Online Skill Discovery using Graph-based Clustering

Jan Hendrik Metzen, University Bremen, Robotics Group, Robert-Hooke-Str. 5, 28359 Bremen, Germany

1 Abstract

We introduce a novel method for skill discovery using the **bottleneck principle** which is based on **bottom-up hierarchical clustering** of the estimated transition graph. In contrast to prior clustering approaches, it can be used **incrementally** and thus several times during the learning process. Furthermore, we show that the choice of the **linkage criterion** is crucial for dealing with non-random sampling policies and stochastic environments.

2 Skill Discovery

- Skill Discovery is one of the major challenges in Hierarchical RL
- Skills should be reusable, distinct, and easy to learn
- Frequency-based approaches (e.g. [2, 3]) and graph-based approaches (e.g. [1])

3 Graph-based Clustering

Graph construction: Based on a multi-set of transitions $T = \{(s_i, a_i, s_i')\}_{i=1...n}$, construct a graph node v for any observed state and an edge $e = (s_i, s_i')$ for each observed state transition.

Edge weight: Annotate each edge with a weight, based on the multiplicity $N(s, a, s')$ of (s, a, s') in T.

- Uniform weights: $w_{uni}((s, s')) = 1$
- On-Policy weights: $w_{on}((s, s')) = \sum_a N(s, a, s')$
- Off-Policy weights: $w_{off}((s, s')) = \sum_a N(s, a, s')/N(s, a)$

Linkage criteria determine to which extent the boundary of two connected, disjoint subgraphs $A, B \subset G$ forms a bottleneck in $G = (V, E, w)$.
Let $c(A, B) = \sum_{e \in E \cap (A \times B)} w(e)$.
Linkage criterion l:

- $M(A, B) = \frac{\min(|A|,|B|)\log(\max(|A|,|B|))}{c(A,B)+c(B,A)}$, (Mannor et al. [1], w_{uni})
- $\hat{N}_{cut}(A, B) = \frac{c(A,B)+c(B,A)}{c(A,V)+c(B,A)} + \frac{c(B,A)+c(A,B)}{c(B,V)+c(A,B)}$, (Şimşek et al. [4], w_{on} or w_{off})

Clustering: Using agglomerative clustering to find close-to-optimal solution for

$$P^* = \arg\min_{P \in \mathcal{P}(V)} |P| \quad \text{s.t.} \quad \max_{p_i \in P, q_i \subset p_i} l(p_i \setminus q_i, q_i) \leq \psi.$$

Algorithm 1 Constrained agglomerative clustering

```
Input: graph G = (V, E, w), constraint set C, linkage criterion l, threshold ψ
Initialize: partition P = {{v}|v ∈ V}
C = C ∪ lambda p1, p2 : (p1 × p2) ∩ E ≠ ∅  # Merge only clusters p_i that are connected in G
loop
    M = {(p1, p2)|(p1, p2) ∈ (P × P) ∧ ⋀_{c∈C} c(p1, p2)}  # Merge-candidates fulfilling constraints
    p1*, p2* = arg min_{p1,p2∈M} l(p1, p2)  # Find merge candidates with minimal linkage
    if l(p1*, p2*) > ψ: return P
    P = (P \ {p1*, p2*}) ∪ {p1* ∪ p2*}  # Merge p1* and p2*
end loop
```

4 OGAHC

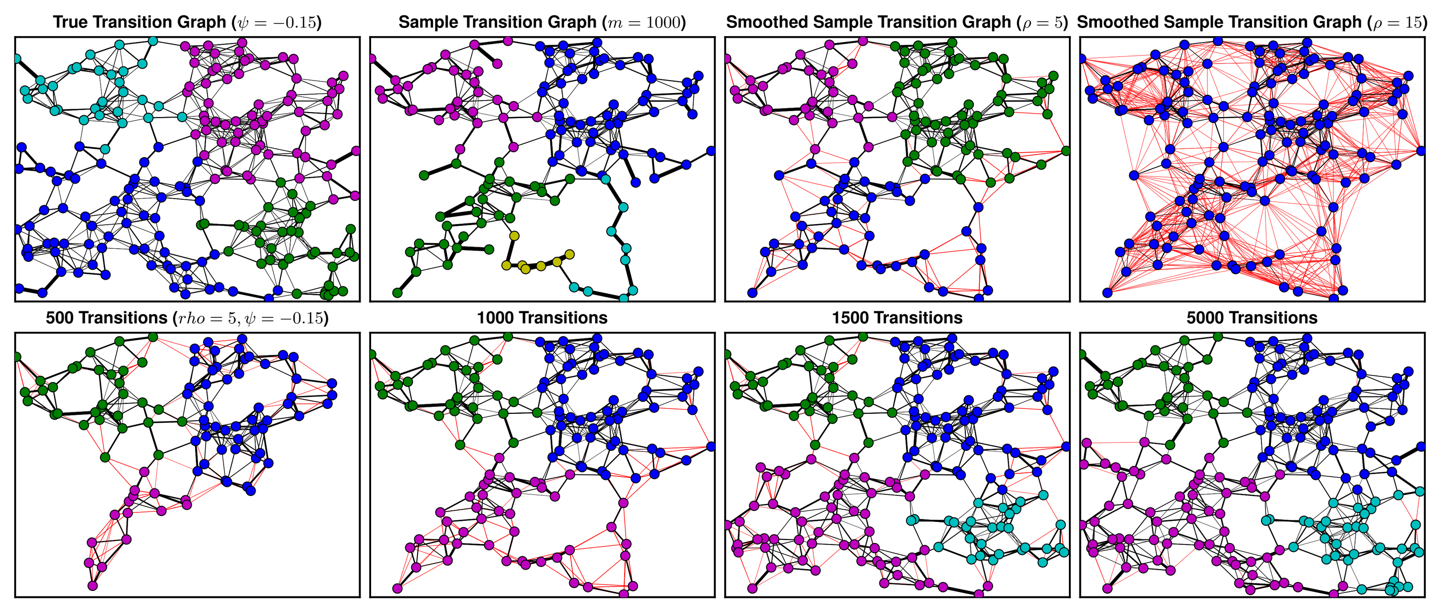
Motivation: Skill discovery should be incremental, i.e. should be conducted several times during learning. For this purpose, OGAHC combines constrained agglomerative clustering with graph smoothing.

Algorithm 2 OGAHC

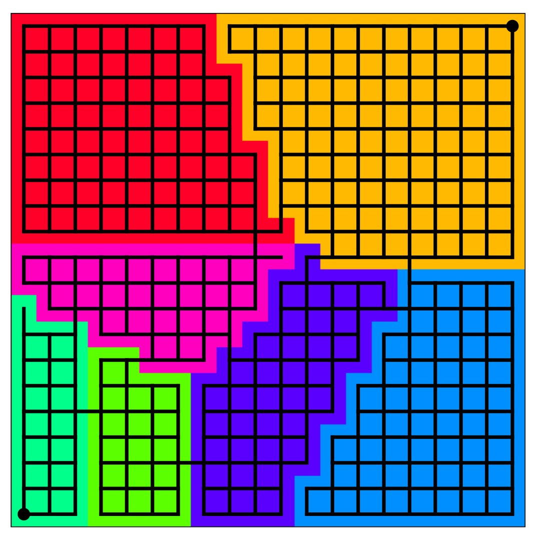
```
Input: linkage criterion l, parameters ρ, ψ, m
Initialize: partition P = ∅, graph G = (∅, ∅),
loop
    (s1, a1, s2, ..., a_{m-1}, s_m) = ACT(AGENT, ENV)  # Observe trajectory of m − 1 steps
    UPDATE(G, (s1, a1, s2, ..., a_{m-1}, s_m))  # Update transition graph with trajectory
    G' = SMOOTH(G, ρ)  # Pseudo transitions for under-explored nodes
    C = ∅  # Constraints for keeping partitions consistent
    for (p_A, p_B) ∈ P × P with p_A ≠ p_B do
        # Must not merge two clusters with elements that had not been merged in last iteration
        C = C ∪ {lambda p1, p2 : (p1 ∪ p2) ∩ p_A = ∅ ∨ (p1 ∪ p2) ∩ p_B = ∅}
    end for
    P = CAC(G', C, l, ψ)  # Partition G' using constrained agglomerative clustering (CAC)
end loop
```

Smoothing: For each s, a with with less than ρ samples $N(s, a)$, a pseudo transition of weight $(\rho - N(s, a))/k$ is added to any of the $k = max(5, \rho)$ nearest neighbors of s. *"Assume dense local connectivity in the face of uncertainty"*.

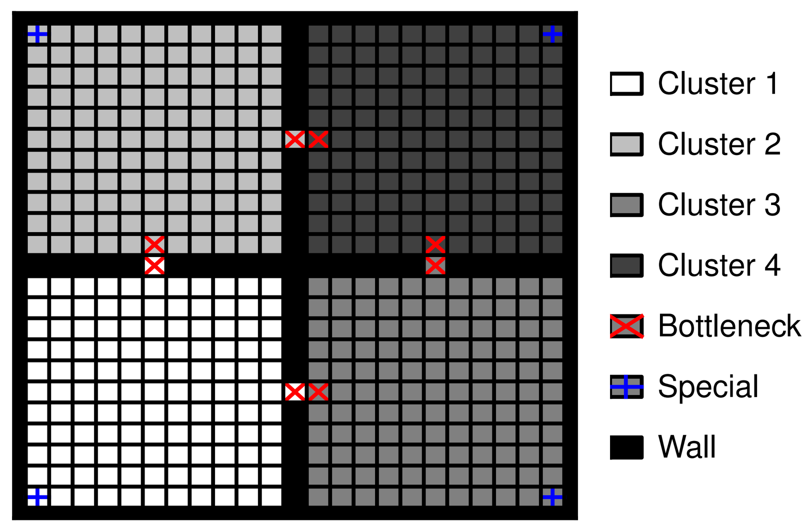
5 Illustration



6 Scenarios



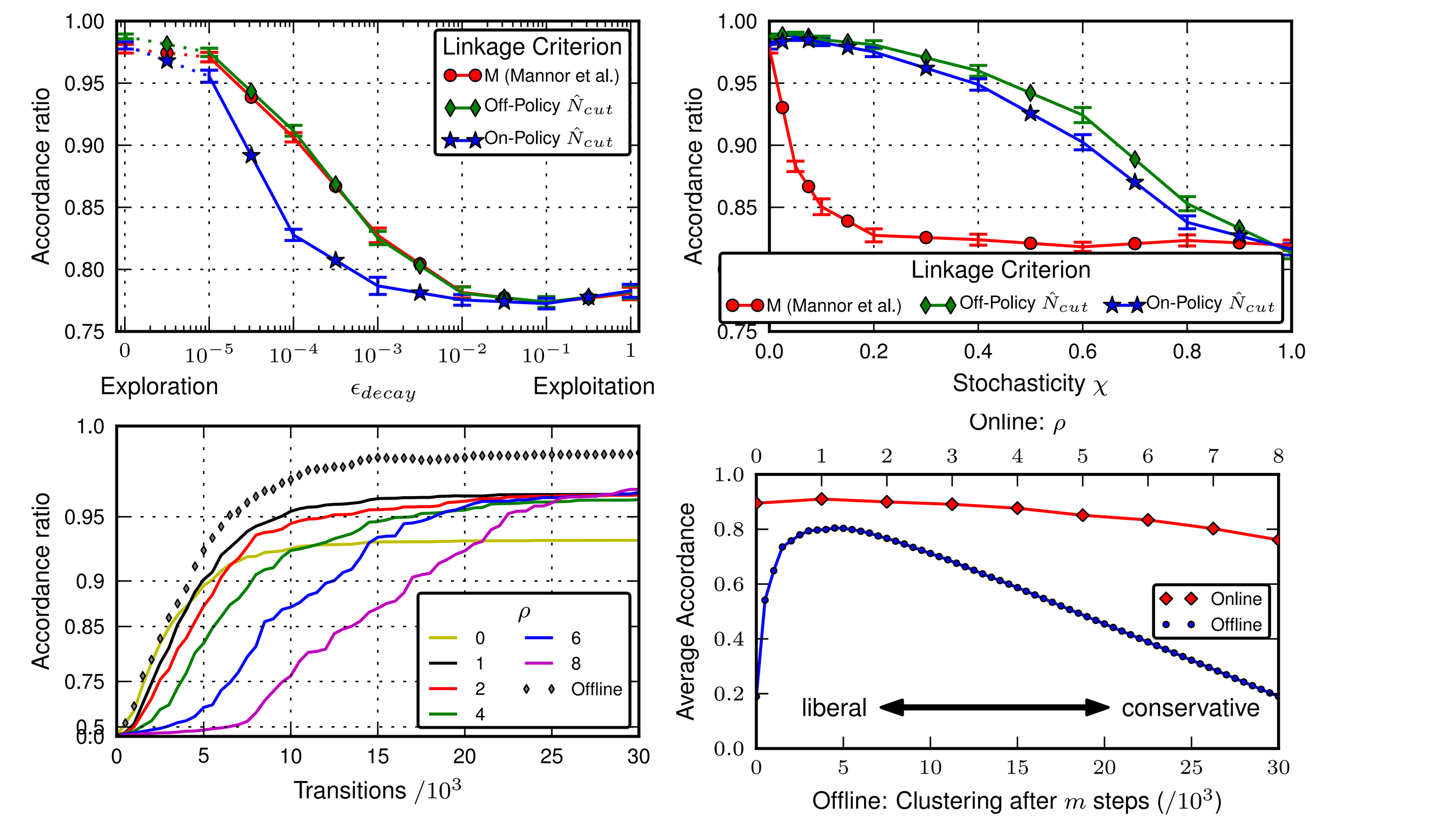
Random graphs: 50 random graphs consisting of 400 states and 7 ground-truth clusters



Multi-task Maze: Simple 23×23 maze world consisting of 4 rooms and 12 different tasks. Shown is a baseline clustering of the domain used for "predefining" skills.

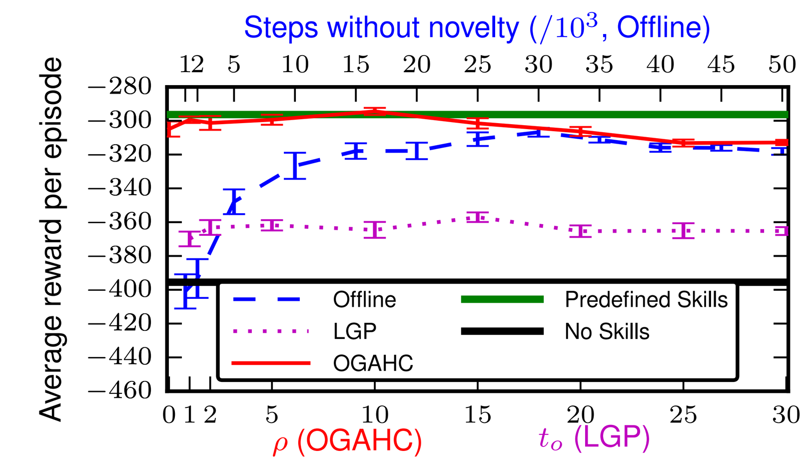
7 Results

Results on Random Graphs



Accordance: $acc(P_1, P_2) = \frac{1}{|S|^2} \sum_{s,\bar{s} \in S} \delta(\delta(P_1(s), P_1(\bar{s})), \delta(P_2(s), P_2(\bar{s})))$

Results on Multi-Task Maze



Average reward per episode during 1000 learning episodes. Comparison to:

- Offline Graph Clustering [1]
- Local Graph Partitioning [4]
- Predefined Skills
- No Skill Learning

8 Outlook

- Evaluation in large and continuous MDPs (graph construction)
- Other smoothing heuristics and graph clustering approaches
- Analysis of computational complexity

9 References

[1] S. Mannor, I. Menache, A. Hoze, and U. Klein.
Dynamic abstraction in reinforcement learning via clustering.
In *Proceedings of the 21st International Conference on Machine Learning*, pages 560–567, 2004.

[2] A. McGovern and A. G. Barto.
Automatic discovery of subgoals in reinforcement learning using diverse density.
In *In Proceedings of the 18th International Conference on Machine Learning*, pages 361–368, 2001.

[3] Ö. Şimşek and A. G. Barto.
Using relative novelty to identify useful temporal abstractions in reinforcement learning.
In *Proceedings of the 21st International Conference on Machine Learning*, pages 751–758, 2004.

[4] Ö. Şimşek, A. P. Wolfe, and A. G. Barto.
Identifying useful subgoals in reinforcement learning by local graph partitioning.
In *Proceedings of the 22nd International Conference on Machine Learning*, pages 816–823. ACM, 2005.

Universität Bremen

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH
Robotics Innovation Center

Contact:
Jan Hendrik Metzen
University Bremen, Robotics Group
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
jhm@informatik.uni-bremen.de