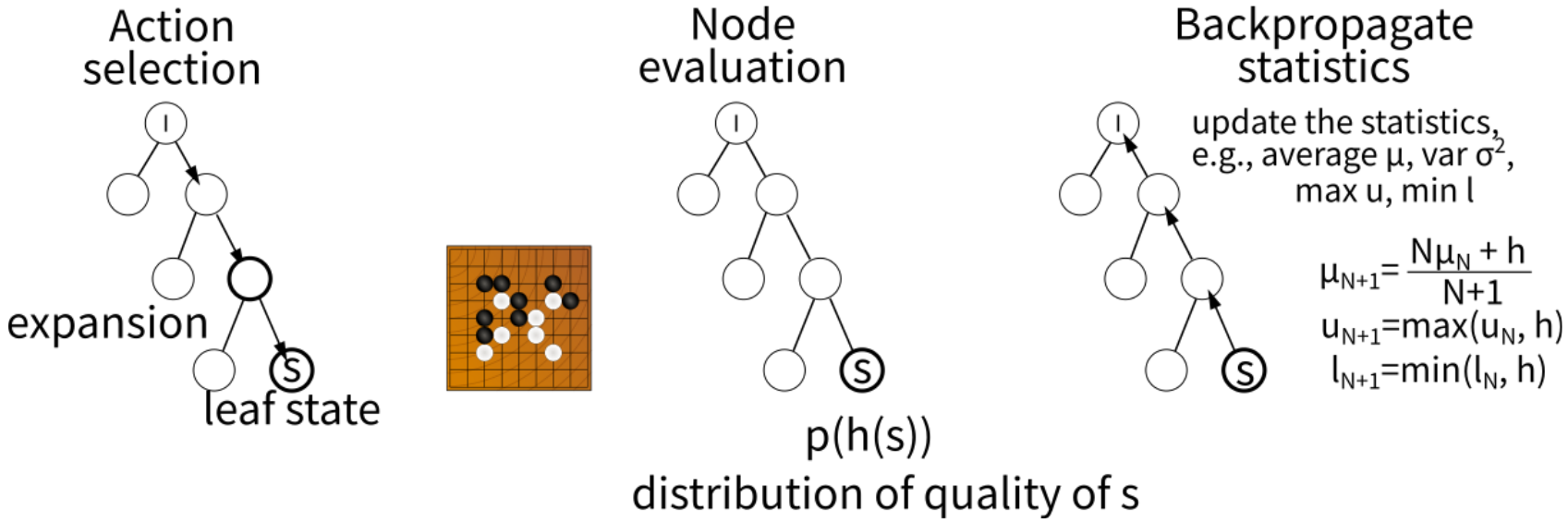
40.5. Monte Carlo Tree Search (General version)



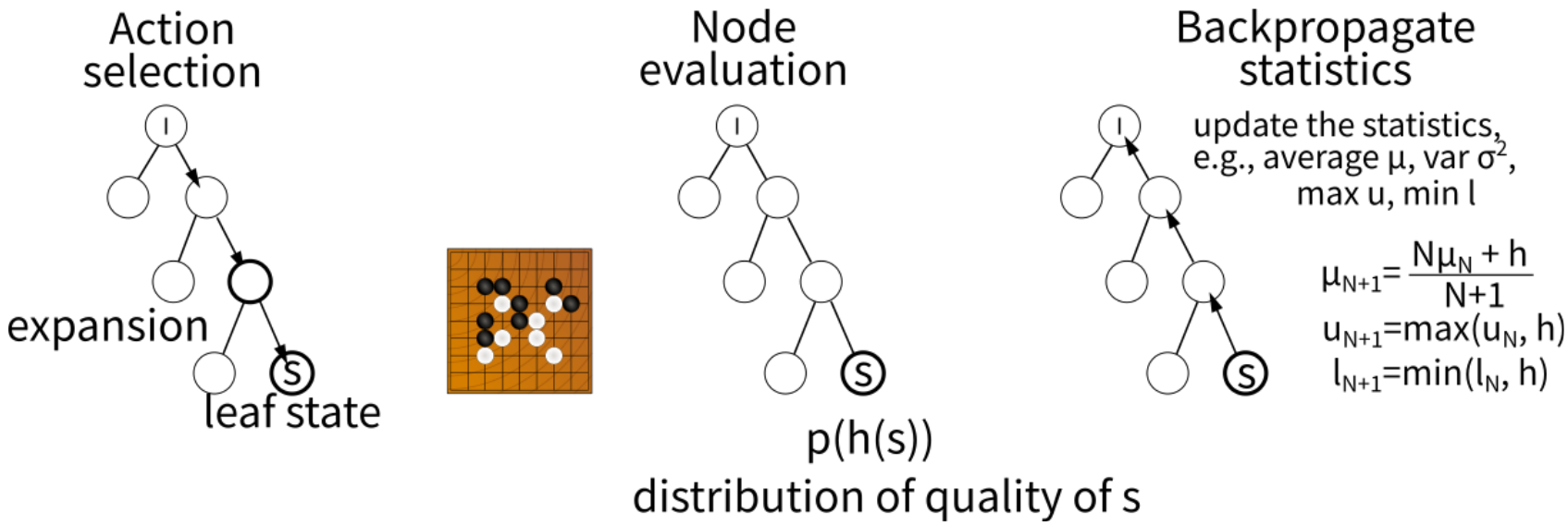
40.6. Monte Carlo Tree Search (General version)



40.7. Monte Carlo Tree Search (General version)

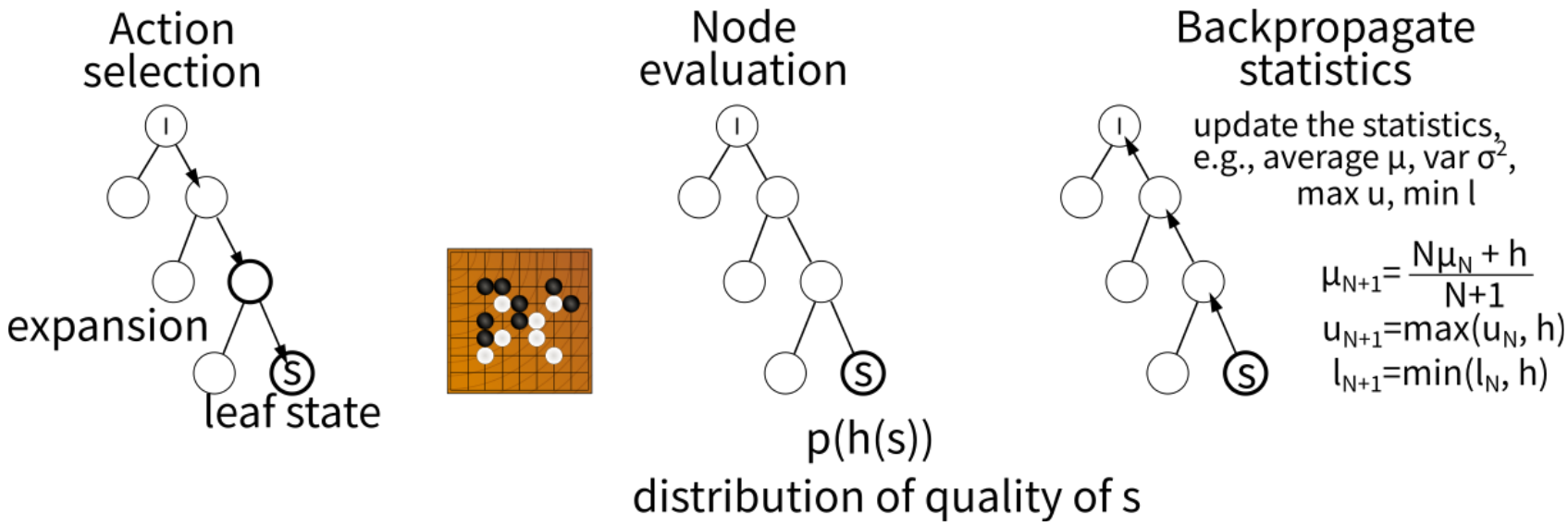


40.8. Monte Carlo Tree Search (General version)



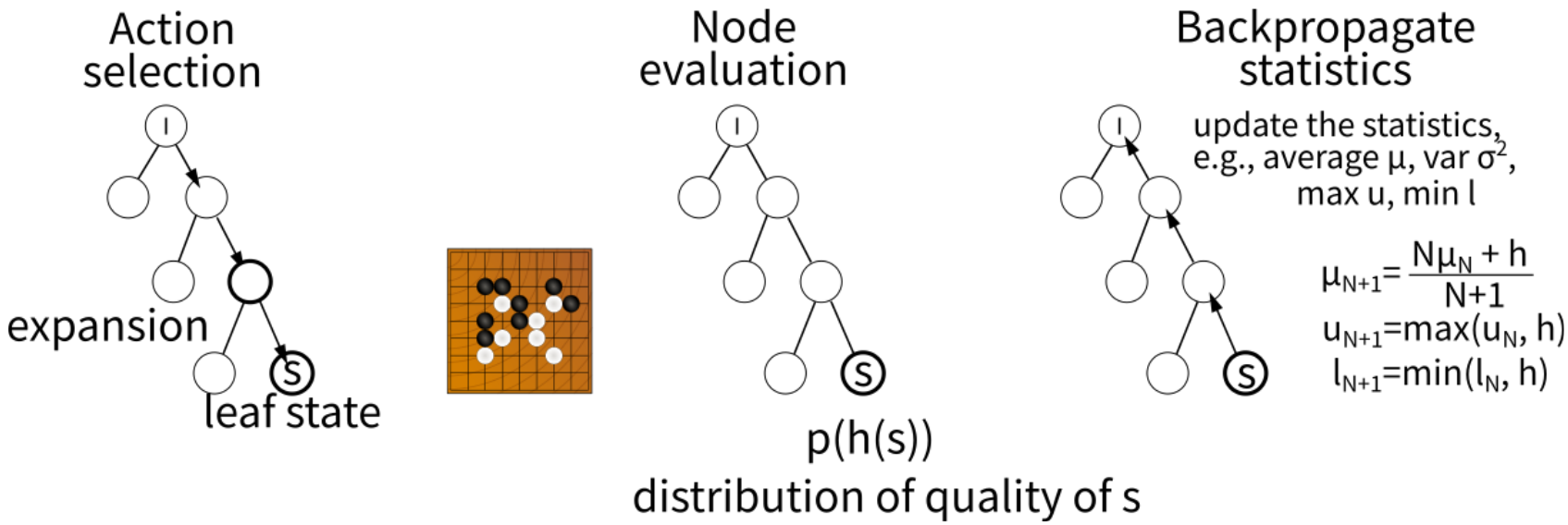
	Heuristic function
Game	**✗ Hand-coded**
(e.g., Go)	**✗ Not accurate**

40.9. Monte Carlo Tree Search (General version)



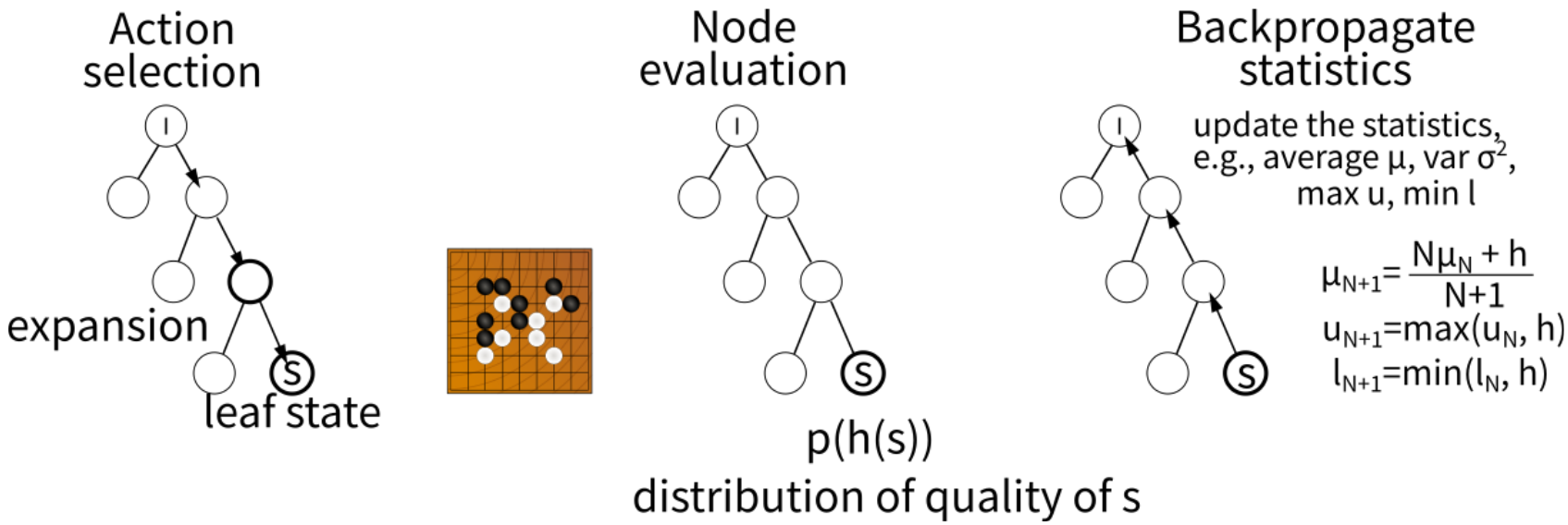
	Heuristic function	Simulation / Rollout
Game	✗ Hand-coded	✔ Finite horizon (9x9)
(e.g., Go)	✗ Not accurate	

40.10. Monte Carlo Tree Search (General version)



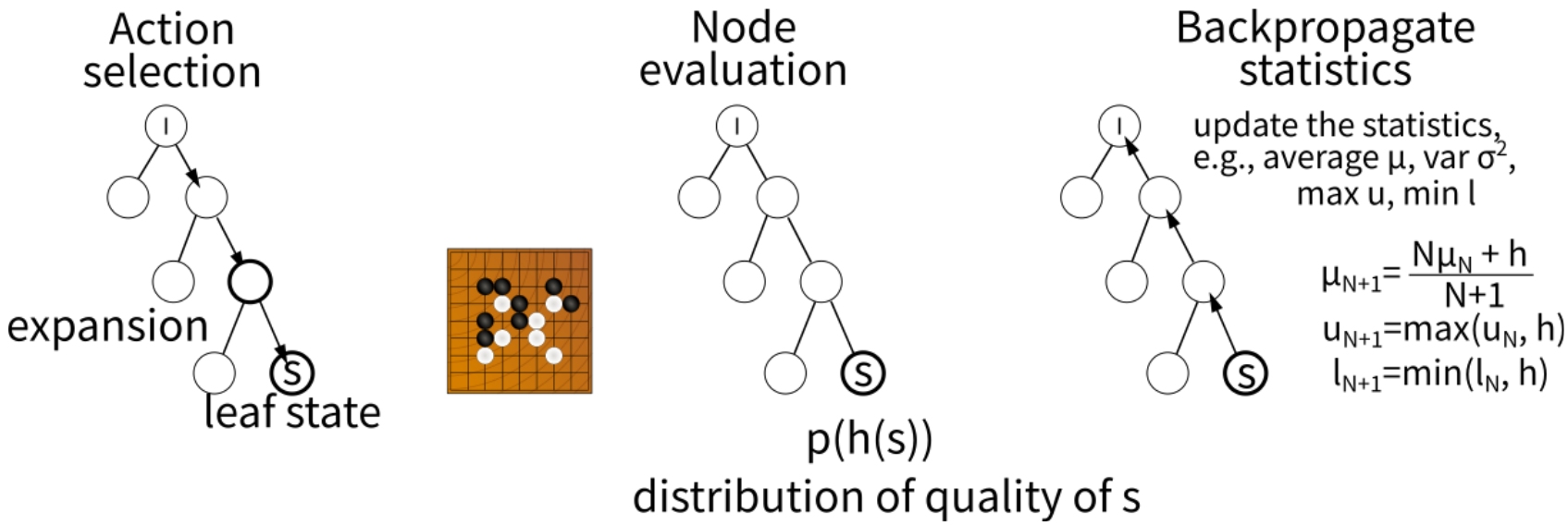
	Heuristic function	Simulation / Rollout
Game	✗ Hand-coded	✔ Finite horizon (9x9)
(e.g., Go)	✗ Not accurate	✔ Dense reward

40.11. Monte Carlo Tree Search (General version)



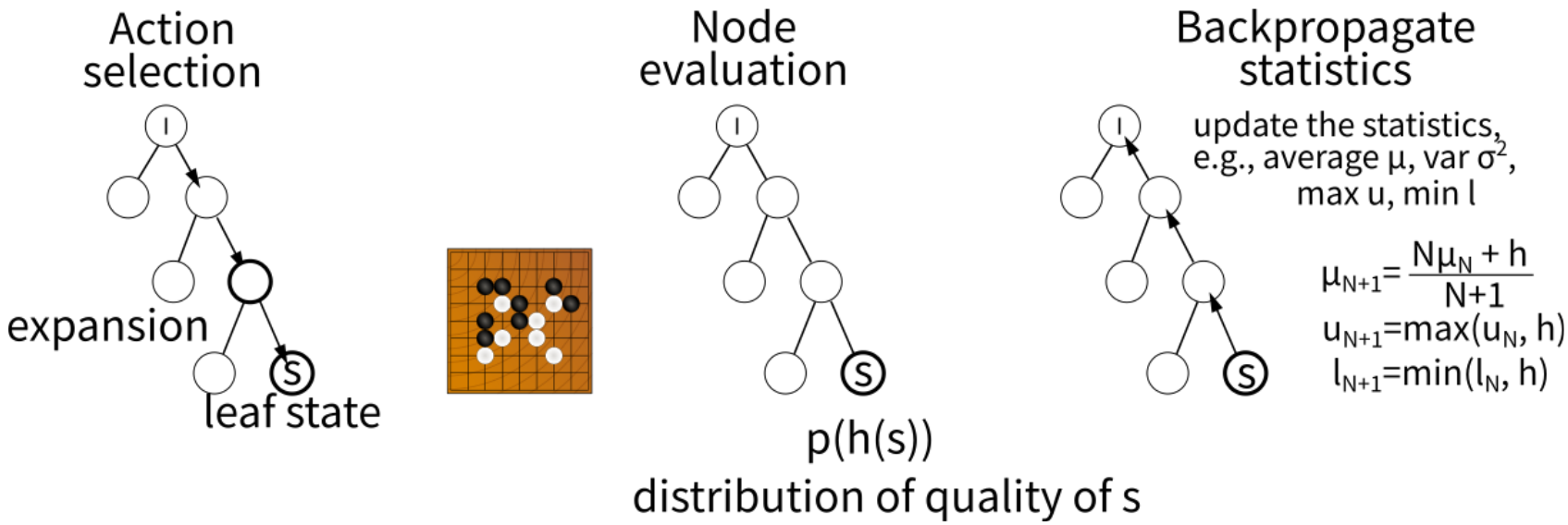
	Heuristic function	Simulation / Rollout
Game (e.g., Go)	✗ Hand-coded ✗ Not accurate	✔ Finite horizon (9x9) ✔ Dense reward
Classical Planning		

40.12. Monte Carlo Tree Search (General version)



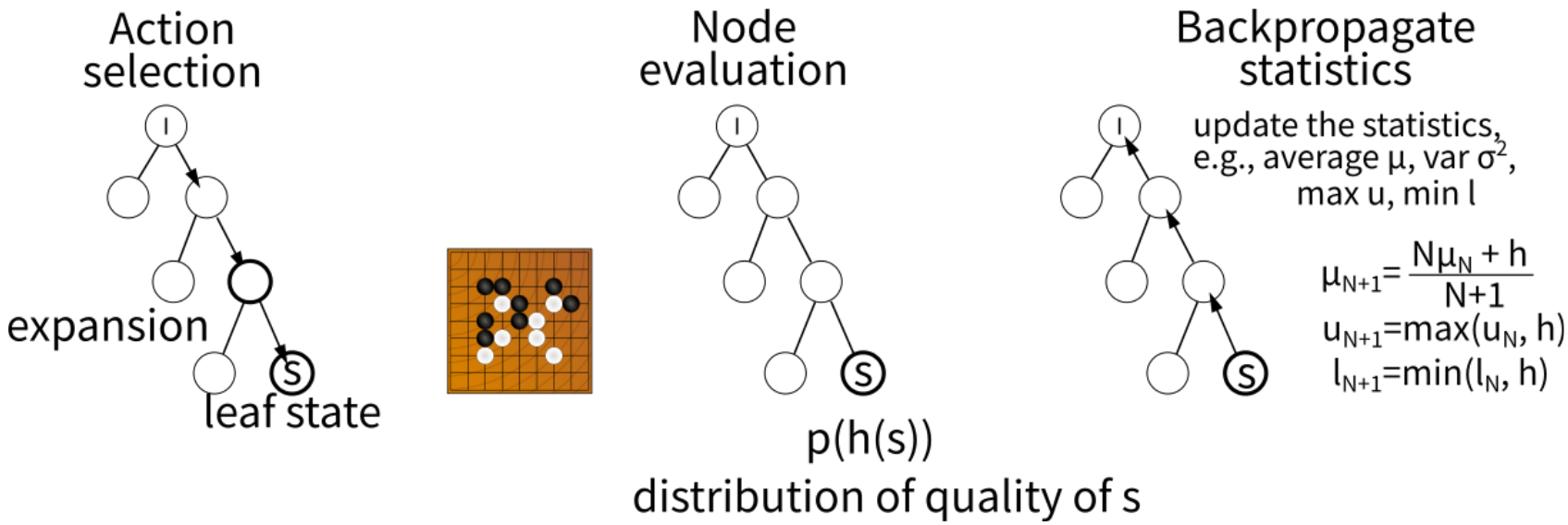
	Heuristic function	Simulation / Rollout
Game (e.g., Go)	✗ Hand-coded ✗ Not accurate	✔ Finite horizon (9x9) ✔ Dense reward
Classical Planning		✗ Infinite horizon

40.13. Monte Carlo Tree Search (General version)



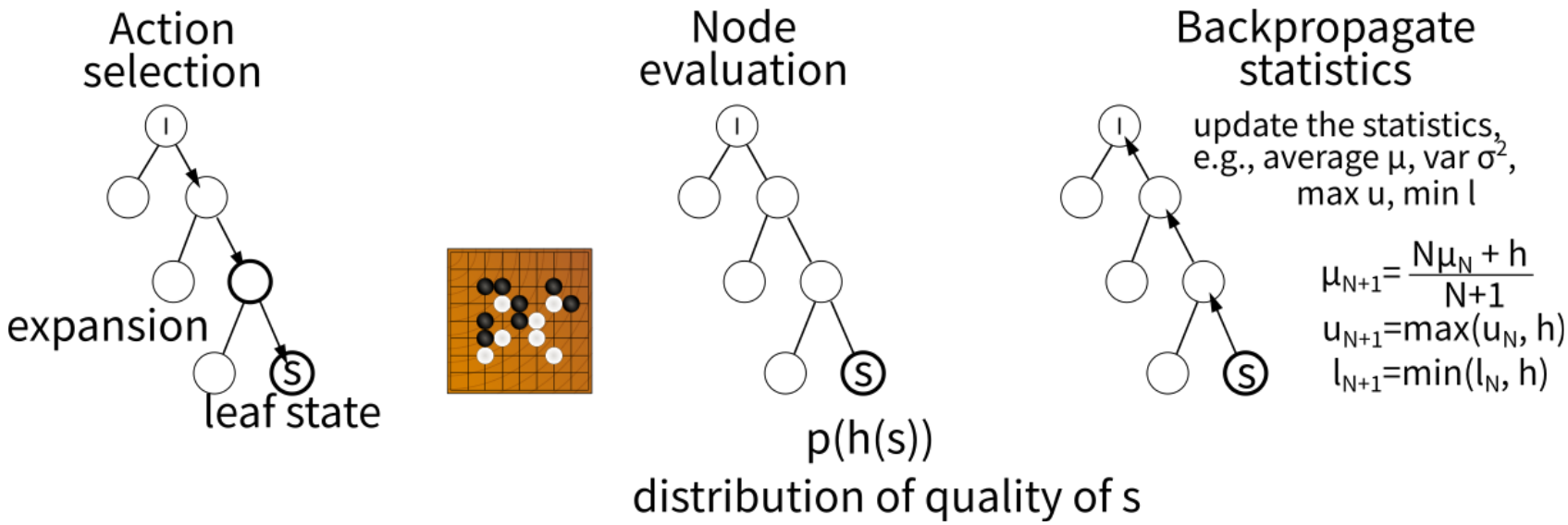
	Heuristic function	Simulation / Rollout
Game	✗ Hand-coded	✔ Finite horizon (9x9)
(e.g., Go)	✗ Not accurate	✔ Dense reward
Classical		✗ Infinite horizon
Planning		✗ Sparse reward

40.14. Monte Carlo Tree Search (General version)



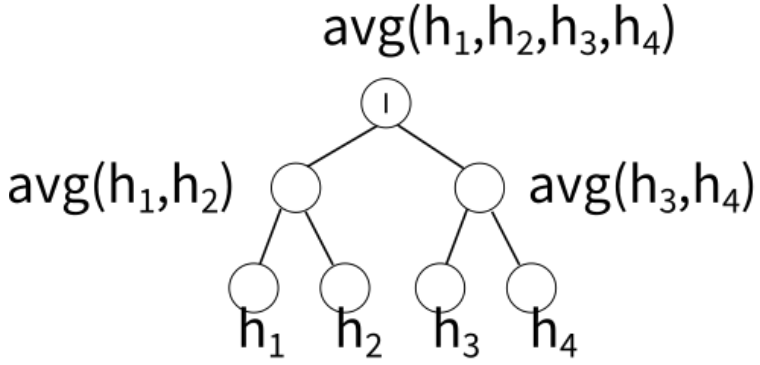
	Heuristic function	Simulation / Rollout
Game	✗ Hand-coded	✔ Finite horizon (9x9)
(e.g., Go)	✗ Not accurate	✔ Dense reward
Classical	✔ Automated	✗ Infinite horizon
Planning		✗ Sparse reward

40.15. Monte Carlo Tree Search (General version)

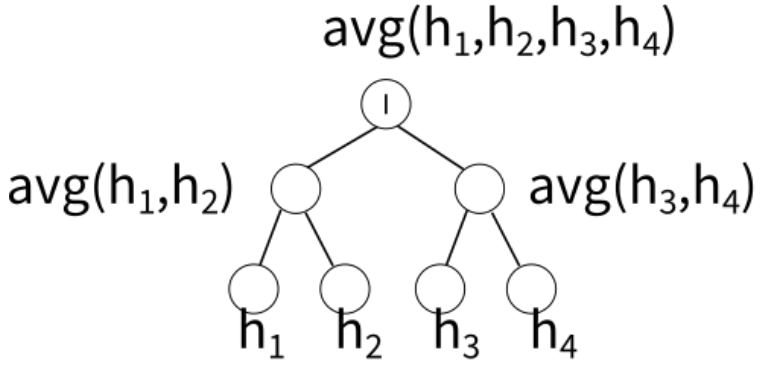


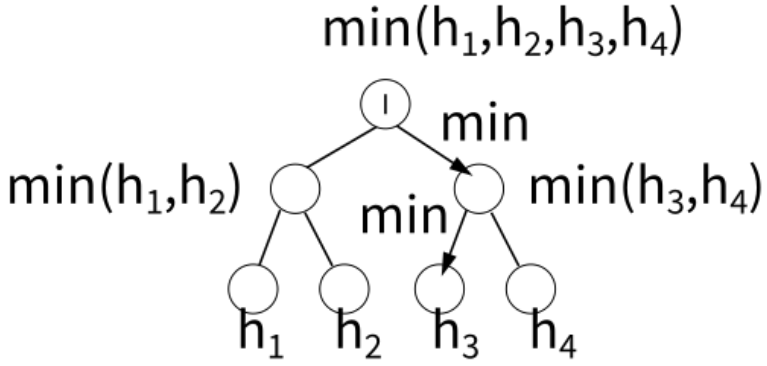
	Heuristic function	**Simulation / Rollout**
Game	✗ Hand-coded	✔ Finite horizon (9x9)
(e.g., Go)	✗ Not accurate	✔ Dense reward
Classical	✔ Automated	✗ Infinite horizon
Planning	✔ Domain-Independent	✗ Sparse reward

41. Backup in MCTS : Avg

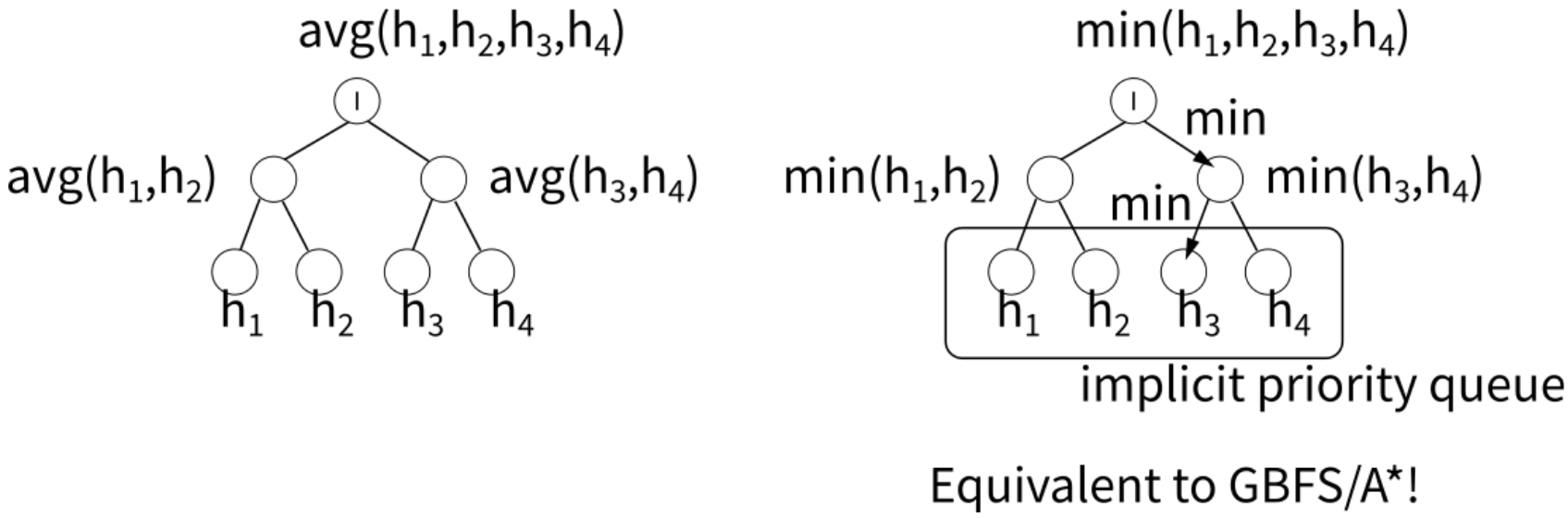


41.1. Backup in MCTS : Avg vs. Min





41.2. Backup in MCTS : Avg vs. Min = *GBFS, A**



GBFS: Traditional SotA

UCT has been ***bad***

(UCT = MCTS+UCB1)

Schulte and Keller. "Balancing exploration and exploitation in classical planning."
SOCS, 2014
Created: 2024-09-06 Fri 14:46

42. Why: *Poor theoretical understanding* of bandits.

$$\text{UCB1}_i = \mu_i - c\sqrt{\frac{\log N}{n_i}} \text{ , for } \mu_i \in [0, c]$$

N : parent visit, n_i : visit for child i

μ_i : mean over the subtree of i

Rewards must have a *known, fixed, shared reward range*.

But $h(s)$ **have *no known range!***

Requirments unsatisfied. All theoretical guarantees violated.

Our solution: Gaussian Bandit defined on **Entire** $\mathbb{R} = [-\infty, \infty]$

$$\text{UCB1-Normal2}_i = \mu_i - \sigma_i\sqrt{\log N} \text{ , for } \mu_i \in \mathbb{R}$$

μ_i, σ_i : mean / stdev over the subtree of i

43. Results (ECAI 2024)

$h =$	h^{FF}		h^{add}		$h^{\max}$		h^{GC}	
$c =$	0.5	1	0.5	1	0.5	1	0.5	1
GUCT	442.8	412.0	435.8	397.8	237.0	228.4	306.6	285.2
*	542.0	458.6	529.2	480.8	248.4	242.2	317.8	310.4
-01	399.8	368.0	375.4	328.8	256.8	237.4	318.4	302.4
*-01	425.6	388.0	404.8	364.4	246.8	233.4	318.0	297.6
-V	361.2	317.4	354.0	310.6	226.2	208.6	278.4	255.4
-Normal	-	283.4	-	265.0	-	212.0	-	233.4
*-Normal	-	318.8	-	300.0	-	215.2	-	246.2
-Normal2	-	**581.8**	-	535.8	-	**316.6**	-	**379.0**
*-Normal2	-	567.2	-	533.8	-	263.0	-	341.0
TTTS-Normal	-	181.0	-	180.0	-	171.4	-	170.8
TTTS-Normal*	-	189.4	-	186.4	-	177.4	-	174.4
GBFS(Pyperplan/FastDownward)	538/539	-	518/517	-	224/226	-	354/349	
WA* $(w = 5)$ (FastDownward)	528	-	522	-	211	-	319	
Softmin-Type(h) (FastDownward)	576.0	-	**542.6**	-	297.2	-	357.6	

44. New Algorithm : Model the minimum *correctly*

Fit the data to…

UCB1	a ***known*** finite support distribution	$[0,1]$, $[3,5]$
UCB1-Normal	a Gaussian distribution	$[-\infty,\infty]$

Do heuristics have *known* bounds e.g. $[3,5]$? *No!*

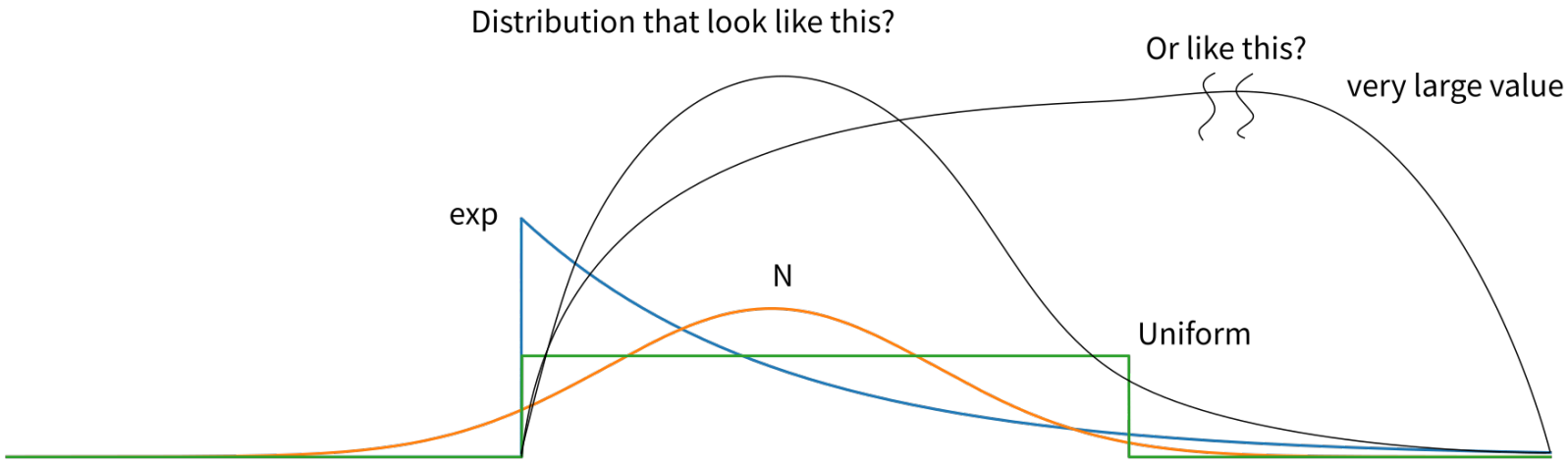
Are heuristic samples $\in [-\infty,\infty]$? *No! It is non-negative!*

$0 \geq h(s)$

Gaussian assumption is *STILL WRONG*

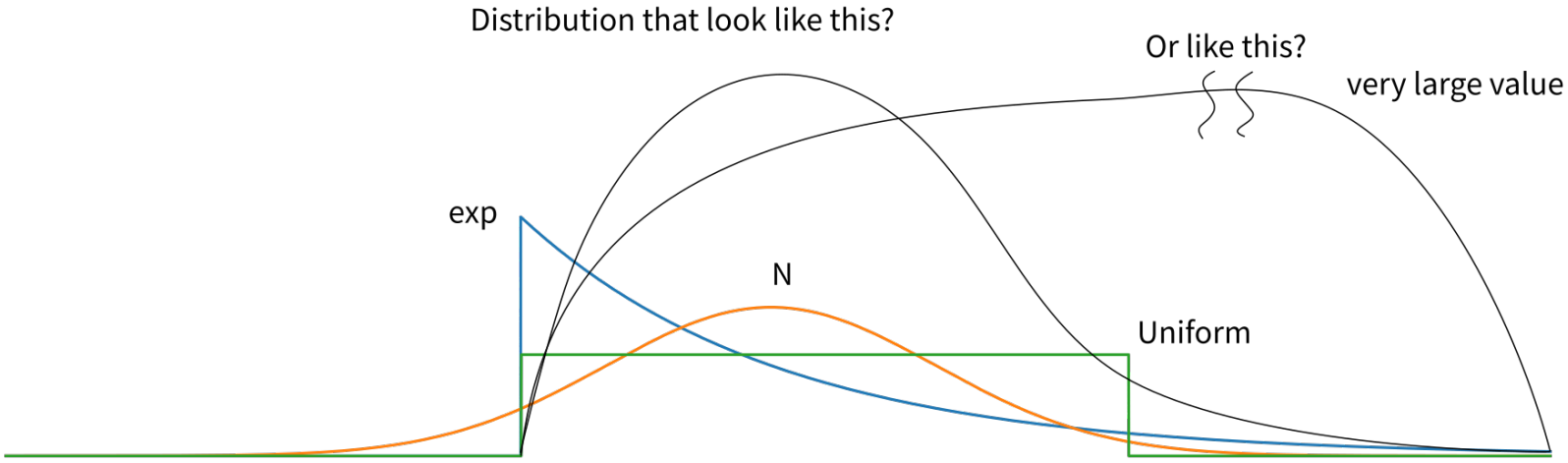
45. What distribution is the right answer?

	Range	How it is distributed?
Gaussian	***Unbounded*** ✗	
Exponential	***Half-Bounded***	
Uniform	***Bounded***	



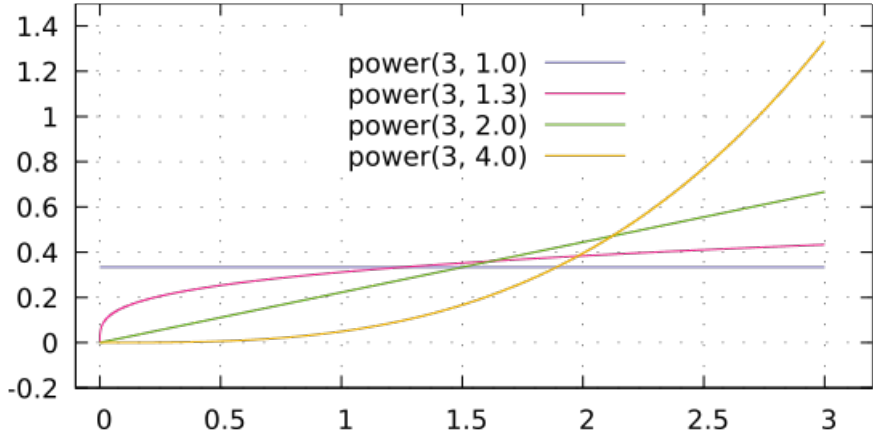
45.1. What distribution is the right answer?

	Range	How it is distributed?
Gaussian	***Unbounded*** ✗	
Exponential	***Half-Bounded***	***Smallest value is most common*** ✗
Uniform	***Bounded***	Equally likely around min and max



45.2. What distribution is the right answer?

	Range	How it is distributed?
Gaussian	***Unbounded*** ✗	
Exponential	***Half-Bounded***	***Smallest value is most common*** ✗
Uniform	***Bounded***	Equally likely around min and max
Power	***Bounded***	***Smallest value is rare***



46. But Why? *What theory* justifies Power?

Central Limit Theorem: Given i.i.d. RVs $x_0, \ldots x_n$, there is a sequence $a_n > 0, b_n$ s.t. $\exists \mu, \sigma$

$$S_n = \sum_{i=0}^{n} \frac{x_i - b_n}{a_n} \longrightarrow \mathcal{N}(\mu, \sigma)$$

46.1. But Why? *What theory* justifies Power?

Central Limit Theorem: Given i.i.d. RVs $x_0, \dots x_n$, there is a sequence $a_n > 0, b_n$ s.t. $\exists \mu, \sigma$

$$S_n = \sum_{i=0}^{n} \frac{x_i - b_n}{a_n} \longrightarrow \mathcal{N}(\mu, \sigma)$$

Extreme Value Theorem type 1: Given i.i.d. RVs $x_0, \dots x_n$, there is a sequence $a_n > 0, b_n$ s.t. $\exists \mu, \sigma, \xi$

$$M_n = \max \frac{x_i - b_n}{a_n} \longrightarrow \mathrm{EVD}(\mu, \sigma, \xi)$$

46.2. But Why? *What theory* justifies Power?

Central Limit Theorem: Given i.i.d. RVs $x_0, \ldots x_n$, there is a sequence $a_n > 0, b_n$ s.t. $\exists \mu, \sigma$

$$S_n = \sum_{i=0}^{n} \frac{x_i - b_n}{a_n} \longrightarrow \mathcal{N}(\mu, \sigma)$$

Extreme Value Theorem type 1: Given i.i.d. RVs $x_0, \ldots x_n$, there is a sequence $a_n > 0, b_n$ s.t. $\exists \mu, \sigma, \xi$

$$M_n = \max \frac{x_i - b_n}{a_n} \longrightarrow \mathrm{EVD}(\mu, \sigma, \xi)$$

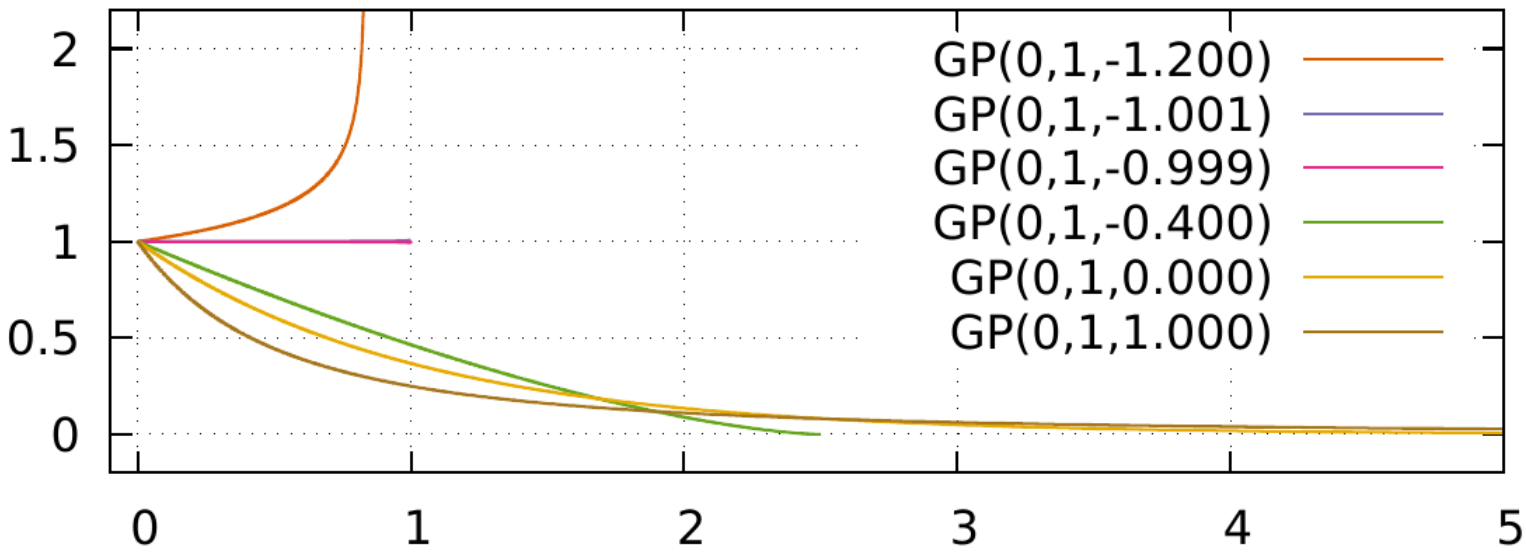
Extreme Value Theorem type 2: Given i.i.d. RVs $x_0 < \ldots < x_k < \ldots < x_n$, there is a sequence $a_n > 0, b_n$ s.t. $\exists \mu, \sigma, \xi$

$$p(x|x > x_k) \longrightarrow \mathrm{GPD}(\mu, \sigma, \xi)$$

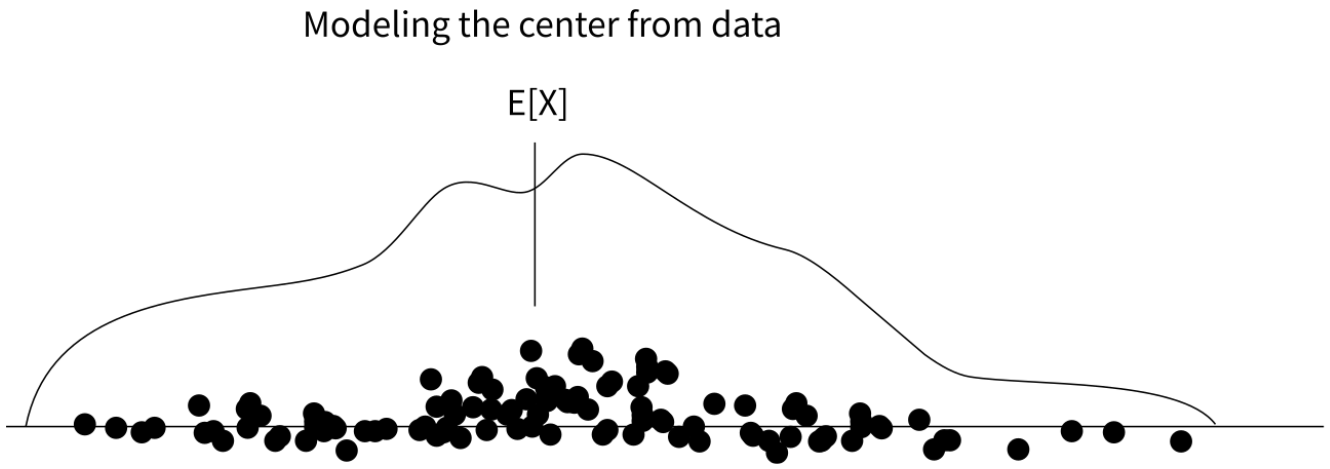
GPD : ***Generalized Pareto Distribution*** is the way to go. **Why?**

46.3. Power, Uniform, Exponential, Pareto are special cases of Generalized Pareto $\mathrm{GPD}(\theta, \sigma, \xi)$.

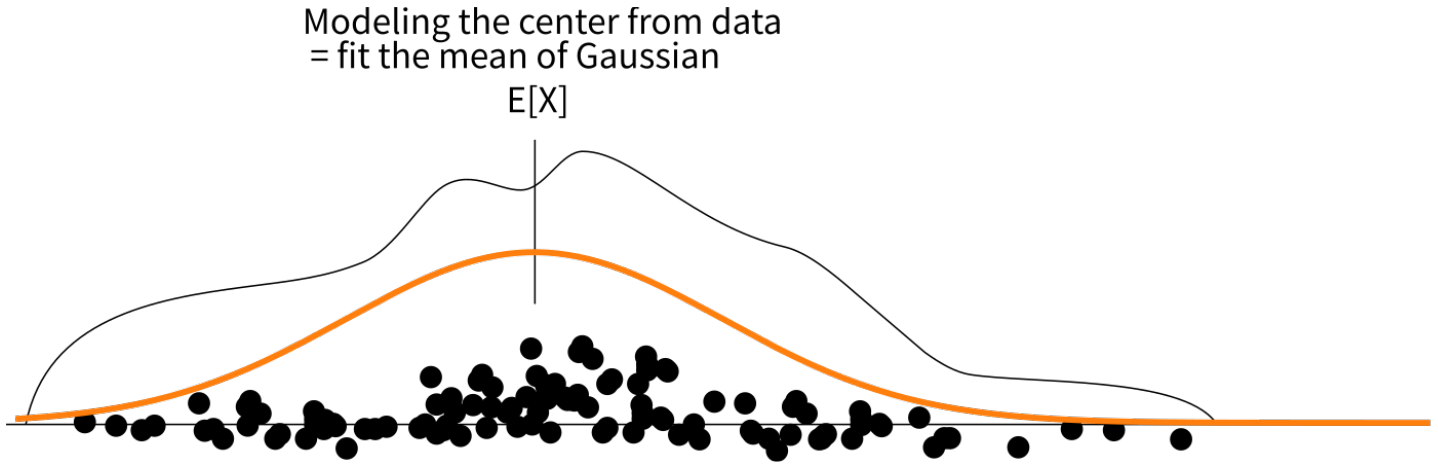
	min	max
$\xi > 0$ = Pareto	θ	∞
$\xi = 0$ = Exponential	θ	∞
$\xi < 0$ = (Flipped) Power	θ	$\theta - \frac{\sigma}{\xi}$
$\xi = -1$ = Uniform	θ	$\theta - \frac{\sigma}{\xi}$
$\xi < -1$	θ	$\theta - \frac{\sigma}{\xi}$


2
1.5
1
0.5
0
0
1
2
3
4
5
GP(0,1,-1.200)
GP(0,1,-1.001)
GP(0,1,-0.999)
GP(0,1,-0.400)
GP(0,1,0.000)
GP(0,1,1.000)

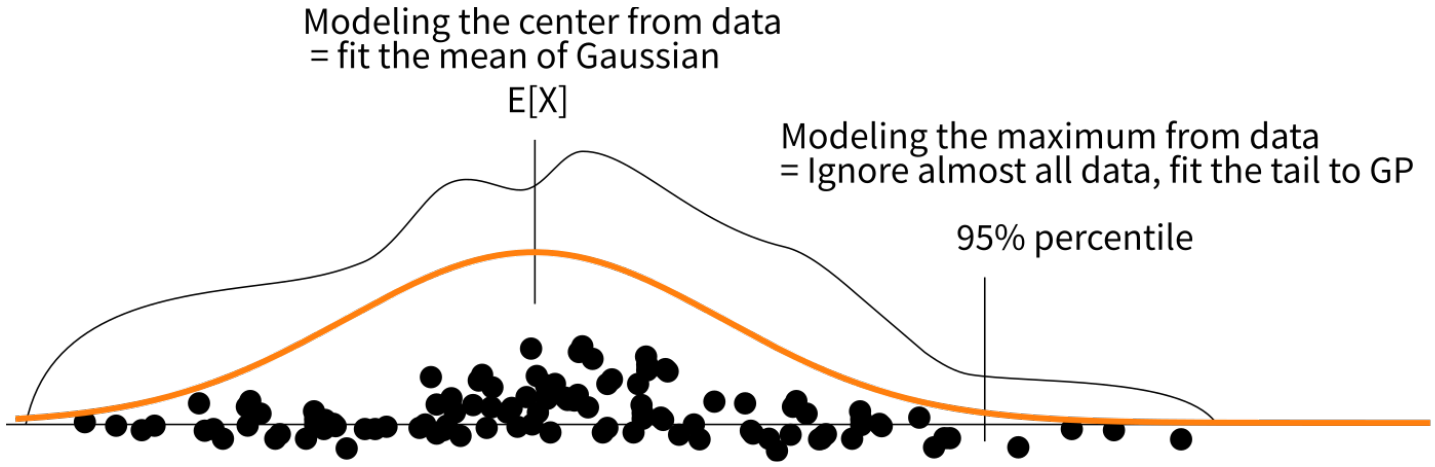
47. Modeling the average



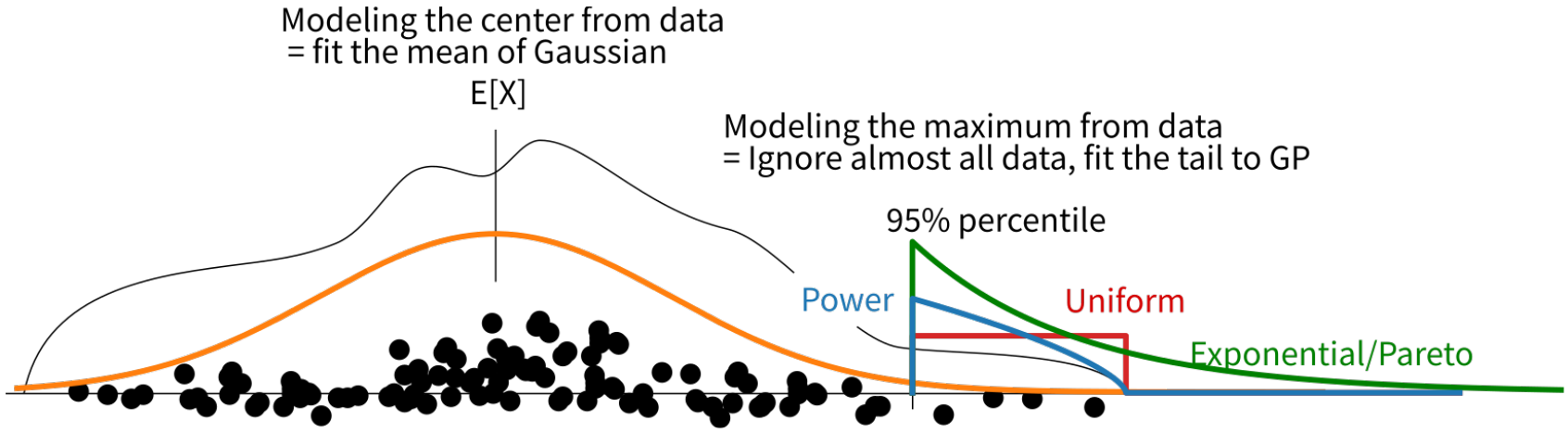
47.1. Modeling the average = Fitting the dist. with Gaussian



47.2. Modeling the maximum



47.3. Modeling the maximum = Fitting the tail dist. with GP



48. New Algorithm : Model the minimum *correctly*

Fitting GPD itself is **difficult** (computational issue)

→ fit subclasses: **Uniform** and **Power**

	Lower bound	Upper bound	Shape
UCB1	***Known*** 0	***Known*** 1	
UCB1-Normal	$-\infty$	∞	
UCB1-Uniform	***Unknown*** ℓ	***Unknown*** u	***Fixed, Flat***
UCB1-Power	***Known*** 0	***Unknown*** u	***Rare near goals***

$$\text{UCB1-Uniform}_i = \frac{u_i+l_i}{2} - (u_i - l_i)\sqrt{6n_i \log N}$$

$$\text{UCB1-Power}_i = \frac{u_i a_i}{a_i+1} - u_i\sqrt{6n_i \log N}$$

49. Results (SoCS 2024 Extended Abstract)

$h =$	h^{FF}	h^{add}	$h^{\max}$	h^{GC}
GBFS(Pyperplan/FD)	538/539	518/517	224/226	354/349
Softmin-Type(h)	576	542.6	297.2	357.6
GUCT	412	397.8	228.4	285.2
-Normal2	581.8	535.8	316.6	379
*-Normal2	567.2	533.8	263	341
+-Normal2	582.8	**549.2**	433	**474.6**
-Power	**596**	541.8	**450.6**	463.2
-Uniform	**594.8**	**543.8**	**450.6**	**463.8**
CHK-Normal	352.8	332.4	220	274.4
CHK-Uniform	359.2	343.4	214.7	284.6
MaxSearch	253.75	243.4	259.8	255.2
RobustUCT	265.6	262.4	232.6	229.8
ThresholdAscent	162.2	163.8	170.4	164.4

50. ECAI 2024 + SoCS 2024: Conclusion

UCT is slow in
Classical Planning

$$\mu - c\sqrt{\frac{\log N}{n_i}}$$

Why?

$h(s) \notin [0, c]$
(fixed range)

(ECAI 2024)
Use Gaussian Bandit!

$$\mu - \sigma\sqrt{\log N}$$

$h(s) \in [-\infty, \infty]$

STILL WRONG!

$0 \leq h(s)$

(SoCS 2024)
Use Uniform / Power!

$$\frac{u_i + l_i}{2} - (u_i - l_i)\sqrt{6n_i \log N}$$

$$\frac{u_i a_i}{a_i + 1} - u_i\sqrt{6n_i \log N}$$

Use bandits correctly!

51. Conclusion & Take-home message

記号接地:

PDDLを **正しく** モデル化しよう!

→グラフィカルモデル

ヒューリスティクス学習:

RL : Reward Shaping を **正しく** やろう!

Discouning , ***Undiscounting***

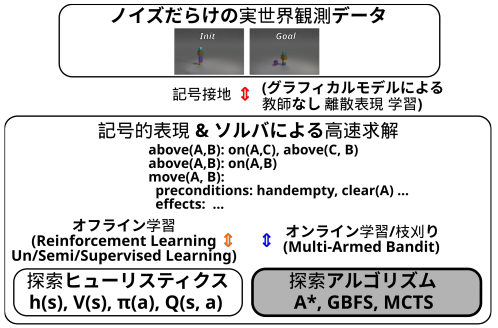
SL : $h^*(s)$ を **正しく** モデル化しよう!

Truncated Gaussian

探索アルゴリズム改善 / Bandit:

$h(s)$ を **正しく** モデル化しよう!

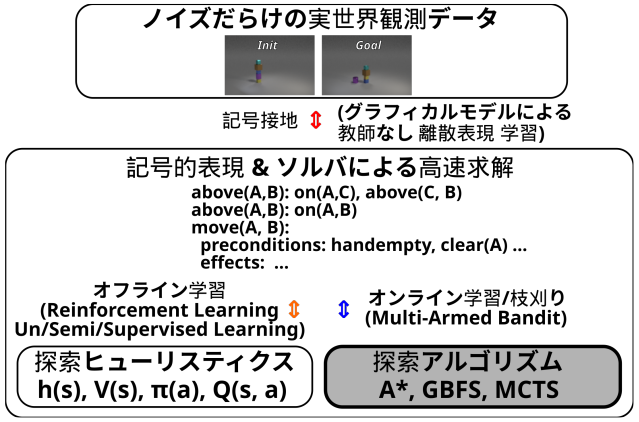
→ Gaussian, Power



正しさへの
こだわりは
いつか
実を結ぶはず
(大変だけど)

52. Deep RL は 正しい か?

研究のやり方に問題があるように思われる



Deep RL は **全部ごった煮**

- 表現学習
- ヒューリスティック
- 探索アルゴリズム
- Active Learning
- Contiual Learning

1つの変更で全部が影響受ける

なぜ性能良くなったのか、わからなくない?

再現性の問題も、ある...

Leslie Kaelbring: RLの評価は最低20シードないと認めない！

The science as a **sane** researcher:

- ***Split*** the architecture into parts
- ***Study each part individually***
- ***Correct mathematical modeling***

こういうアプローチに立ち返るのがいいと思います...

53. LLM + Planning (, Reasoning) が成功するには?

単体ではおそらくプランニング不可 by Rao (Subbarao Kambhampati)

おおよそRLと同じような問題を抱えている (ごった煮)

全部一度にこなそうとしている

表現学習：ここだけはとても汎化している

探索：全くしていない(Reflex Agent) か 非常に原始的 (CoT)

ヒューリスティクス：「常識」の丸暗記

プランニングに対する **解像度** が低い

Gundawar, Atharva, et al. "Robust Planning with LLM-Modulo Framework: Case Study in Travel Planning." arXiv
Kambhampati, Subbarao, et al. "LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks." arXiv
Valmeekam, Karthik, et al. "On the planning abilities of large language models-a critical investigation." Neurips 2023
Created: 2024-09-06 Fri 14:46

54. 自分が将来的にやりそうなネタ

表現学習 : LLM を使ってPDDL生成 (Latplanと同じ)

ヒューリスティクス : LLM を使ってロボットのガイド

Planning Modulo LLM : 探索の中からLLMを使う

探索 : LLM で CDCL とかやると面白そう

現在あまりインターンは採用していないが、共同研究は可能!

MIT-IBM Watson AI Lab