

```
% EX_9_4_DYNAQPLUS_SCRIPT - Implements the DynaQ Algorithm on the simple maze example
found in Chapter 9
%
% Written by:
% --
% John L. Weatherwax                2007-12-03
%
% email: wax@alum.mit.edu
%
% Please send comments and especially bug reports to the
% above email address.
%
%-----

close all;
clc;

% sample_discrete.m
addpath( genpath( '../.../FullBNT-1.0.4/KPMstats/' ) );

% the learning rate:
%alpha = 1e-1;
alpha = 2e-1;

% the probability of a random action (non-greedy):
epsilon = 0.1;

% the discount factor:
gamma = 0.95;
%gamma = 1.0;

% get our initial mazes:
%mz_fn = @ mk_ex_9_1_mz;

% (the blocking maze):
% changes at 1000 timesteps ... seems to be solved when we have a large enough number of
timesteps
%mz_fn = @ mk_ex_9_2_mz;

% (the opening maze):
% changes at 3000 timesteps
mz_fn = @ mk_ex_9_3_mz;

MZ = mz_fn(0); [sideII,sideJJ] = size(MZ);

% the beginning and terminal states (in matrix notation):
%s_start = [ 3, 1 ];
%s_end    = [ 1, 9 ];
s_start = [ 6, 4 ];
s_end    = [ 1, 9 ];

MAX_N_STEPS=30;
MAX_N_STEPS=3e3;
MAX_N_STEPS=5e3;
%MAX_N_STEPS=3e4;

% the number of steps to do in planning:
%nPlanningSteps = 0;
nPlanningSteps = 5;
```

```

%nPlanningSteps = 50;

% a factor relating how important revisiting old states is, relative to
% the past recieved reward coming from these states/action pairs ...
%kappa = 0.02;
kappa = 2/sqrt(MAX_N_STEPS);

%--
% USE modified (example 9.4) dynaQplus:
%--

[Q,ets,cr] =
ex_9_4_dynaQplus(alpha,epsilon,gamma,kappa,nPlanningSteps,mz_fn,s_start,s_end,MAX_N_STEPS
);

% compute the (negative) cost to go and the optimal (greedy with respect to the state-
value function) policy:
pol_pi = zeros(sideII,sideJJ); V = zeros(sideII,sideJJ);
for ii=1:sideII,
    for jj=1:sideJJ,
        sti = sub2ind( [sideII,sideJJ], ii, jj );
        [V(ii,jj),pol_pi(ii,jj)] = max( Q(sti,:) );
    end
end

plot_mz_policy(pol_pi,mz_fn(MAX_N_STEPS),s_start,s_end);
title( 'policy (1=>up,2=>down,3=>right,4=>left); start=green; stop=red' );
fn = sprintf('ex_9_4_dqp_policy_nE_%d',MAX_N_STEPS); saveas( gcf, fn, 'png' );

if( 0 )
    figure; imagesc( V ); colormap(flipud(jet)); colorbar;
    title( 'state value function' );
    %fn = sprintf('ex_9_4_dqp_state_value_fn_nE_%d',MAX_N_STEPS); saveas( gcf, fn, 'png' );

    figure; plot( 1:length(ets), ets, '-x' );
    title( 'number of timesteps to reach goal' ); grid on;
    xlabel('episode number'); ylabel('number of timesteps required'); drawnow;
    %fn = sprintf('ex_9_4_dqp_q_learning_rate_nPS_%d',nPlanningSteps); saveas( gcf, fn,
'png' );

    figure; plot( (1:MAX_N_STEPS), cr(2:end), '-x' );
    title( 'cummulative reward' ); grid on; %axis( [ 0 3000, 0, cr(3001)] );
    xlabel('timestep index'); ylabel('cum. reward'); drawnow;
    %fn = sprintf('ex_9_4_dqp_q_cum_reward_nPS_%d',nPlanningSteps); saveas( gcf, fn, 'png'
);
end

%--
% USE dynaQplus:
%--

[Q,ets,cr] =
dynaQplus_maze(alpha,epsilon,gamma,kappa,nPlanningSteps,mz_fn,s_start,s_end,MAX_N_STEPS);

% compute the (negative) cost to go and the optimal (greedy with respect to the state-
value function) policy:
pol_pi = zeros(sideII,sideJJ); V = zeros(sideII,sideJJ);
for ii=1:sideII,
    for jj=1:sideJJ,
        sti = sub2ind( [sideII,sideJJ], ii, jj );

```

```
[V(ii,jj),pol_pi(ii,jj)] = max( Q(sti,:) );
end
end

plot_mz_policy(pol_pi,mz_fn(MAX_N_STEPS),s_start,s_end);
title( 'policy (1=>up,2=>down,3=>right,4=>left); start=green; stop=red' );
fn = sprintf('ex_9_4_qp_policy_compare_nE_%d',MAX_N_STEPS); saveas( gcf, fn, 'png' );

if( 0 )
    figure; imagesc( V ); colormap(flipud(jet)); colorbar;
    title( 'state value function' );
    %fn = sprintf('XXX_state_value_fn_nE_%d',MAX_N_STEPS); saveas( gcf, fn, 'png' );

    figure; plot( 1:length(ets), ets, '-x' );
    title( 'number of timesteps to reach goal' ); grid on;
    xlabel('episode number'); ylabel('number of timesteps required'); drawnow;
    %fn = sprintf('XXX_q_learning_rate_nPS_%d',nPlanningSteps); saveas( gcf, fn, 'png' );

    figure; plot( (1:MAX_N_STEPS), cr(2:end), '-x' );
    title( 'cumulative reward' ); grid on; %axis( [ 0 3000, 0, cr(3001)] );
    xlabel('timestep index'); ylabel('cum. reward'); drawnow;
    %fn = sprintf('XXX_q_cum_reward_nPS_%d',nPlanningSteps); saveas( gcf, fn, 'png' );
end

clear functions;
%close all;
return;
```