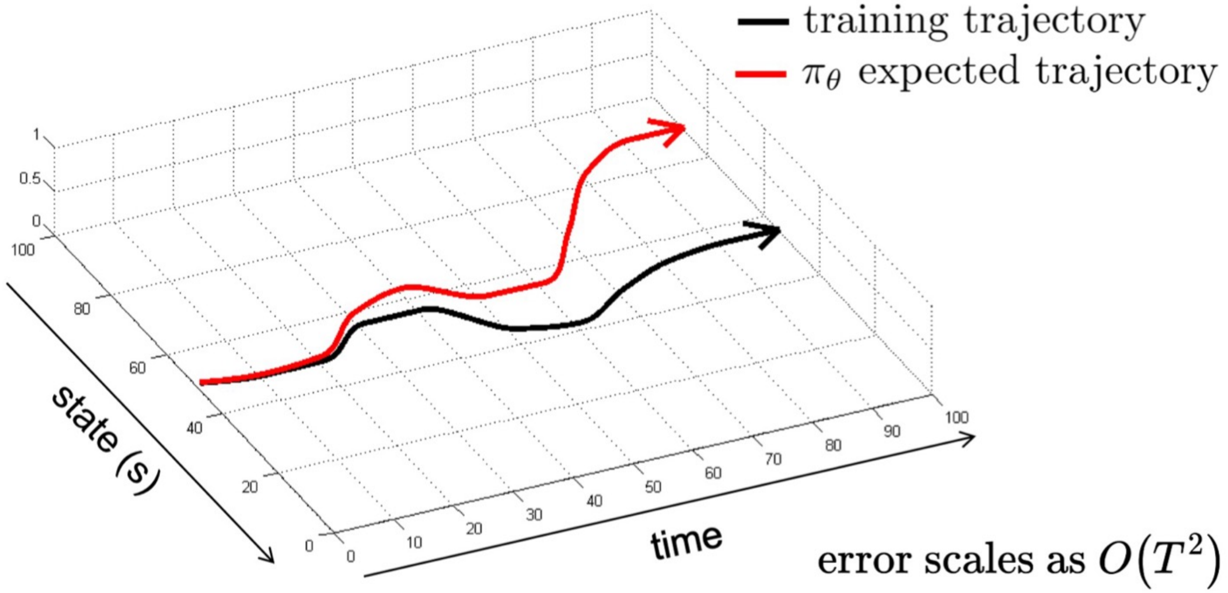
Hierarchical Diffusion for Offline Decision Making

Wenhao Li[1,2] , Xiangfeng Wang[*,3] , Bo Jin[*,4,5] , and Hongyuan Zha[1,2]

[*] Corresponding authors
[1] School of Data Science, The Chinese University of Hong Kong, Shenzhen, China.
[2] Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China.
[3] School of Computer Science and Technology, East China Normal University, Shanghai, China.
[4] School of Software Engineering, Tongji University, Shanghai, China.
[5] Shanghai Research Institute for Intelligent Autonomous Systems, Shanghai, China

OBJECTIVES

The lack of real-time interactions makes offline RL, in general, notoriously challenging, as value estimators can suffer from potential exponentially increasing variance in terms of the temporal horizon [1].



it is still an open problem to design an offline RL method that is less affected by the horizon and solves the long-horizon problem effectively.

OBJECTIVES

To deal with long temporal extension, an alternative RL solution is to decompose a long-horizon task into a hierarchy of subproblems, i.e., by hierarchical reinforcement learning (HRL), which also sees its utilization in offline decision-making. **Unfortunately, the instabilities of these methods due to the deadly triad, limited data access, and reward sparsity still remain largely unresolved.**

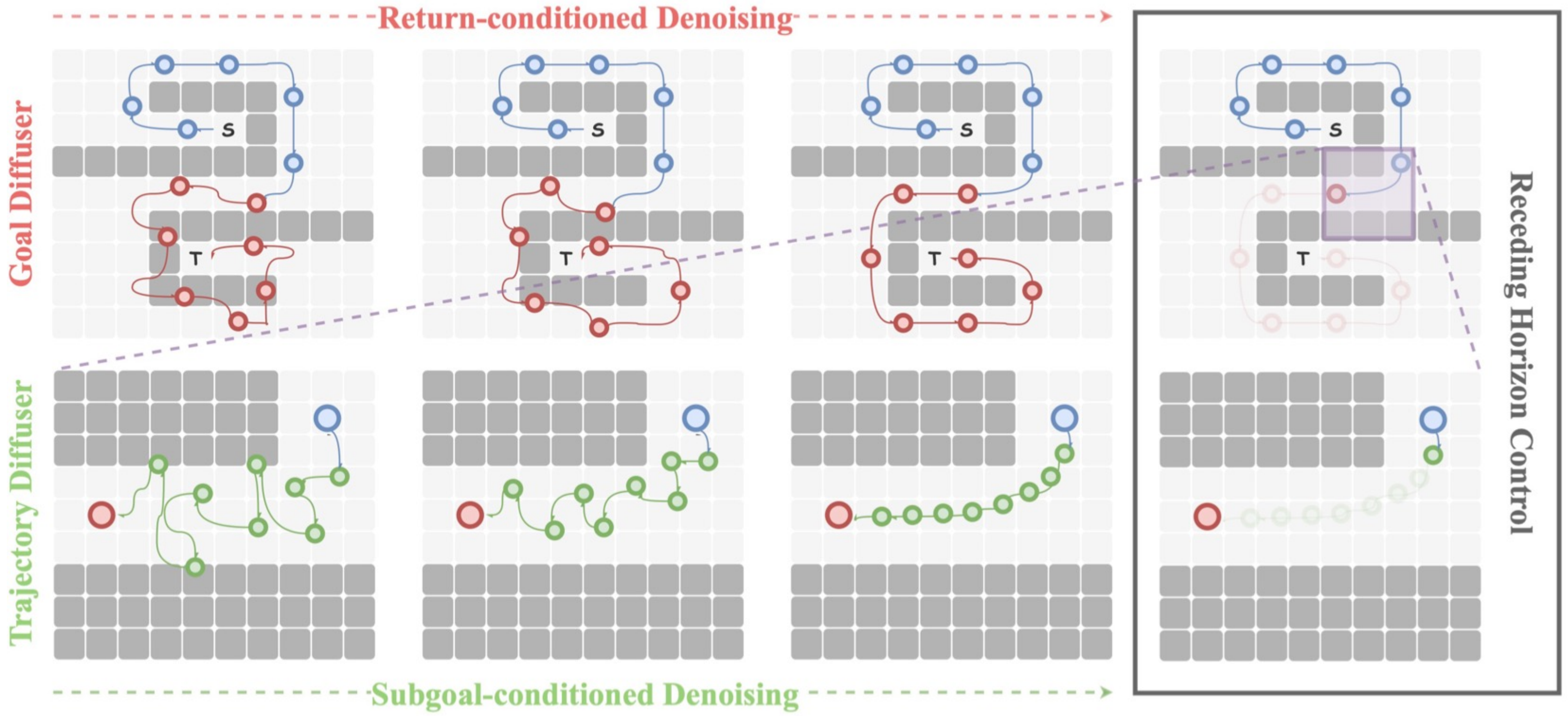
The success of RL methods leveraging conditional generative models on standardized benchmarks motivates us to consider the following question:

Can the above challenges be mitigated or even avoided using a conditional generation model that introduces a hierarchical structure?

CONTRIBUTIONS

- ❑ Introducing goals into the control-as-inference framework and formulating offline long-horizon decision-making as conditional generative modeling;
- ❑ Employing a hierarchical framework, HDMI, in which the goal diffuser learns a reward-conditional diffusion for the subgoal discovering, and the trajectory diffuser learns a goal-conditional diffusion for the action generation;
- ❑ Utilizing a planning-based subgoal extractor and transformer-based diffusion to deal with the sub-optimal data pollution and long-range subgoal dependency issues.

OVERALL ARCHITECTURE

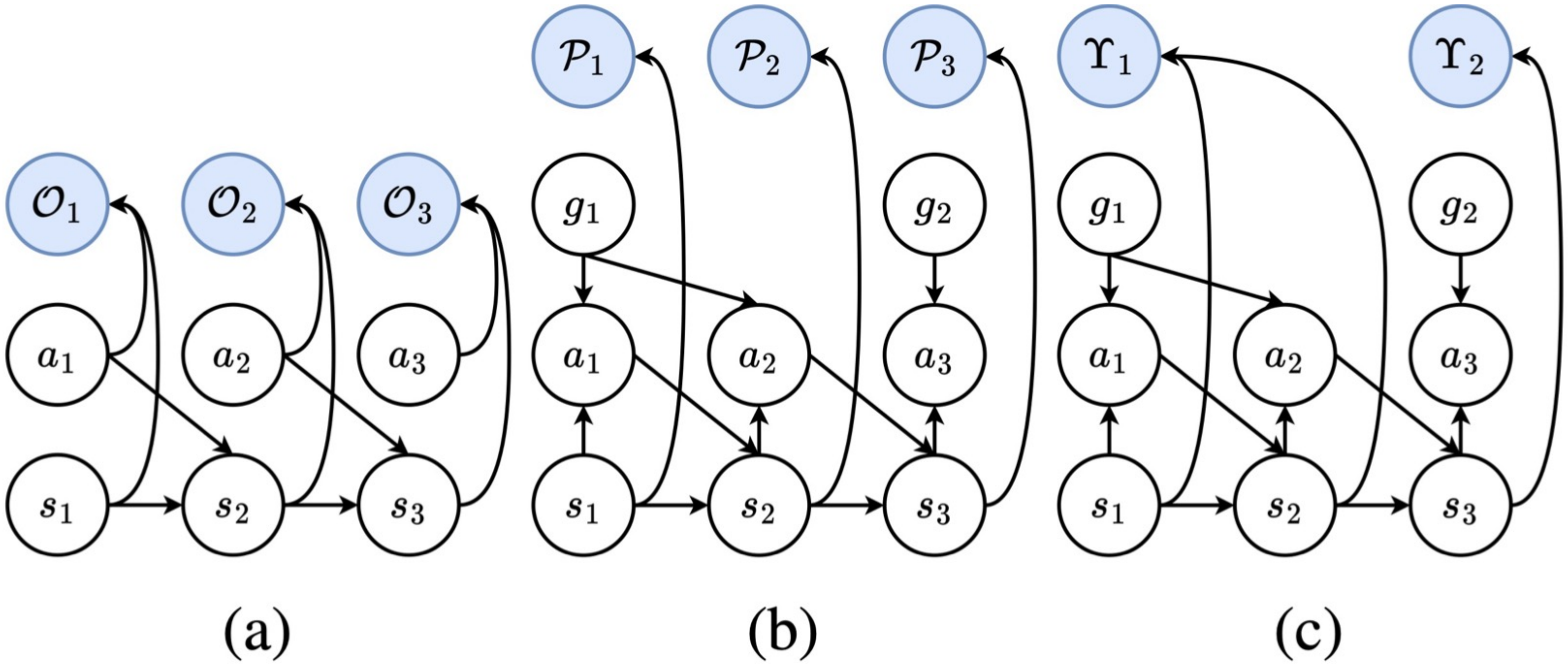


We first sample the subgoals based on the return and then sample the actions corresponding to the subgoal. We use receding horizon control to avoid error accumulation due to stochastics.

CONDITIONAL GENERATIVE MODELING

To take advantage of the generative model, we need to transform the offline long-horizon decision-making, i.e., obtaining the most probable subgoal and action distributions that maximize the expected return, into a goal-based, conditional generative modeling problem.

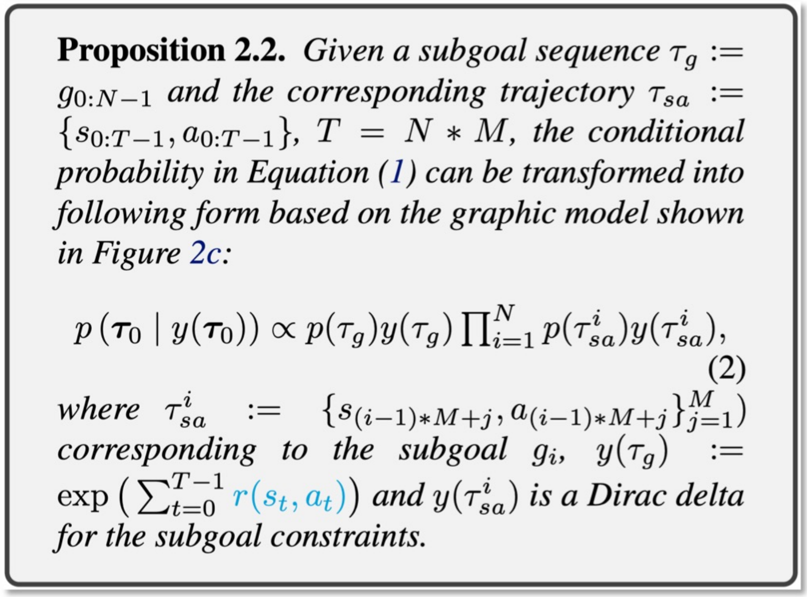
CONDITIONAL GENERATIVE MODELING



Graphic models for decision-making: (a) The states and actions form the backbone of the graphic model; (b) Augmenting goals to embed a long-horizon problem into this graphic model; (c) By summarizing the goal-augmented optimality variables, the new goal-oriented graphic model more naturally models the probability distribution of subgoals and highlights the hierarchical structure. Similar with [2], we condition the optimality variables as being **true** and then infer the most probable higher-level subgoal and lower-level action sequence or distributions.

CONDITIONAL GENERATIVE MODELING

With **Proposition 2.2**, we transform offline decision-making into the goal-based, conditional generative modeling with a hierarchical form. We will further introduce how to construct the training dataset, parameterize and train the model based on the dataset, and finally sample our interested subgoal and action sequences from it.



Proposition 2.2. *Given a subgoal sequence $\tau_g := g_{0:N-1}$ and the corresponding trajectory $\tau_{sa} := \{s_{0:T-1}, a_{0:T-1}\}$, $T = N * M$, the conditional probability in Equation (1) can be transformed into following form based on the graphic model shown in Figure 2c:*

$$p\left(\boldsymbol{\tau}_0 \mid y(\boldsymbol{\tau}_0)\right) \propto p(\tau_g)y(\tau_g)\prod_{i=1}^{N} p(\tau_{sa}^i)y(\tau_{sa}^i), \tag{2}$$

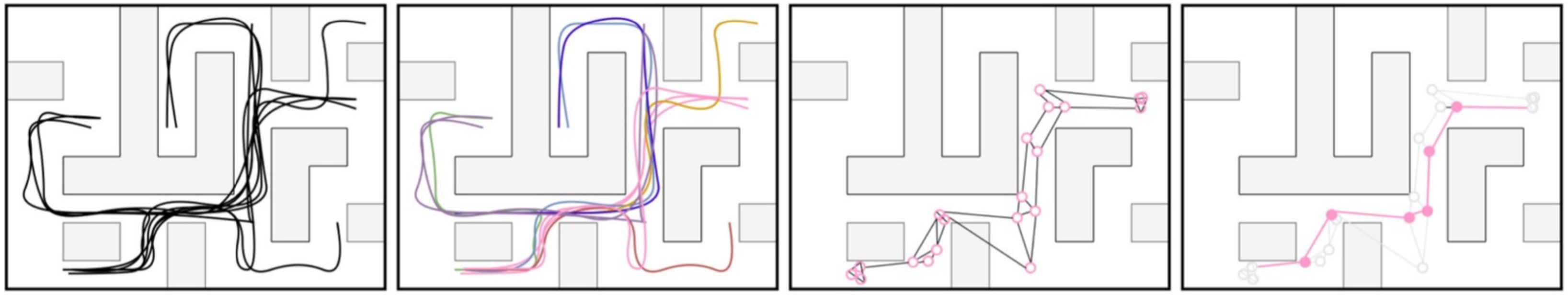
where $\tau_{sa}^i := \{s_{(i-1)*M+j}, a_{(i-1)*M+j}\}_{j=1}^{M})$ corresponding to the subgoal g_i, $y(\tau_g) := \exp\left(\sum_{t=0}^{T-1} r(s_t, a_t)\right)$ and $y(\tau_{sa}^i)$ is a Dirac delta for the subgoal constraints.

MERHOD (3 PARTS)

Unfortunately, in the offline dataset, no subgoals correspond to each trajectory in advance. This requires us to preprocess the dataset and extract high-quality subgoal sequences. The critical challenge is that suboptimal trajectories pollute the dataset. For example, in a goal-reaching task, the two trajectories to the goal may differ significantly in length. Thus extracting the corresponding subgoals from each trajectory independently will not guarantee optimality.

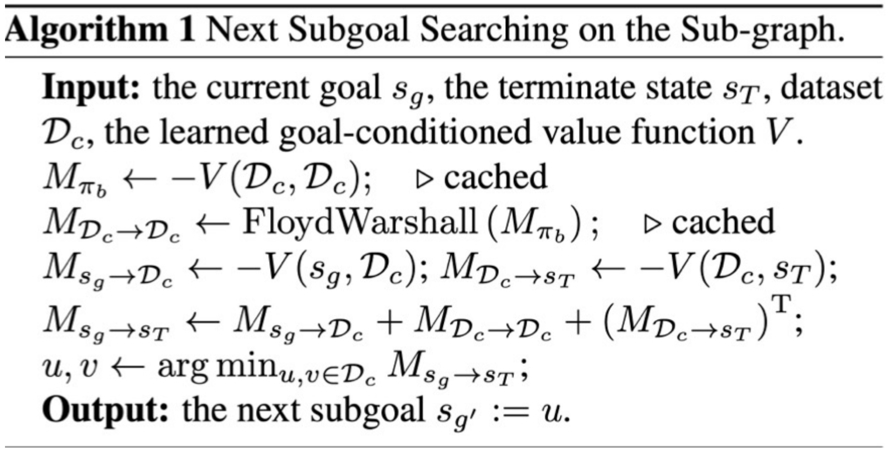
PART 1: GOAL EXTRACTION

We borrow a planning-based online RL method, SoRB [3], which can automatically find subgoals by learning graphic abstractions of the environment. This graph is constructed via an extra RL task, where a goal-conditioned value function provides edge weights, and nodes are taken to be observations. Using graph search to find the shortest path, we can automatically generate subgoals, even in high-dimensional environments.


(a) Initialization. (b) Clustering. (c) Graph Construction. (d) Planning.

PART 1: GOAL EXTRACTION

For the original dataset D, we first cluster all trajectories using the mini-batch k-means++ algorithm to aggregate trajectories with the same initial and the terminal state into a sub-dataset Dc; next, we transform Dc based on the distributional off-policy RL to a weighted directed graph Gc , where the nodes represent the states and the weights represent the predicted shortest distance;



Algorithm 1 Next Subgoal Searching on the Sub-graph.

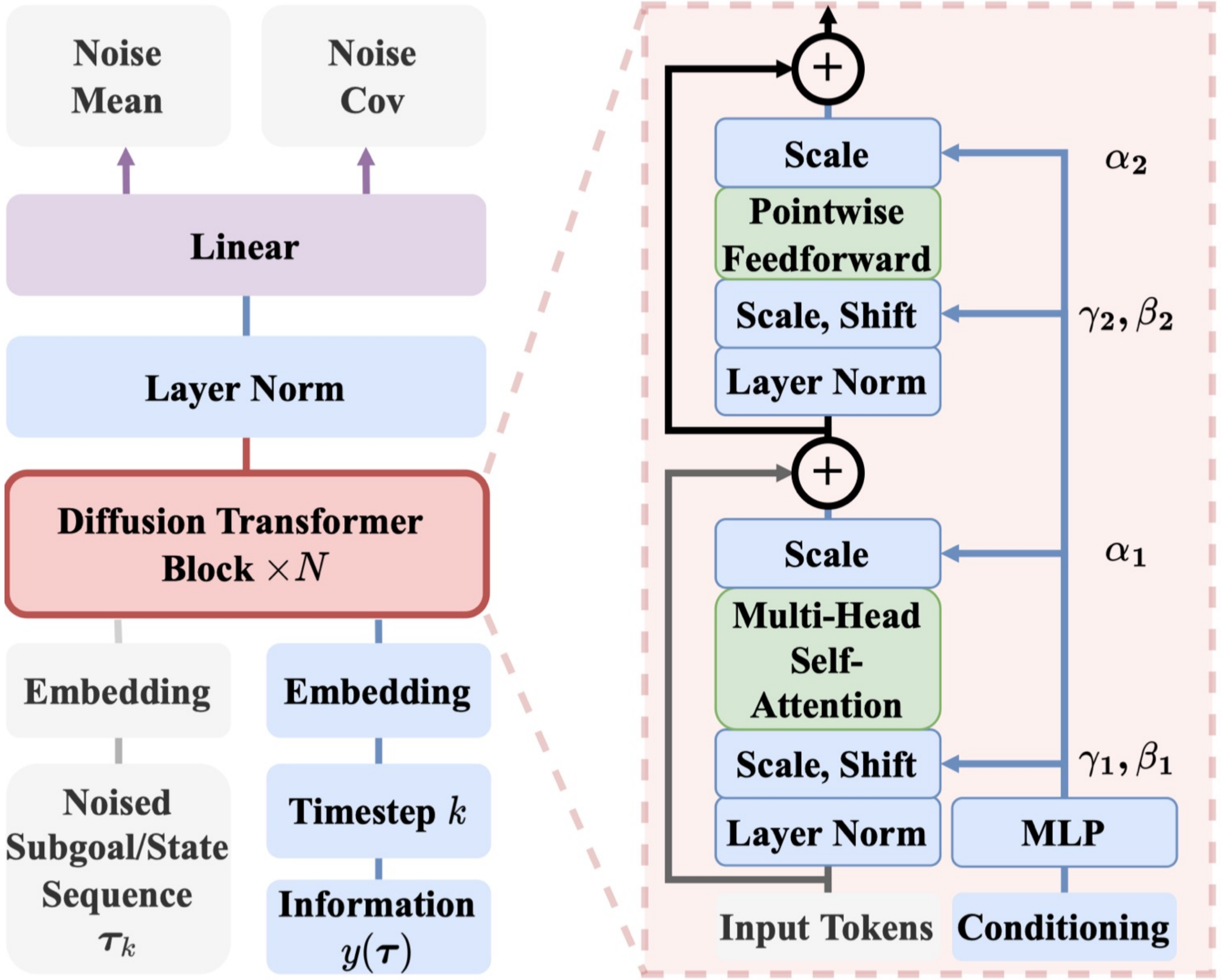
```
**Input:** the current goal $s_g$, the terminate state $s_T$, dataset $\mathcal{D}_c$, the learned goal-conditioned value function $V$.
$M_{\pi_b} \leftarrow -V(\mathcal{D}_c, \mathcal{D}_c)$;    ▷ cached
$M_{\mathcal{D}_c \to \mathcal{D}_c} \leftarrow \text{FloydWarshall}\,(M_{\pi_b})$;    ▷ cached
$M_{s_g \to \mathcal{D}_c} \leftarrow -V(s_g, \mathcal{D}_c)$; $M_{\mathcal{D}_c \to s_T} \leftarrow -V(\mathcal{D}_c, s_T)$;
$M_{s_g \to s_T} \leftarrow M_{s_g \to \mathcal{D}_c} + M_{\mathcal{D}_c \to \mathcal{D}_c} + (M_{\mathcal{D}_c \to s_T})^{\mathrm{T}}$;
$u, v \leftarrow \arg\min_{u,v \in \mathcal{D}_c} M_{s_g \to s_T}$;
**Output:** the next subgoal $s_{g'} := u$.
```

Finally, we use the proposed offline version of SoRB [3] to search for the shortest path from the initial to the terminal state on Gc and obtain the subgoal sequence.

PART 2: DIFFUSION MODEL PARAMETERIZATION

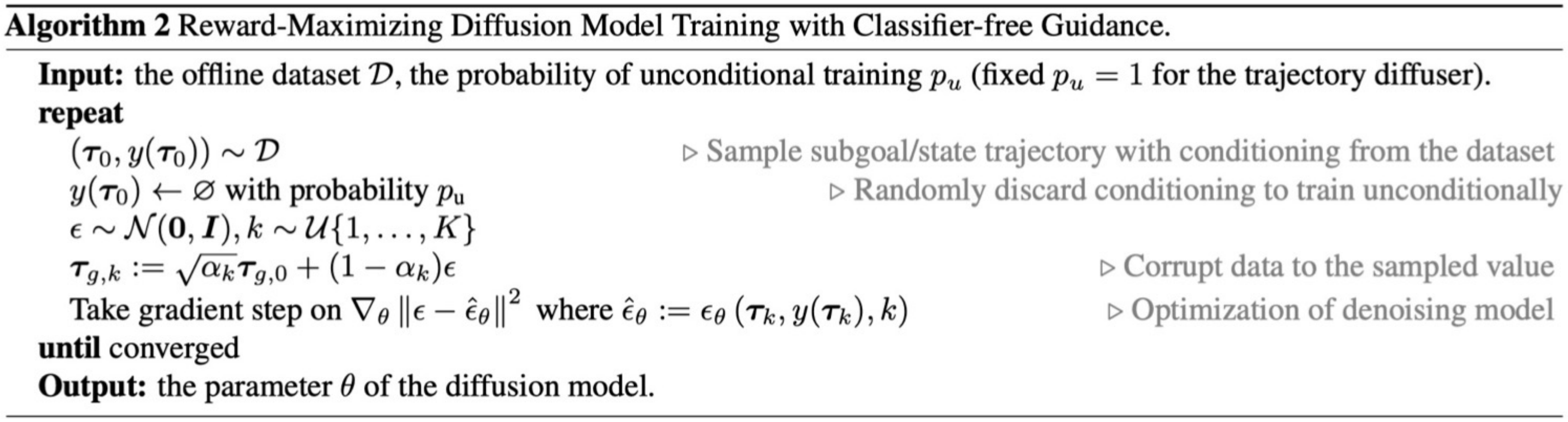
Existing work has shown that the capture of correlations between elements (e.g., image pixels or trajectory) is critical to the success of diffusion models. Different from related works that use the diffusion model to denoise the entire state trajectory or state-action trajectory, we denoise the **sparser** subgoal trajectory in the goal diffusion. The long-range dependence between subgoals makes the U-Net-like structure based on local convolution no longer the optimal choice. This motivates us to use the transformer [4] as the skeleton of the diffusion model.

PART 2: DIFFUSION MODEL PARAMETERIZATION



PART 3: HIERARCHICAL DIFFUSION

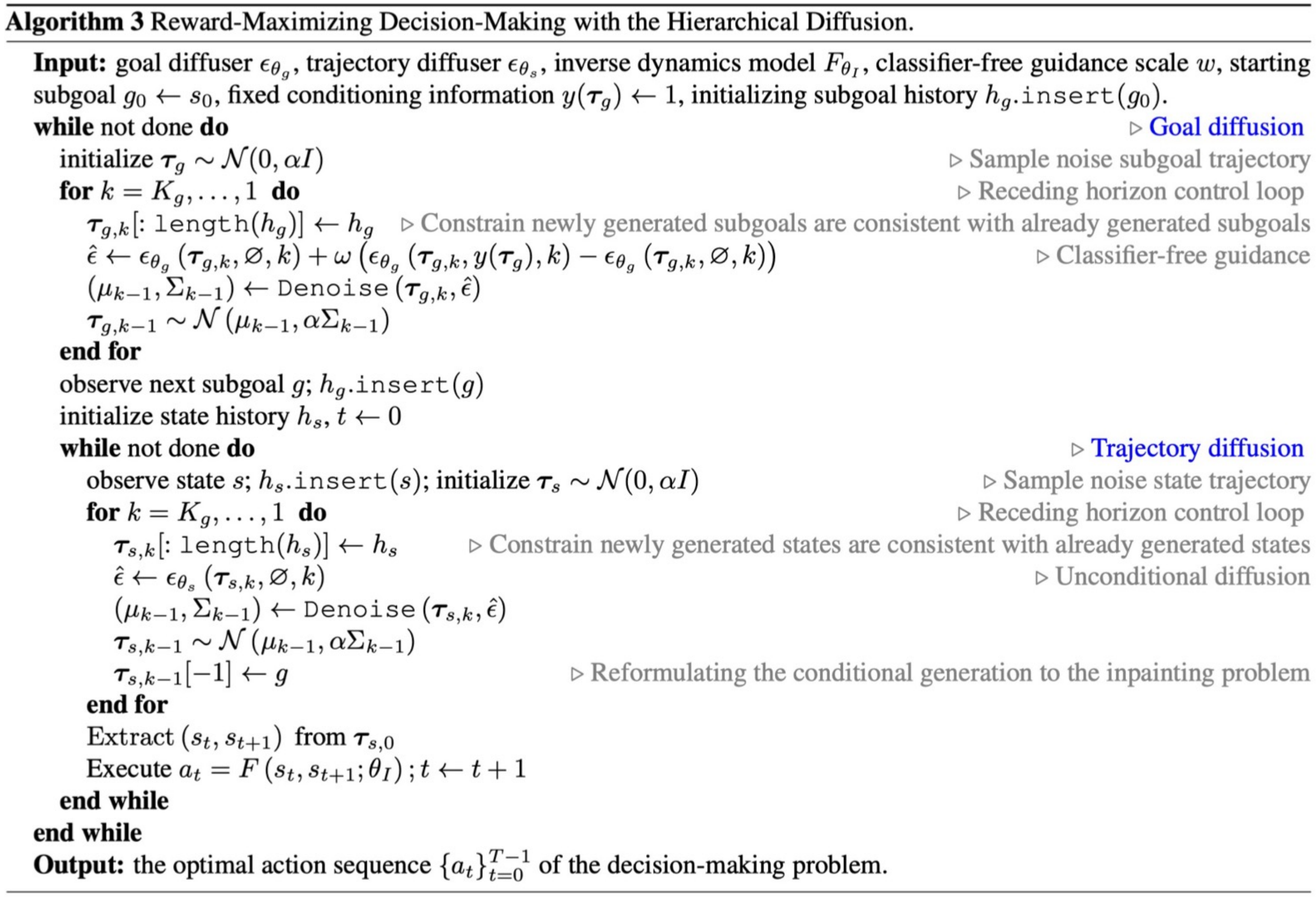
After the model is trained, sampling from HDMI is equivalent to complete the planning. Reward-maximizing decision-making with the hierarchical diffusion is shown in following, and for the goal-reaching task, only a simple modification is required, as shown in Algorithm 5 in the paper.



Algorithm 2 Reward-Maximizing Diffusion Model Training with Classifier-free Guidance.

```
Input: the offline dataset $\mathcal{D}$, the probability of unconditional training $p_u$ (fixed $p_u = 1$ for the trajectory diffuser).
repeat
    $(\boldsymbol{\tau}_0, y(\boldsymbol{\tau}_0)) \sim \mathcal{D}$    ▷ Sample subgoal/state trajectory with conditioning from the dataset
    $y(\boldsymbol{\tau}_0) \leftarrow \varnothing$ with probability $p_{\mathrm{u}}$    ▷ Randomly discard conditioning to train unconditionally
    $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), k \sim \mathcal{U}\{1, \ldots, K\}$
    $\boldsymbol{\tau}_{g,k} := \sqrt{\alpha_k}\boldsymbol{\tau}_{g,0} + (1 - \alpha_k)\epsilon$    ▷ Corrupt data to the sampled value
    Take gradient step on $\nabla_\theta \left\|\epsilon - \hat{\epsilon}_\theta\right\|^2$ where $\hat{\epsilon}_\theta := \epsilon_\theta(\boldsymbol{\tau}_k, y(\boldsymbol{\tau}_k), k)$    ▷ Optimization of denoising model
until converged
Output: the parameter $\theta$ of the diffusion model.
```

PART 3: HIERARCHICAL DIFFUSION



Algorithm 3 Reward-Maximizing Decision-Making with the Hierarchical Diffusion.

```
Input: goal diffuser $\epsilon_{\theta_g}$, trajectory diffuser $\epsilon_{\theta_s}$, inverse dynamics model $F_{\theta_I}$, classifier-free guidance scale $w$, starting
subgoal $g_0 \leftarrow s_0$, fixed conditioning information $y(\boldsymbol{\tau}_g) \leftarrow 1$, initializing subgoal history $h_g.\texttt{insert}(g_0)$.
while not done do                                                          ▷ Goal diffusion
  initialize $\boldsymbol{\tau}_g \sim \mathcal{N}(0, \alpha I)$                                      ▷ Sample noise subgoal trajectory
  for $k = K_g, \ldots, 1$ do                                               ▷ Receding horizon control loop
    $\boldsymbol{\tau}_{g,k}[:\texttt{length}(h_g)] \leftarrow h_g$   ▷ Constrain newly generated subgoals are consistent with already generated subgoals
    $\hat{\epsilon} \leftarrow \epsilon_{\theta_g}(\boldsymbol{\tau}_{g,k}, \varnothing, k) + \omega\left(\epsilon_{\theta_g}(\boldsymbol{\tau}_{g,k}, y(\boldsymbol{\tau}_g), k) - \epsilon_{\theta_g}(\boldsymbol{\tau}_{g,k}, \varnothing, k)\right)$   ▷ Classifier-free guidance
    $(\mu_{k-1}, \Sigma_{k-1}) \leftarrow \texttt{Denoise}(\boldsymbol{\tau}_{g,k}, \hat{\epsilon})$
    $\boldsymbol{\tau}_{g,k-1} \sim \mathcal{N}(\mu_{k-1}, \alpha\Sigma_{k-1})$
  end for
  observe next subgoal $g$; $h_g.\texttt{insert}(g)$
  initialize state history $h_s$, $t \leftarrow 0$
  while not done do                                                        ▷ Trajectory diffusion
    observe state $s$; $h_s.\texttt{insert}(s)$; initialize $\boldsymbol{\tau}_s \sim \mathcal{N}(0, \alpha I)$          ▷ Sample noise state trajectory
    for $k = K_g, \ldots, 1$ do                                             ▷ Receding horizon control loop
      $\boldsymbol{\tau}_{s,k}[:\texttt{length}(h_s)] \leftarrow h_s$     ▷ Constrain newly generated states are consistent with already generated states
      $\hat{\epsilon} \leftarrow \epsilon_{\theta_s}(\boldsymbol{\tau}_{s,k}, \varnothing, k)$                                  ▷ Unconditional diffusion
      $(\mu_{k-1}, \Sigma_{k-1}) \leftarrow \texttt{Denoise}(\boldsymbol{\tau}_{s,k}, \hat{\epsilon})$
      $\boldsymbol{\tau}_{s,k-1} \sim \mathcal{N}(\mu_{k-1}, \alpha\Sigma_{k-1})$
      $\boldsymbol{\tau}_{s,k-1}[-1] \leftarrow g$          ▷ Reformulating the conditional generation to the inpainting problem
    end for
    Extract $(s_t, s_{t+1})$ from $\boldsymbol{\tau}_{s,0}$
    Execute $a_t = F(s_t, s_{t+1}; \theta_I)$; $t \leftarrow t + 1$
  end while
end while
Output: the optimal action sequence $\{a_t\}_{t=0}^{T-1}$ of the decision-making problem.
```

NUMERICAL RESULTS (PART)

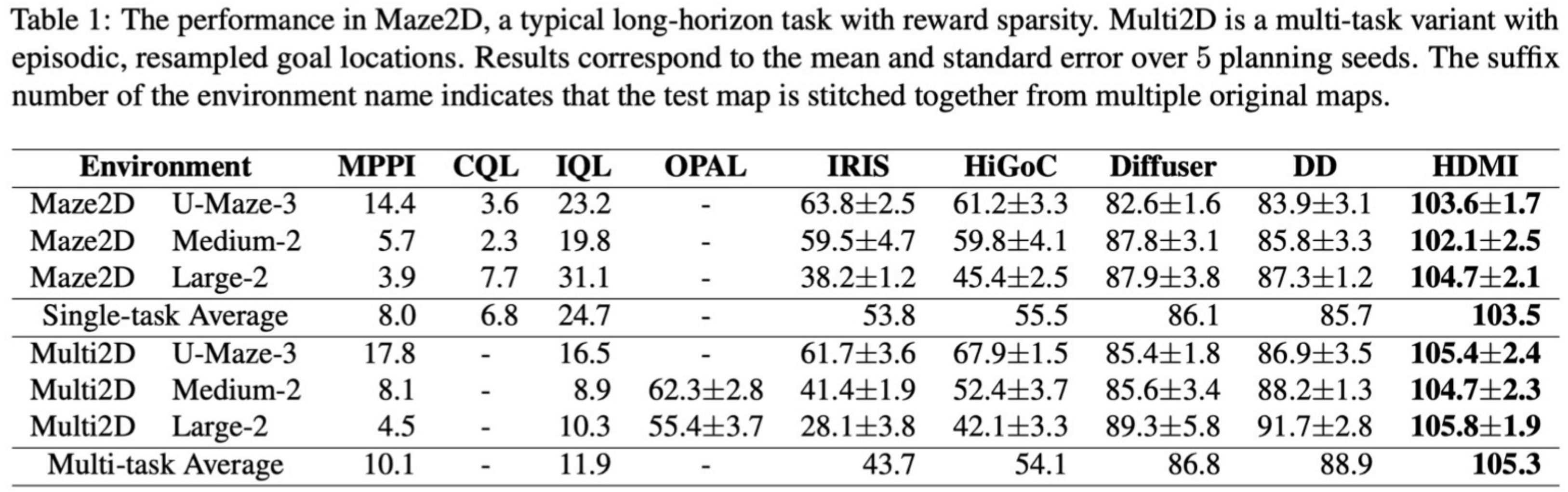


Table 1: The performance in Maze2D, a typical long-horizon task with reward sparsity. Multi2D is a multi-task variant with episodic, resampled goal locations. Results correspond to the mean and standard error over 5 planning seeds. The suffix number of the environment name indicates that the test map is stitched together from multiple original maps.

Environment		MPPI	CQL	IQL	OPAL	IRIS	HiGoC	Diffuser	DD	HDMI
Maze2D	U-Maze-3	14.4	3.6	23.2	-	63.8±2.5	61.2±3.3	82.6±1.6	83.9±3.1	**103.6±1.7**
Maze2D	Medium-2	5.7	2.3	19.8	-	59.5±4.7	59.8±4.1	87.8±3.1	85.8±3.3	**102.1±2.5**
Maze2D	Large-2	3.9	7.7	31.1	-	38.2±1.2	45.4±2.5	87.9±3.8	87.3±1.2	**104.7±2.1**
Single-task Average		8.0	6.8	24.7	-	53.8	55.5	86.1	85.7	**103.5**
Multi2D	U-Maze-3	17.8	-	16.5	-	61.7±3.6	67.9±1.5	85.4±1.8	86.9±3.5	**105.4±2.4**
Multi2D	Medium-2	8.1	-	8.9	62.3±2.8	41.4±1.9	52.4±3.7	85.6±3.4	88.2±1.3	**104.7±2.3**
Multi2D	Large-2	4.5	-	10.3	55.4±3.7	28.1±3.8	42.1±3.3	89.3±5.8	91.7±2.8	**105.8±1.9**
Multi-task Average		10.1	-	11.9	-	43.7	54.1	86.8	88.9	**105.3**

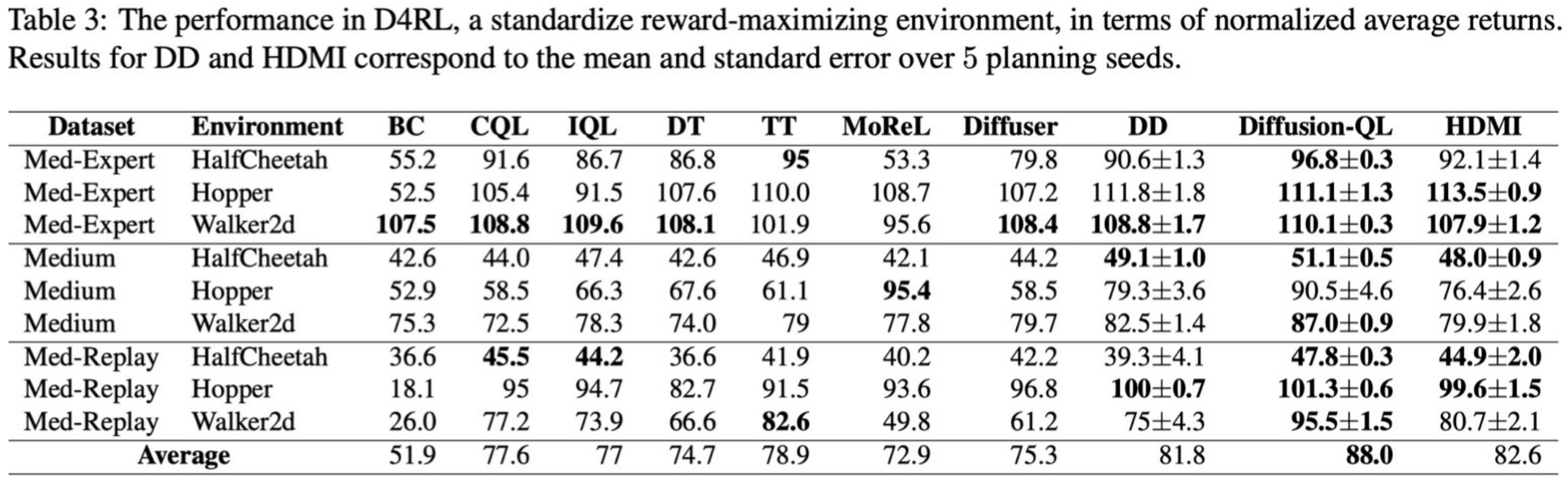


Table 3: The performance in D4RL, a standardize reward-maximizing environment, in terms of normalized average returns. Results for DD and HDMI correspond to the mean and standard error over 5 planning seeds.

Dataset	Environment	BC	CQL	IQL	DT	TT	MoReL	Diffuser	DD	Diffusion-QL	HDMI
Med-Expert	HalfCheetah	55.2	91.6	86.7	86.8	**95**	53.3	79.8	90.6±1.3	**96.8±0.3**	92.1±1.4
Med-Expert	Hopper	52.5	105.4	91.5	107.6	110.0	108.7	107.2	111.8±1.8	**111.1±1.3**	**113.5±0.9**
Med-Expert	Walker2d	**107.5**	**108.8**	**109.6**	**108.1**	101.9	95.6	**108.4**	**108.8±1.7**	**110.1±0.3**	**107.9±1.2**
Medium	HalfCheetah	42.6	44.0	47.4	42.6	46.9	42.1	44.2	**49.1±1.0**	**51.1±0.5**	**48.0±0.9**
Medium	Hopper	52.9	58.5	66.3	67.6	61.1	**95.4**	58.5	79.3±3.6	90.5±4.6	76.4±2.6
Medium	Walker2d	75.3	72.5	78.3	74.0	79	77.8	79.7	82.5±1.4	**87.0±0.9**	79.9±1.8
Med-Replay	HalfCheetah	36.6	**45.5**	**44.2**	36.6	41.9	40.2	42.2	39.3±4.1	**47.8±0.3**	**44.9±2.0**
Med-Replay	Hopper	18.1	95	94.7	82.7	91.5	93.6	96.8	**100±0.7**	**101.3±0.6**	**99.6±1.5**
Med-Replay	Walker2d	26.0	77.2	73.9	66.6	**82.6**	49.8	61.2	75±4.3	**95.5±1.5**	80.7±2.1
Average		51.9	77.6	77	74.7	78.9	72.9	75.3	81.8	**88.0**	82.6

NUMERICAL RESULTS (PART)

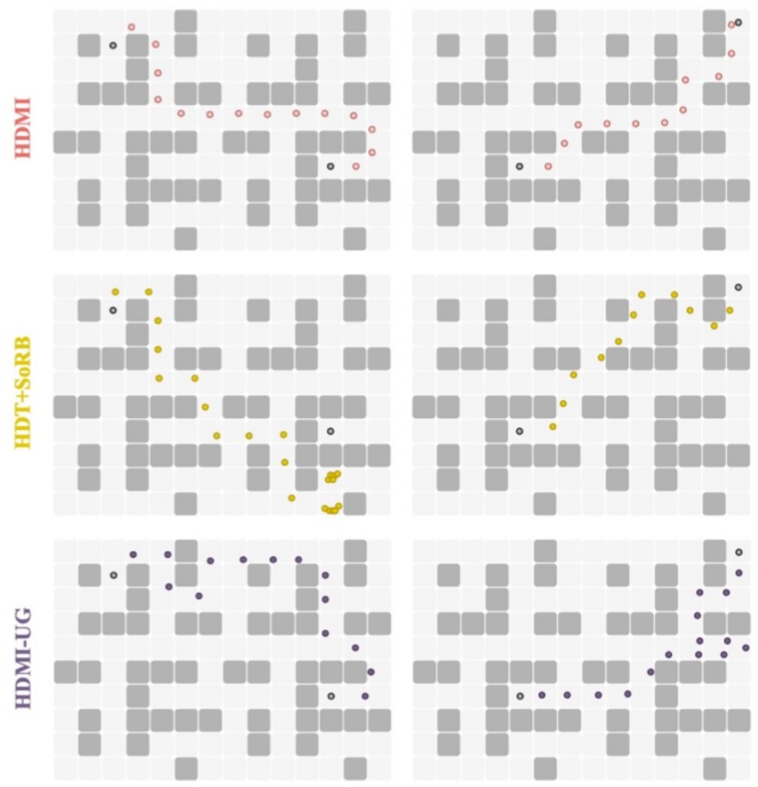


Figure 6: Subgoal sampled by baselines in `Large-2`

Table 4: The performance of HDMI and baselines in FinRL, a realistic long-horizon reward-maximizing environment. Results are measured same as (Qin et al., 2022).

Dataset	Det.	BC	CQL	MB-PPO	DD	HDMI
FinRL-L-99	150	136	**487**	328	372	415
FinRL-L-999	150	137	416	656	721	**733**
FinRL-M-99	300	355	700	**1213**	830	1007
FinRL-M-999	300	504	621	698	712	**754**
FinRL-H-99	441	252	**671**	484	609	658
FinRL-H-999	441	270	444	787	782	**801**

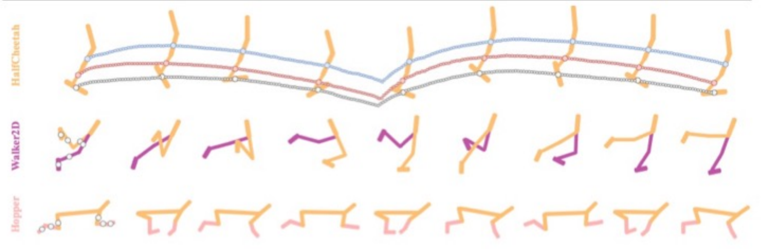


Figure 5: Visualization of partial subgoals and states sampled by HDMI in D4RL. Small circles indicate states, while large circles indicate subgoals. For the sake of clear presentation, we downsample the sequence and only visualized the state sequence sampled in the `Hopper` environment. The action sequences are not directly shown.

REFERENCES

[1] Ren, T. et, al. Nearly horizon-free offline reinforcement learning. In NeurIPS, 2021.

[2] Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv preprint, 2018.

[3] Eysenbach, B, et, al. Search on the replay buffer: Bridging planning and reinforcement learning. In NeurIPS, 2019.

[4] Peebles, W. and Xie, S. Scalable diffusion models with transformers. arXiv preprint, 2022.