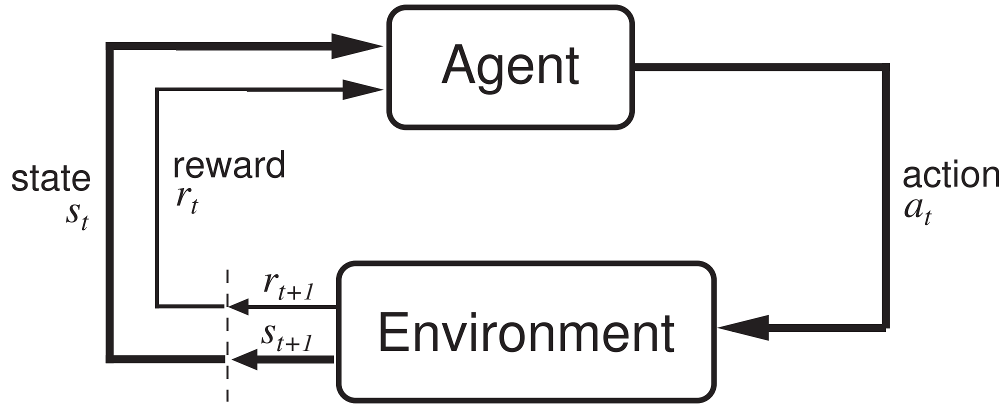
Automatic Abstraction in Reinforcement Learning Using Ant System Algorithm

Mohsen Ghafoorian
Intelligent Systems Laboratory,
Sharif University of Technology

Index

- Reinforcement Learning
- Hierarchical Reinforcement Learning
- Proposed Method
  - Transition Graph Creation
  - Sub-goal Identification
  - Skill Creation
- Experimental Results
  - Taxi Environment
  - Playroom Environment
- References

Reinforcement Learning



- Action Selection based on agent policy:
- Agent's goal:
  - Maximizing cumulative discounted reward:

$$\pi : S \times A \rightarrow [0, 1]$$

$$R = E\left\{\sum_{t=0}^{\infty} \gamma^t r_t\right\}$$

- Approximation of Action-value function:

$$\hat{Q}(s,a) = (1-\alpha)\hat{Q}(s,a) + \alpha\left[R(s,a) + \gamma \max_{a' \in A} \hat{Q}(s',a')\right]$$

Q Learning Algorithm:

Input: (α, γ)

Initialize $\widehat{Q}(s, a)$ randomly

Repeat for each episode

indicate policy π using $\widehat{Q}$

initialize s

Repeat

choose action a considering policy π

do action a and observe reward $R(s, a)$ and next state s'

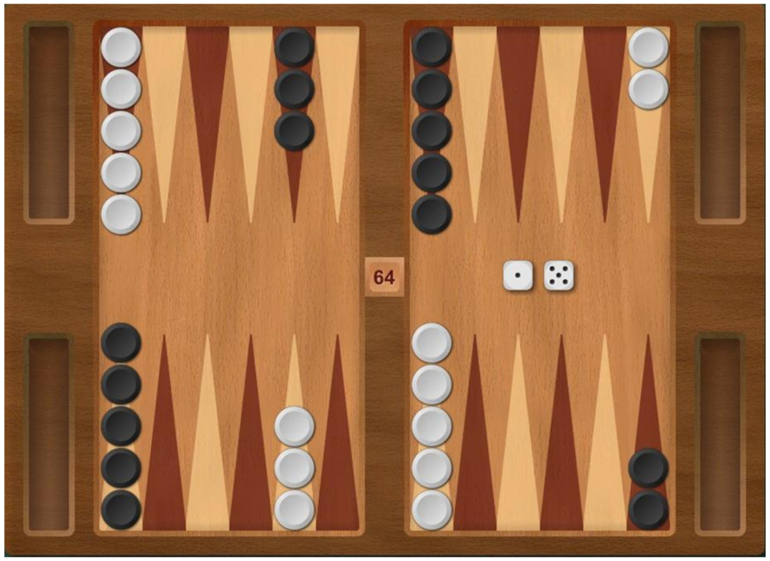
$$\widehat{Q}(s, a) \leftarrow (1 - \alpha)\widehat{Q}(s, a) + \alpha \left[R(s, a) + \gamma \max_{a' \in A} \widehat{Q}(s', a') \right]$$

until s is a final state.

Reinforcement Learning

Back gammon Environment:

- Number of states: about 10^{20}
- Number of learning episodes until convergence: 1.5 millions!



Hierarchical Reinforcement Learning

In enormous environments, use abstraction:

- State Abstraction
- Temporal Abstraction

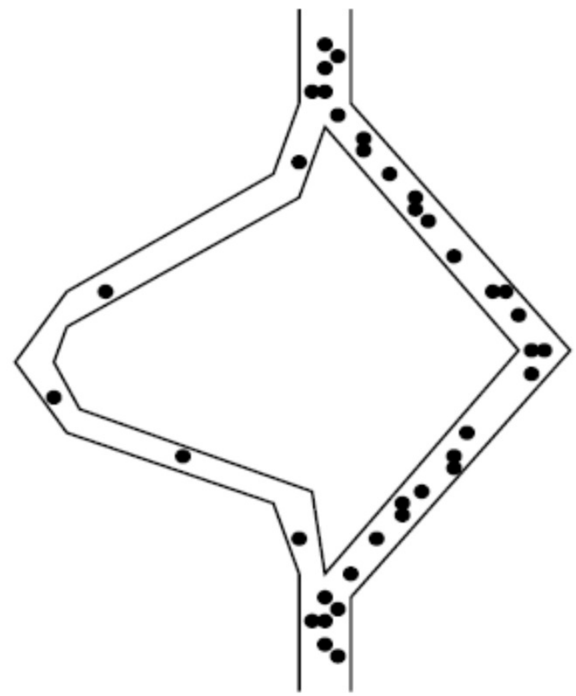
Hierarchical Reinforcement Learning

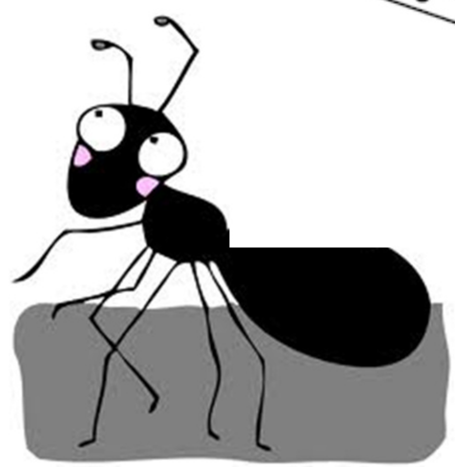
Option Framework:

- Formal description of macro actions.
- Option is a sorted tuple: $\mathrm{o} = (I, \pi, \beta)$
  - I: Set of states that o is permitted on.
  - π: Agent's policy, $\pi: S \times A \rightarrow [0, 1]$
  - $\beta(s)$: Function to indicate episode finishing.
- Primitive actions as options
  - I: the state action is permitted on.
  - $\pi(s, a) = 1$
  - $\beta(s) = 1$

Ant Colony Optimization

- Ant System
  - Find Shortest path from s to t
  - n_t episodes
  - n_k ants
  - Stochastic path creation based on pheromone values.





Ant Colony Optimization – Ant System

– Path generation

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}(t)}{\sum_{j\in N_i^k(t)} \alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}(t)} & if\ j \in N_i^k(t) \\ 0 & if\ j \notin N_i^k(t) \end{cases}$$

– Evaporation:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t)$$

- Pheromone deposit:

$$\Delta\tau_{ij}^{k}(t) \propto \frac{1}{L^k(t)}$$

– Use taboo list.

Proposed Method – Bird's Eye View

1- Transition Graph Generation

- Interaction in env
- Interaction history stored in weighted directed graph

2- Sub-goal Identification

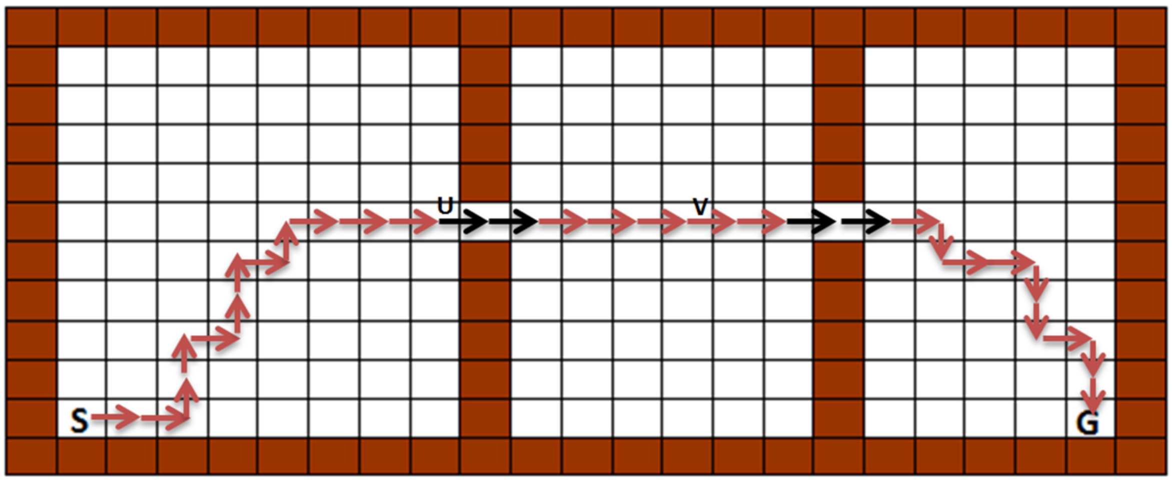
- Execution of ant system
- Roughness consideration of edges on shortest path
- Selection of edges with lower amounts of Roughness
- Deletion of consecutive bottleneck edges

3- Skill Generation

- Shortest path segmentation
- Identification of communities on the shortest path
- Acquiring of optimal policies in each community.

Proposed Method- Sub-goal Identification

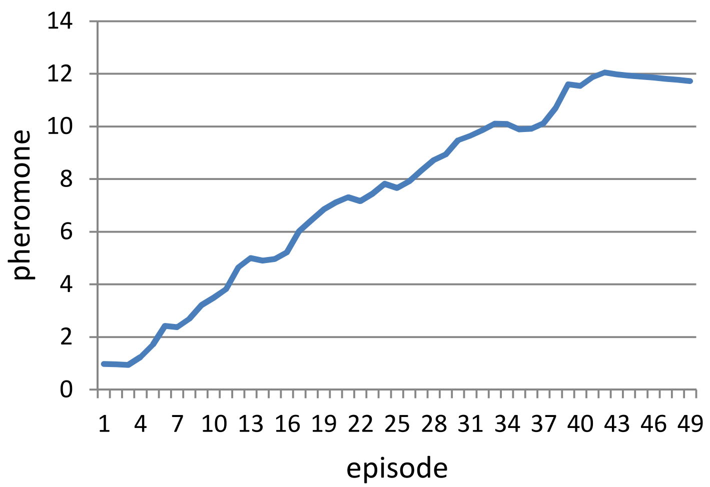
Execution of ant system on transition graph



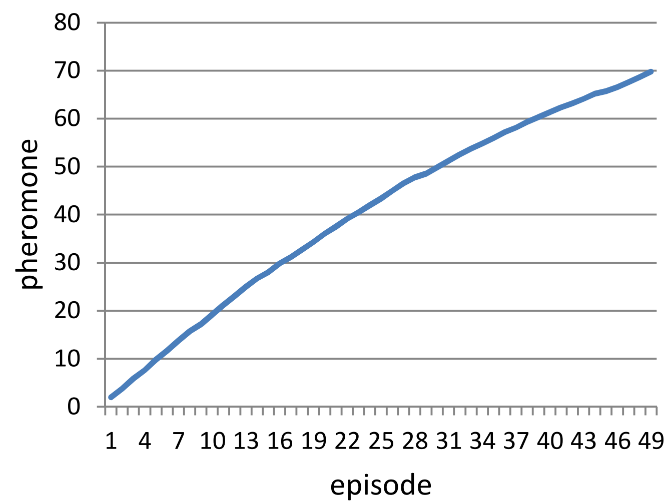
- Regular participation of u in generated shortest paths
- Irregular participation of v in generated shortest paths

Proposed Method- Sub-goal Identification

- Comparison of pheromone values of u and v during ant system



Variation of pheromone values of v in time



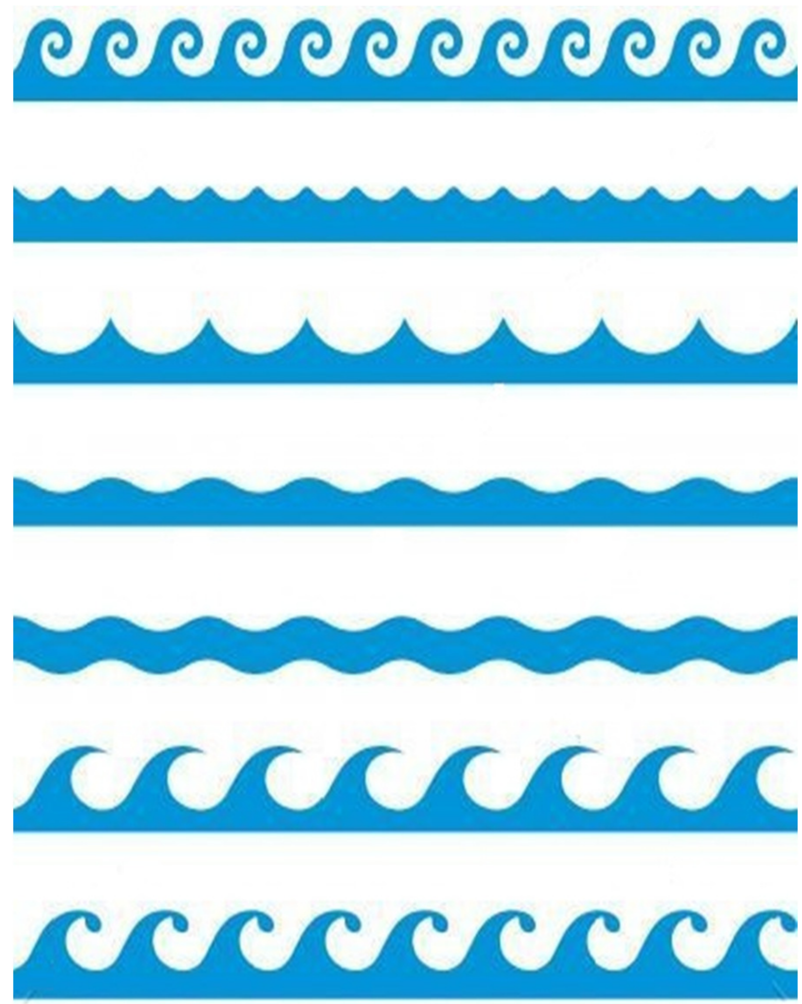
Variation of pheromone values of u in time

Proposed Method- Sub-goal Identification

Proposed Criteria for separation of these edges, called **Roughness:**

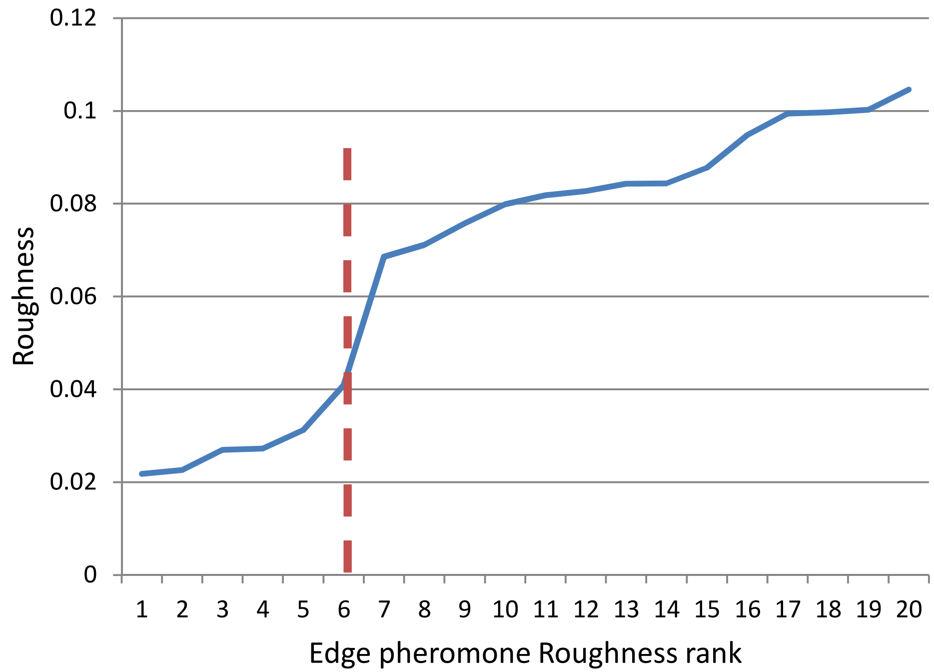
$$- R_F = \frac{\sigma_M^2}{(\max\limits_i F_i - \min\limits_i F_i)^2}$$

Where $M_i = F_{i+1} - F_i$



Proposed Method- Sub-goal Identification

- Sorting shortest path edges based on pheromone values



Edge roughness values based on edge pheromone Roughness rank

Proposed Method- Sub-goal Identification

- Separation of bottleneck edges:
  - Using threshold values:
    - For pheromone values: τ_v
    - For pheromone increase slope: τ_d

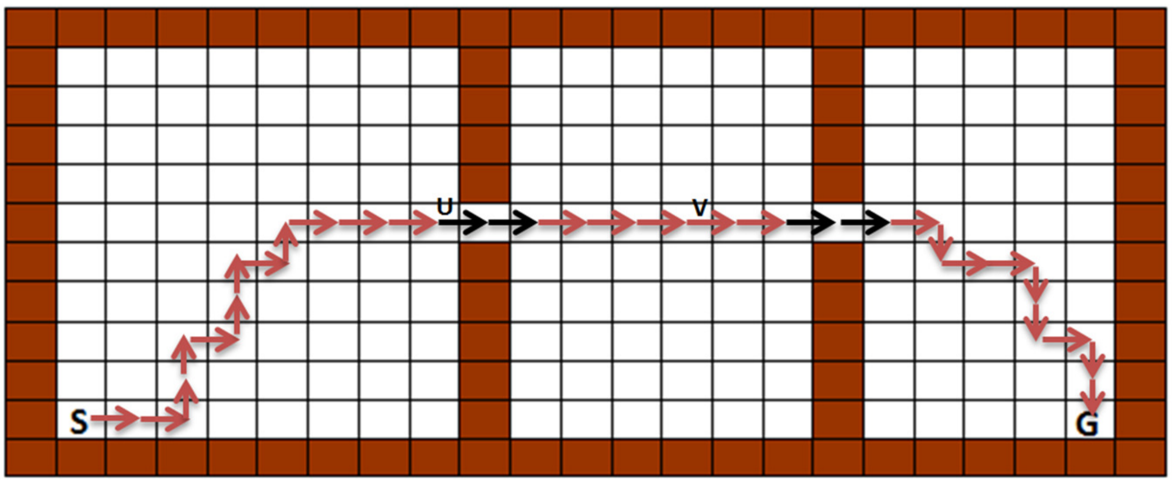
b is the rank border between bottleneck and non-bottleneck edges iff:

$$Fail(b) = true\ and\ \forall i < b : Fail(i) = false$$

$$Fail(i) = (d_i > \tau_d \, . \, d_{init}\ or\ v_i > \tau_v . v_0)$$

Proposed Method- Sub-goal Identification

- Removing consecutive bottleneck edges.
- Getting vertices on bottleneck edges as final sub-goals.



Proposed Method- Sub-goal Identification

- Sub-goal discovery algorithm

Algorithm 1 The proposed method for sub-goal detection

Input: $(n_k, t_d, \alpha, \rho, \tau_d, \tau_v)$
Output: SubGoals: a list of sub-goals
Run Ant System (τ_d, α, ρ) and have SP with shortest path.
Sort SP increasingly according to field R_P.
$v_0 \leftarrow SP[0].R_P$
for $i \leftarrow 1$ to length(SP) **do**
    $d_{init} \leftarrow SP[i].R_P - SP[i-1].R_P$
    if $d_{init} \neq 0$ **then**
        exit the loop.
    end if
end for
for $b \leftarrow 1$ to length(SP) **do**
    if $(SP[b].R_P - SP[b-1].R_P > \tau_d.d_{init}) \vee (SP[b].R_P > v_0.\tau_v)$ **then**
        exit the loop.
    end if
end for
for Adjacent :adjacent set of edges in $SP[0 \ldots b-1]$ **do**
    $best \leftarrow argmin_i\{Adjacent.Edges[i].R_P\}$
    SubGoals.add(Adjacent.Edges[best].head)
end for

Proposed Method- Sub-goal Identification

- Incremental variance calculation:

$$\sigma_n^2 = \frac{\sum_{i=1}^{n}(x_i - \overline{X}_n)^2}{n} = \frac{y_n}{n}$$

$$y_n = \sum_{i=1}^{n}(x_i - \overline{X}_n)^2 = \sum_{i=1}^{n} {x_i}^2 + n{\overline{X}_n}^2 - 2\overline{X}_n \sum_{i=1}^{n} x_i$$

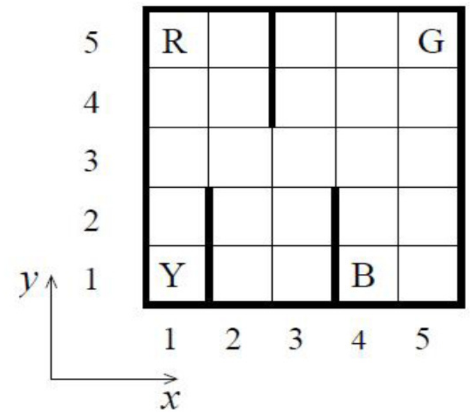
$$y_n = \sum_{i=1}^{n} {x_i}^2 + n\left(\frac{s_n}{n}\right)^2 - 2\frac{s_n}{n}s_n = \sum_{i=1}^{n} {x_i}^2 - \frac{{s_n}^2}{n}$$

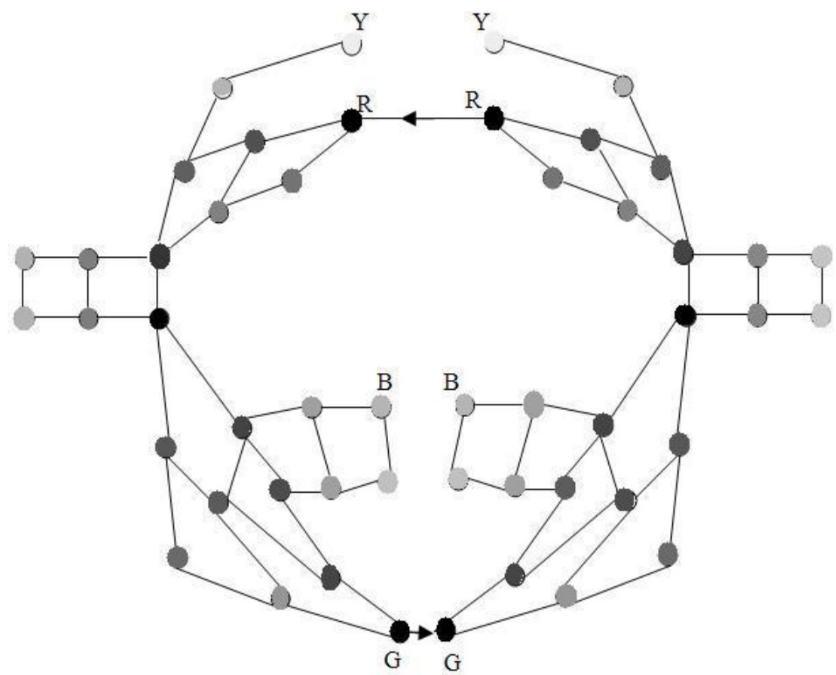
$$y_{n+1} = \sum_{i=1}^{n+1} {x_i}^2 - \frac{s_{n+1}^2}{n+1}$$

$$y_{n+1} = y_n + x_{n+1}^2 - \left(\frac{s_{n+1}^2}{n+1} - \frac{{s_n}^2}{n}\right)$$

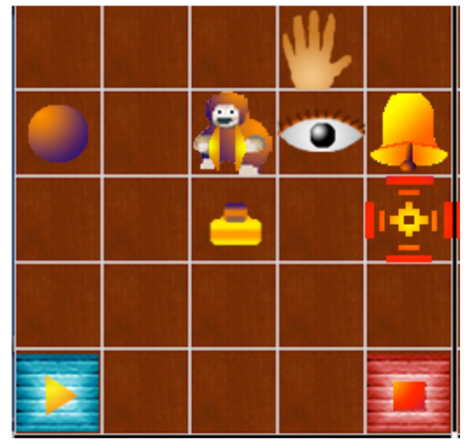
Environments

- Taxi Environment
  - Goal: take person to destination
  - Actions:
    - Movement in 4 directions
    - Take passenger in taxi
    - Take passenger out of taxi
  - Reward
    - +10: taking the passenger in taxi
    - +20: take passenger out in destination
    - -1: every other action

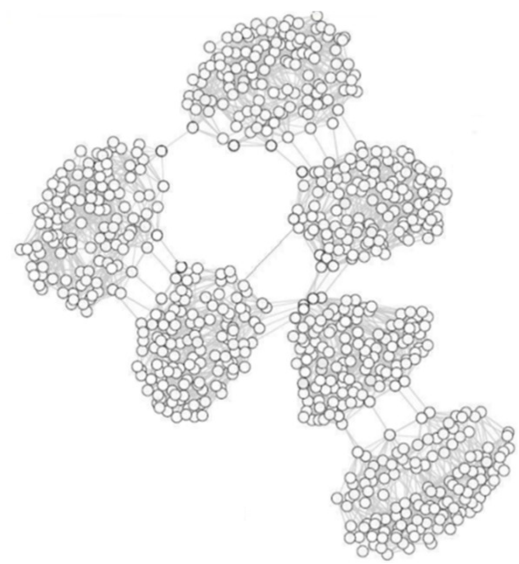




Environments

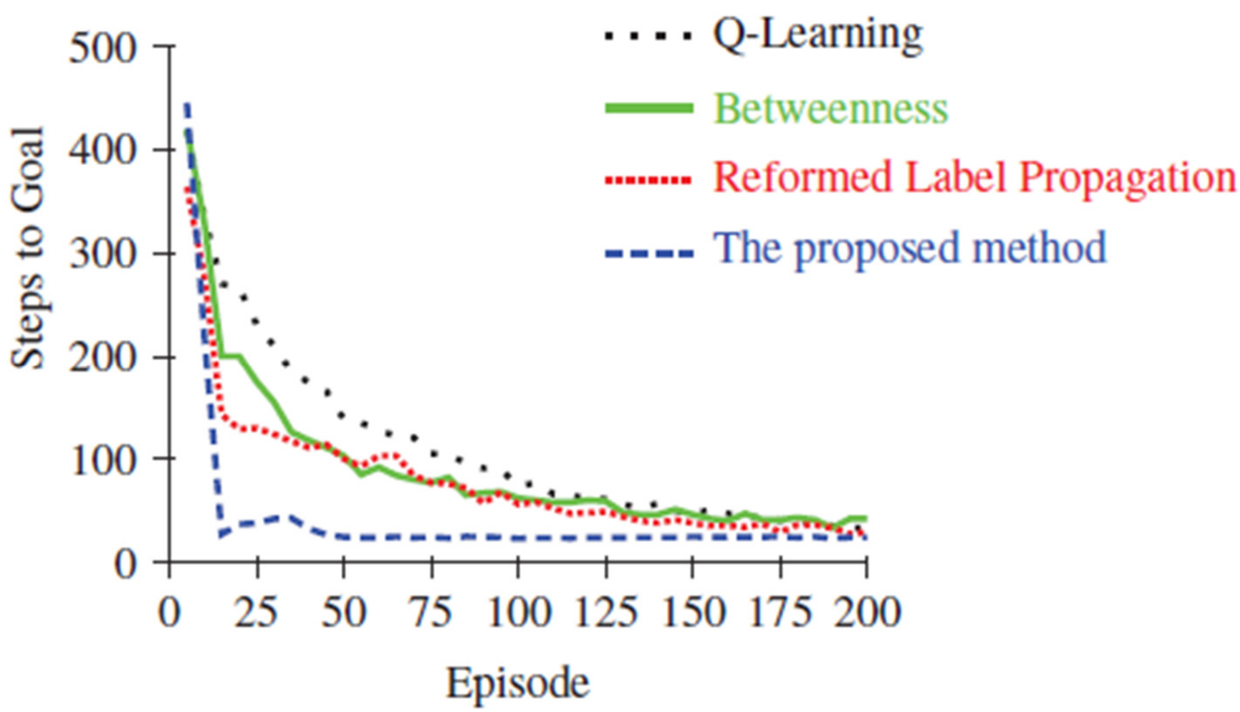


- Playroom environment
  - Goal: making monkey scream
  - Actions:
    - 1) look at a random object
    - 2) look at object at hand
    - 3) hold object it is looking at
    - 4) look at object marker is placed on
    - 5) place marker on object it is looking at
    - 6) move object in hand to location it is looking at
    - 7) turn over light switch
    - 8) press music button
    - 9) hit ball toward the marker.
  - Rewards:
    - +1000: reaching the goal
    - -1: every other action



Experimental Results

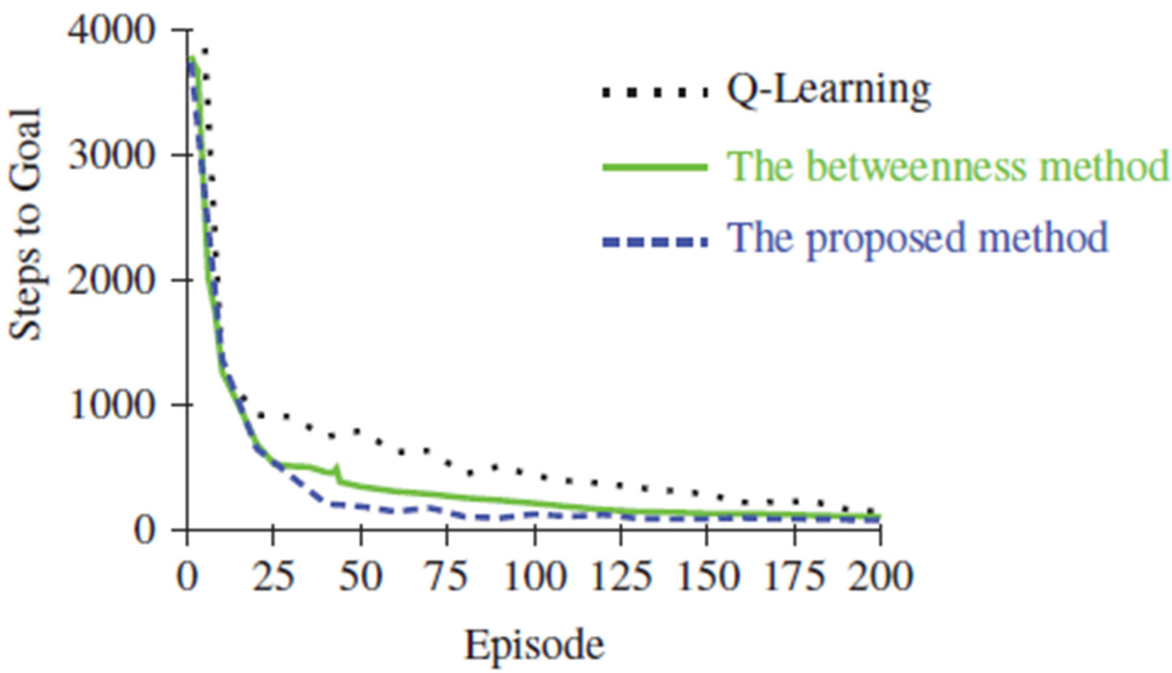
- Taxi environment



$n_t = 10, n_k = 25, \alpha = 0.9, \rho = 0.98, \tau_v = 1.01, \tau_d = 1.5$

Experimental Results

- Playroom Environment



$n_t = 200, n_k = 10, \alpha = 0.9, \rho = 0.98, \tau_v = 2.0, \tau_d = 1.5$

References

[1] Fortunato, S. Community detection in graphs. Physics Reports, 486(3-5):75–174, 2010.

[2] Simsek, O. and Barto, A.G. Identifying useful subgoals in reinforcement learning by local graph partitioning. In Proceedings of the 22nd international conference on machine learning, pp. 816–823. ACM, 2005.

[3] Menache, I., Mannor, S., and Shimkin, N. Q-cut dynamic discovery of sub-goals in reinforcement learning. Machine Learning: ECML 2002, pp. 187– 195. 2002.

[4] Simsek, O. and G., Barto A. Skill characterization based on betweenness. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., Eds., Advances in Neural Information Processing Systems 21, pp. 1497-1504, 2009.

[5] Kazemitabar, S. J. and Beigy, H., Automatic Discovery of Subgoals in Reinforcement Learning Using Strongly Connected Components. ICONIP, pp. 829-834, 2008.

References

[6] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc.Natl. Acad. Sci. USA 99, 2002.

[7] M. E. J. Newman, Modularity and community structure in networks, Proc. Natl. Acad. Sci.USA 103, 8577–8582, 2006.

[8] M. E. J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) 2004.

[9] M. E. J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6), 2004.

[10] Kazemitabar, S. J. and Beigy, H., Using Strongly Connected Components as a Basis for Autonomous Skill Acquisition in Reinforcement Learning. ISNN (1), pp. 794-803, 2009.

[11] Andries P. Engelbrecht. Computational intelligence: an introduction, 2nd edition. John Wiley & Son, 2007. ISBN 978-0-470-03561-0.