singletest.m (AN algorithm, objectworld)


```
% Convenience script for running a single test.
addpaths;
test_result = runtest('an',struct(),'linearmdp',...
    'objectworld',struct('n',32,'determinism',0.7,'seed',1,'continuous',0),...
    struct('training_sample_lengths',8,'training_samples',16,'verbosity',2));

% Visualize solution.
% printresult(test_result);
visualize(test_result);
```

runtest(algorithm, algorithm_params, mdp_model, mdp, mdp_params, test_params)

algorithm = 'an' AN algorithm을 사용

algorithm_params = struct() algorithm_params는 빈 struct.

mdp_model = 'linearmdp' mdp_model은 linearmdp를 사용

mdp = 'objectworld' 실행하는 mdp 종류는 'objectworld'

mdp_params = struct ('n', 32, 'determinism', 0.7, 'seed', 1, 'continuous', 0)

mdp를 build하기 위해 필요한 mdp의 parameters.

test_params = struct ('training_sample_lengths', 8, 'training_samples', 16, 'verbosity', 2)

test를 위한 parameters.

runtest.m


```
% Run IRL test with specified algorithm and example.
function test_result = runtest(algorithm,algorithm_params,...
    mdp_model,mdp,mdp_params,test_params)

% test_result - structure that contains results of the test:
%   see evaluateirl.m
% algorithm - string specifying the IRL algorithm to use; one of:
%   firl - NIPS 2010 FIRL algorithm
%   bfirl - Bayesian FIRL algorithm
% algorithm_params - parameters of the specified algorithm:
%   FIRL:
%       seed (0) - initialization for random seed
%       iterations (10) - number of FIRL iterations to take
%       depth_step (1) - increase in depth per iteration
%       init_depth (0) - initial depth
%   BFIRL:
%       seed (0) - initialization for random seed
% mdp_model - string specifying MDP model to use for examples:
%   standardmdp - standard MDP model
% mdp - string specifying example to test on:
%   gridworld
% mdp_params - string specifying parameters for example:
%   Gridworld:
%       seed (0) - initialization for random seed
%       n (32) - number of cells along each axis
%       b (4) - size of macro cells
%       determinism (1.0) - probability of correct transition
%       discount (0.9) - temporal discount factor to use
% test_params - general parameters for the test:
%   test_models - models to test on
%   test_metrics - metrics to use during testing
%   training_samples (32) - number of example trajectories to query
%   training_sample_lengths (100) - length of each sample trajectory
%   true_features ([]) - alternative set of true features

% Make sure relevant paths are added.
addpaths;

% Set default test parameters.
test_params = setdefaulttestparams(test_params);

% Construct MDP and features.
[mdp_data,r,feature_data,true_feature_map] = feval(strcat(mdp,'build'),mdp_params);
if ~isempty(test_params.true_features),
    true_feature_map = test_params.true_features;
end;

% Solve example.
mdp_solution = feval(strcat(mdp_model,'solve'),mdp_data,r);
```

- test 하고자하는 algorithm 이름
  ex) am, gpirl ---
- 각 algorithm에 필요한 parameter
  struct 형태로 입력,
  struct()로 입력할 경우 각 algorithm에 default parameter 사용.
  ex) AN (아래 andefaultparams.m 참조)
  struct는 'seed', 'all_features', 'true_features'로 구성.
  default value는 각각 0, 1.0
- mdp_model
  적용하는 mdp의 형태
  standardmdp 또는 linearmdp 두 가지를 지원하는 듯함.
- mdp
  적용하는 mdp 예제
  ex) gridword, Highway
- mdp_params
  적용하는 mdp 예제를 결정하기 위한 parameter들.
- test_params
  IRL 함수를 위해 주어지는 조건

mdpbuild를 실행해 mdp와 feature 생성

feval(fun, variable) : fun 이름을 가지는 함수를 variable을 입력으로 실행시킬.

mdpbuild 함수를 mdp_params 에 대해 실행.

String concatenate

ex) mdp = gridword 이면

strcat(mdp, 'build') = gridwordbuild.

test_params.true_features 값이 있으면 이를 true_feature_map 으로 사용

주어진 mdp_data와 r을 이용해 mdp를 풀라.

```
% Sample example trajectories.
if isempty(test_params.true_examples),
    example_samples = sampleexamples(mdp_model,mdp_data,mdp_solution,test_params);
else
    example_samples = test_params.true_examples;
end;

% Run IRL algorithm.
irl_result = feval(strcat(algorithm,'run'),algorithm_params,mdp_data,mdp_model,...
    feature_data,example_samples,true_feature_map,test_params.verbosity);

% Evaluate result.
test_result = evaluateirl(irl_result,r,example_samples,mdp_data,mdp_params,...
    mdp_solution,mdp,mdp_model,test_params.test_models,...
    test_params.test_metrics,feature_data,true_feature_map);
test_result.algorithm = algorithm;
```

test_params.true_examples 가 공란일 때

sampleexamples 를 이용해 주어진 변수들에 대한 example_sample 생성.

있을 때 바로 example_samples 로 사용.

evaluateirl 함수를 이용해 irl_result 들을 평가.

setdefaulttestparams.m


```
% Set default general parameters for the test.
function test_params = setdefaulttestparams(test_params)

% Create default parameters.
default_params = struct(...
    'verbosity',2,...
    'training_samples',32,...
    'training_sample_lengths',100,...
    'true_features',[],...
    'true_examples',[],...
    'test_models',{{'standardmdp','linearmdp'}},...
    'test_metrics',{{'misprediction','policydist','featexp','value',...
        'featvar','reward','rewarddemean','rewardmomentmatch'}});

% Set parameters.
test_params = filldefaultparams(test_params,default_params);
```

test_params을 받아 default_params의 field 중 test_params에 없는 field에 default_params의 해당 field의 field 값을 채워줌

즉, test_params로 입력한 field 값은 그대로 사용하고 없는 field 값은 default 값을 채워넣어 모든 field의 값을 가진 structure로 만들어줌.

filedefaultparams.m


```
% Fill in default parameters of a structure.
function params = filldefaultparams(params,defaults)

% Get default field names.
defaultfields = fieldnames(defaults);

% Step over all fields in the defaults structure.
for i=1:length(defaultfields),
    if ~isfield(params,defaultfields{i}),
        params.(defaultfields{i}) = defaults.(defaultfields{i});
    end;
end;
```

입력된 structure의 fieldname들을 반환해준다.

default parameter structure의 fieldname을 받아온다.

params에 defaultfield(i)가 있는지 없는지 체크한다.

없으면 params에 defaultfield(i) field를 만들고

defaults.(defaultfields(i)) 값을 저장한다.

objectworldbuild.m


```
% Construct the Objectworld MDP structures.
function [mdp_data,r,feature_data,true_feature_map] = objectworldbuild(mdp_params)

% mdp_params - parameters of the objectworld:
%       seed (0) - initialization for random seed
%       n (32) - number of cells along each axis
%       placement_prob (0.05) - probability of placing object in each cell
%       c1 (2) - number of primary "colors"
%       c2 (2) - number of secondary "colors"
%       determinism (1.0) - probability of correct transition
%       discount (0.9) - temporal discount factor to use
% mdp_data - standard MDP definition structure with object-world details:
%       states - total number of states in the MDP
%       actions - total number of actions in the MDP
%       discount - temporal discount factor to use
%       sa_s - mapping from state-action pairs to states
%       sa_p - mapping from state-action pairs to transition probabilities
%       map1 - mapping from states to c1 colors
%       map2 - mapping from states to c2 colors
%       c1array - array of locations by c1 colors
%       c2array - array of locations by c2 colors
% r - mapping from state-action pairs to rewards

% Fill in default parameters.
mdp_params = objectworlddefaultparams(mdp_params);

% Set random seed.
rand('seed',mdp_params.seed);

% Build action mapping.
sa_s = zeros(mdp_params.n^2,5,5);
sa_p = zeros(mdp_params.n^2,5,5);
for y=1:mdp_params.n,
    for x=1:mdp_params.n,
        s = (y-1)*mdp_params.n+x;
        successors = zeros(1,1,5);
        successors(1,1,1) = s;
        successors(1,1,2) = (min(mdp_params.n,y+1)-1)*mdp_params.n+x;
        successors(1,1,3) = (y-1)*mdp_params.n+min(mdp_params.n,x+1);
        successors(1,1,4) = (max(1,y-1)-1)*mdp_params.n+x;
        successors(1,1,5) = (y-1)*mdp_params.n+max(1,x-1);
        sa_s(s,:,:) = repmat(successors,[1,5,1]);
        sa_p(s,:,:) = reshape(eye(5,5)*mdp_params.determinism +...
            (ones(5,5)-eye(5,5))*((1.0-mdp_params.determinism)/4.0),...
            1,5,5);
    end;
end;

% Construct map.
map1 = zeros(mdp_params.n^2,1);
```

objectworld를 build하는데 필요한 parameters

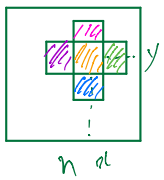
objectworld build를 통해 만들어질 MDP 문제의 Data

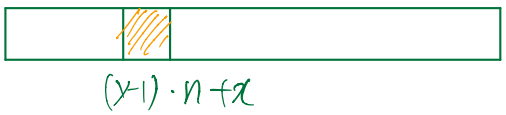
— MDP problem의 reward.

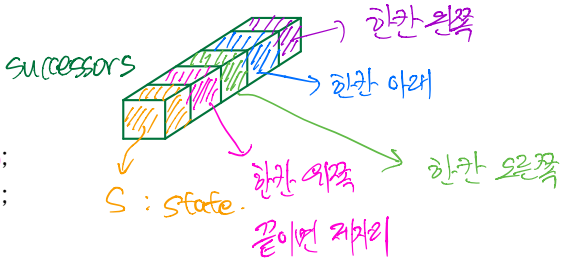
objectworlddefaultparams 함수를 이용해 정의되지 않은 mdp_params 값을 default 값으로 설정.

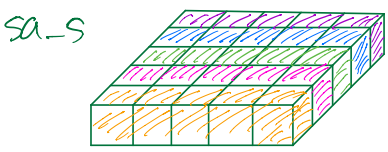
random number generator seed 설정.

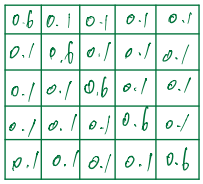
action, state, probability

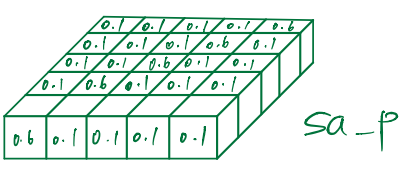


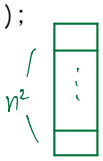












```
map2 = zeros(mdp_params.n^2,1);
c1array = cell(mdp_params.c1,1);
c2array = cell(mdp_params.c2,1);
% Place objects in "rounds", with 2 colors each round.
% This ensures, for example, that increasing c1 from 2 to 4 results in all
% of the objects from c1=2 being placed, plus additional "distractor"
% objects. This prevents the situation when c1 is high of not placing any
% objects with c1=1 or c1=2 (which makes the example useless for trying to
% infer any meaningful reward).
for round=1:ceil(mdp_params.c1*0.5),
    initc1 = (round-1)*2;
    if initc1+1 == mdp_params.c1,
        % Always choose the leftover c1.
        prob = mdp_params.placement_prob*0.5;
        maxc1 = 1;
    else
        % Choose from two c1 colors.
        prob = mdp_params.placement_prob;
        maxc1 = 2;
    end;
    for s=1:mdp_params.n^2,
        if rand(1,1) < prob && map1(s) == 0,
            % Place object.
            c1 = initc1+ceil(rand(1,1)*maxc1);
            c2 = ceil(rand(1,1)*mdp_params.c2);
            map1(s) = c1;
            map2(s) = c2;
            c1array{c1} = [c1array{c1};s];
            c2array{c2} = [c2array{c2};s];
        end;
    end;
end;

% Create MDP data structure.
mdp_data = struct(...
    'states',mdp_params.n^2,...
    'actions',5,...
    'discount',mdp_params.discount,...
    'determinism',mdp_params.determinism,...
    'sa_s',sa_s,...
    'sa_p',sa_p,...
    'map1',map1,...
    'map2',map2,...
    'c1array',{c1array},...
    'c2array',{c2array});

% Construct feature map.
[feature_data,true_feature_map,r] = objectworldfeatures(mdp_params,mdp_data);
```

c1이 위치할 장소를 나타내기위한 cell 생성

c1=2
round = 1
initc1 = 0
prob
maxc1=2
5=k.
c1=1또는2.
c2=1또는2.
map1(k) = c1
map2(k) = c2

c1=3
round= 1,2
initc1= 0, 2.
prob maxc1=2
0.5prob maxc1=1
5=k, initc1=0
c1=1또는2.
c2=1또는2.
map1(k) = c1
map2(k) = c2
5=k. initc1=2
c1= 3
c2=1또는2.
map1(k) = c1
map2(k) = c2

c1=4
round= 1.2
initc1=0.2
prob maxc1=2
prob maxc1=2
5=k, initc1=0
c1=1또는2.
c2=1또는2.
map1(k) = c1
map2(k) = c2
5=k. initc1=2
c1= 3또는4
c2=1또는2
map1(k) = c1
map2(k) = c2

비교할 상황

이걸 상황.

maxc1=1 이면 1
maxc1=2 이면 1 또는 2
mdp_params.c2=2
c2=1또는2.

c1array의 c1 cell에 아래와 행에 s를 추가.

주어진 mdp_params와 이를 통해 생성한 mdp structure인 mdp_data를 바탕으로 feature map 생성.

Objectworld default params.m


```
% Fill in default parameters for the objectworld example.
function mdp_params = objectworlddefaultparams(mdp_params)

% Create default parameters.
default_params = struct(...
    'seed',0,...
    'n',32,...
    'placement_prob',0.05,...
    'c1',2,...
    'c2',2,...
    'continuous',0,...
    'determinism',1.0,...
    'discount',0.9);

% Set parameters.
mdp_params = filldefaultparams(mdp_params,default_params);

% Construct default reward tree.
c1 = mdp_params.c1;
c2 = mdp_params.c2;
step = c1+c2;
%{
r_tree = struct('type',1,'test',1+step*2,'total_leaves',3,...        % Test distance to c1 1 shape
    'gtTree',struct('type',0,'index',1,'mean',[-2,-2,-2,-2,-2]),... % Penalty for beeing close to c1
shape
    'ltTree',struct('type',1,'test',2+step*1,'total_leaves',2,...    % Test distance to c1 2 shape
        'gtTree',struct('type',0,'index',2,'mean',[1 1 1 1 1]),...   % Reward for being close
        'ltTree',struct('type',0,'index',3,'mean',[0 0 0 0 0])));    % Neutral reward for any other
state.
%}
r_tree = struct('type',1,'test',1+step*2,'total_leaves',3,...        % Test distance to c1 1 shape
    'ltTree',struct('type',0,'index',1,'mean',[0,0,0,0,0]),... % Neutral reward for being elsewhere
    'gtTree',struct('type',1,'test',2+step*1,'total_leaves',2,...    % Test distance to c1 2 shape
        'gtTree',struct('type',0,'index',2,'mean',[1 1 1 1 1]),...   % Reward for being close
        'ltTree',struct('type',0,'index',3,'mean',[-2 -2 -2 -2 -2])));   % Penalty otherwise

% Create default parameters.
default_params = struct(...
    'r_tree',r_tree);

% Set parameters.
mdp_params = filldefaultparams(mdp_params,default_params);
```

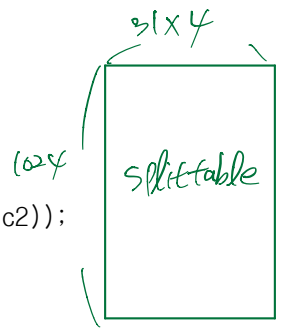
r_tree
- type 1
- test 1+step*2
- total_leaves 3
- ltTree
  - type 0
  - index 1
  - mean [0,0,0,0,0]
- gtTree
  - type 1
  - test 2+step
  - total_leaves 2
  - gtTree
    - type 0
    - index 2
    - mean [1,1,1,1,1]
  - ltTree
    - type 0
    - index 3
    - mean [-2,-2,-2,-2,-2]

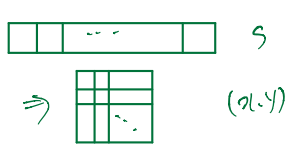
7.1 Objectworld Experiments

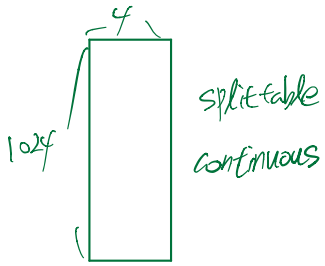
The objectworld is an $N \times N$ grid of states with five actions per state, corresponding to steps in each direction and staying in place. Each action has a 30% chance of moving in a different random direction. Randomly placed objects populate the objectworld, and each is assigned one of C inner and outer colors. Object placement is randomized in the transfer environments, while N and C remain the same. There are $2C$ continuous features, each giving the Euclidean distance to the nearest object with a specific inner or outer color. In the discrete feature case, there are $2CN$ binary features, each one an indicator for a corresponding continuous feature being less than $d \in \{1, ..., N\}$. The true reward is positive in states that are both within 3 cells of outer color 1 and 2 cells of outer color 2, negative within 3 cells of outer color 1, and zero otherwise. Inner colors and all other outer colors are distractors. The algorithms were provided example paths of length 8, and the number of examples and colors was varied to determine their ability to handle limited data and distractors.

objectworldfeatures.m


```
1 % Construct the raw features for the objectworld domain.
2 function [feature_data,true_feature_map,r] = objectworldfeatures(mdp_params,mdp_data)
3
4 % mdp_params - definition of MDP domain
5 % mdp_data - generic definition of domain
6 % feature_data - generic feature data:
7 %   splittable - matrix of states to features
8 %   stateadjacency - sparse state adjacency matrix
9
10 % Fill in default parameters.
11 mdp_params = objectworlddefaultparams(mdp_params);
12
13 % Construct adjacency table.
14 stateadjacency = sparse([],[],[],mdp_data.states,mdp_data.states,...
15     mdp_data.states*mdp_data.actions);
16 for s=1:mdp_data.states,
17     for a=1:mdp_data.actions,
18         stateadjacency(s,mdp_data.sa_s(s,a,1)) = 1;
19     end;
20 end;
21
22 % Construct discrete and continuous split tables.
23 splittable = zeros(mdp_data.states,(mdp_params.n-1)*(mdp_params.c1+mdp_params.c2));
24 splittablecont = zeros(mdp_data.states,mdp_params.c1+mdp_params.c2);
25 for s=1:mdp_data.states,
26     % Get x and y positions.
27     y = ceil(s/mdp_params.n);
28     x = s-(y-1)*mdp_params.n;
29
30     % Determine distances to each type of object.
31     c1dsq = sqrt(2*(mdp_params.n^2))*ones(mdp_params.c1,1);
32     c2dsq = sqrt(2*(mdp_params.n^2))*ones(mdp_params.c2,1);
33     for i=1:mdp_params.c1,
34         for j=1:length(mdp_data.c1array{i}),
35             cy = ceil(mdp_data.c1array{i}(j)/mdp_params.n);
36             cx = mdp_data.c1array{i}(j)-(cy-1)*mdp_params.n;
37             d = sqrt((cx-x)^2 + (cy-y)^2);
38             c1dsq(i) = min(c1dsq(i),d);
39         end;
40     end;
41     for i=1:mdp_params.c2,
42         for j=1:length(mdp_data.c2array{i}),
43             cy = ceil(mdp_data.c2array{i}(j)/mdp_params.n);
44             cx = mdp_data.c2array{i}(j)-(cy-1)*mdp_params.n;
45             d = sqrt((cx-x)^2 + (cy-y)^2);
46             c2dsq(i) = min(c2dsq(i),d);
47         end;
48     end;
49
50     % Build corresponding feature table (discrete).
```







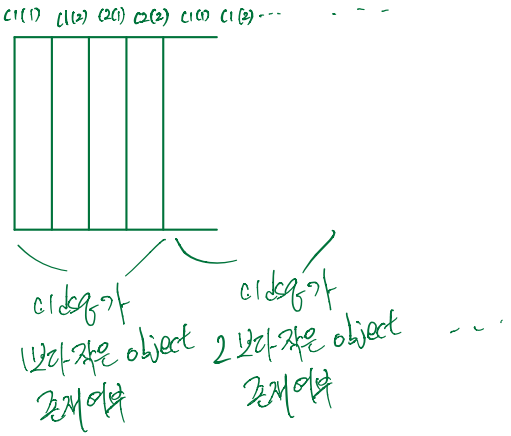
c1 중 i번째 색깔을 갖는 object의 위치 vector.

state s와 (i,j) object 사이의 거리

state 모에 대해, 바깔쪽의 색깔이 c1(i)인 object 들 중 가장 가까운 object와의 거리를 c1dsq(i)로 정한다.

마찬가지로 작업을 c2에 대해서도 진행.

```
51      for d=1:mdp_params.n-1,
52          strt = (d-1)*(mdp_params.c1+mdp_params.c2);
53          for i=1:mdp_params.c1,
54              splittable(s,strt+i) = c1dsq(i) < d;
55          end;
56          strt = (d-1)*(mdp_params.c1+mdp_params.c2)+mdp_params.c1;
57          for i=1:mdp_params.c2,
58              splittable(s,strt+i) = c2dsq(i) < d;
59          end;
60      end;
61
62      % Build corresponding feature table (continuous).
63      splittablecont(s,1:mdp_params.c1) = c1dsq;
64      splittablecont(s,mdp_params.c1+1:mdp_params.c1+mdp_params.c2) = c2dsq;
65  end;
66
67  % Return feature data structure.
68  feature_data = struct('stateadjacency',stateadjacency,'splittable',splittable);
69
70  % Construct true feature map.
71  true_feature_map = sparse([],[],[],mdp_data.states,..
72      mdp_params.r_tree.total_leaves,mdp_data.states);
73  for s=1:mdp_data.states,
74      % Determine which leaf state belongs to.
75      [leaf,~] = cartcheckleaf(mdp_params.r_tree,s,feature_data);
76      true_feature_map(s,leaf) = 1;
77  end;
78
79  % Fill in the reward function.
80  R_SCALE = 5;
81  r = cartaverage(mdp_params.r_tree,feature_data)*R_SCALE;
82
83  % Optionally, replace splittable.
84  if mdp_params.continuous,
85      feature_data.splittable = splittablecont;
86  end;
87
```



cartcheckleaf.m


```
% Check which leaf of tree contains leaf.
function [leaf,val] = cartcheckleaf(tree,s,feature_data)

% Check if this is a leaf.
if tree.type == 0,
    % Return result.
    leaf = tree.index;
    val = tree.mean;
else
    % Recurse.
    if feature_data.splittable(s,tree.test) == 0,
        branch = tree.ltTree;
    else
        branch = tree.gtTree;
    end;
    [leaf,val] = cartcheckleaf(branch,s,feature_data);
end;

```

r_tree
- type 1
- test 1+stepx2 Step=C1+C2
- total_leaves 3
- ltTree: type 0, index 1, mean [0,0,0,0,0] ← leaf ∵type 0
- gtTree: type 1, test 2+step, total_leaves 2
  - gtTree: type 0, index 2, mean [1,1,1,1,1] ← leaf ∵type 0
  - ltTree: type 0, index 3, mean [-2,-2,-2,-2,-2] ← leaf ∵type 0

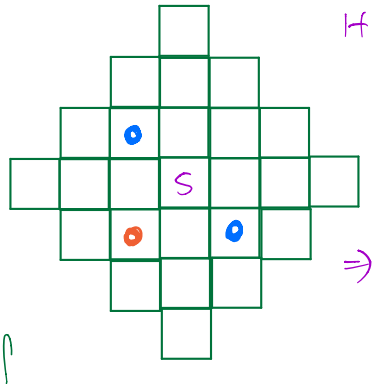
r_tree.type = 1

else

feature_data.splittable(s, r_tree.test) ⇒ 3번째 action을 판단할 수 있는 방법

1+stepx2 = 9 바깥쪽 상관 C(1) 인 object는 존재안함

C(1) = ● C(2) = ●



⇒ feature_data.splittable(s, r_tree.test) = 1

if feature_data.splittable(s, r_tree.test) == 0

branch = r_tree.ltTree. type 0, index 1, mean [0,0,0,0,0]

Cartcheckleaf(r_tree.ltTree, s, feature_data)

⇒ leaf = r_tree.ltTree.index = 1

val = r_tree.ltTree.mean = [0,0,0,0,0]

else

feature_data.splittable(s, r_tree.test) = 1

branch = r_tree.gtTree type 1, test 2+step, total_leaves 2

cartcheckleaf (r_tree.gtTree, s, feature_data)

r_tree.gtTree.type = 1

else.

if feature_data.splittable (s, r_tree.gtTree.test) == 0
2+ step

↳ 2번의 action로 도달할 수 있는 범위에 바깥쪽 셀이 C1(2)인 object로 존재여부

branch = r_tree.gtTree.ltTree
type 0
index 3
mean [-2, -2, -2, -2, -2]

cartcheckleaf (r_tree.gtTree.ltTree, s, feature_data)

⇒ leaf = r_tree.gtTree.ltTree.index = 3
val = r_tree.gtTree.ltTree.mean = [-2, -2, -2, -2, -2]

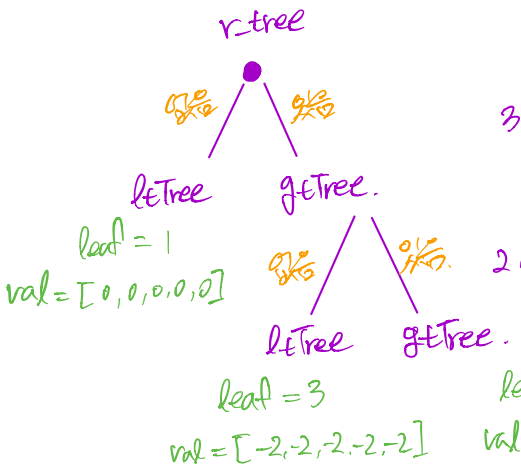
else

feature_data.splittable (s, r_tree.gtTree.test) = 1

branch = r_tree.gtTree.gtTree
type 0
index 2
mean [1, 1, 1, 1, 1]

cartcheckleaf (r_tree.gtTree.gtTree, s, feature_data)

⇒ leaf = r_tree.gtTree.gtTree.index = 2
val = r_tree.gtTree.gtTree.mean = [1, 1, 1, 1, 1]



3 cell 내에 바깥쪽 셀이 C1(1)인 object 존재여부

2 cell 내에 바깥쪽 셀이 C1(2)인 object 존재여부

cartaverage.m


```
1 % Return average reward for given regression tree.
2 function R = cartaverage(tree,feature_data)
3
4 if tree.type == 0,
5     % Simply return the average.
6     R = repmat(tree.mean,size(feature_data.splittable,1),1);
7 else
8     % Compute reward on each side.
9     ltR = cartaverage(tree.ltTree,feature_data);
10    gtR = cartaverage(tree.gtTree,feature_data);
11
12    % Combine.
13    ind = repmat(feature_data.splittable(:,tree.test),1,size(ltR,2));
14    R = (1-ind).*ltR + ind.*gtR;
15 end;
16
```

1024 x 1024

$n^2 = 1024$

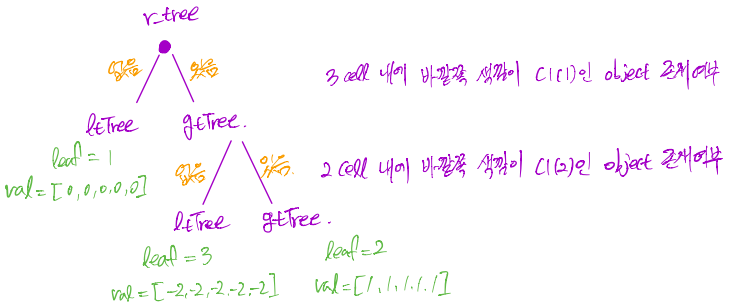
1024 x 1

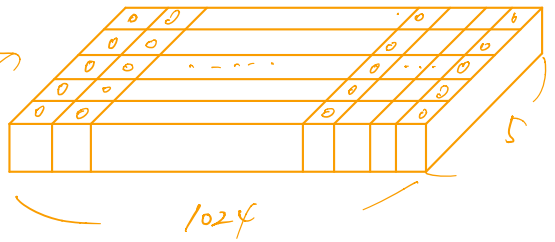
2 cell 이내에 있을 때

ltR

2 cell 이내에 없을 때

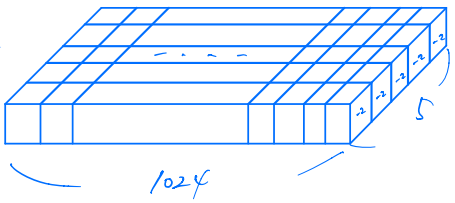
gtR.

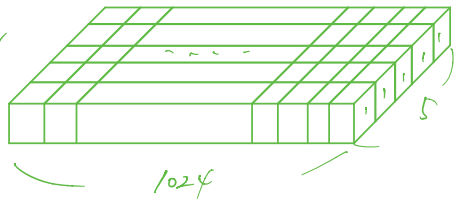


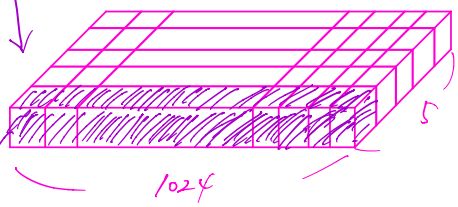


ltR = cartaverage(r_tree.ltTree.ltTree, feature_data)

gtR = cartaverage(r_tree.gtTree.gtTree, feature_data)







각 state에서 2 cell 이내에 바깥쪽 상관이 C1(2)인 object가 존재하는지 여부.

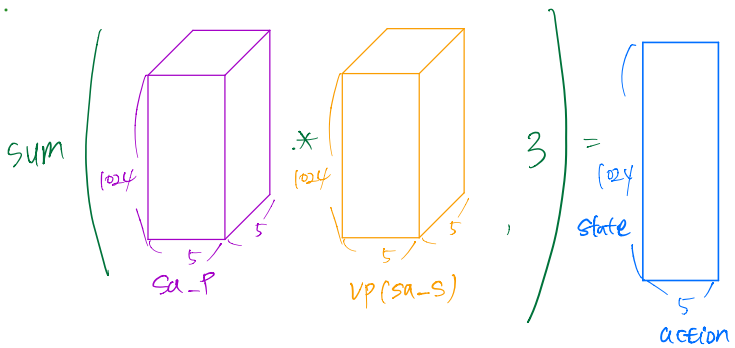
linearmdpsolve.m


```
1 % Solve a linear MDP and return value function, Q function, and policy.
2 function mdp_solution = linearmdpsolve(mdp_data,r)
3
4 % Run value iteration to compute the value function and policy.
5 [v,q,p] = linearvalueiteration(mdp_data,r);
6
7 % Return solution.
8 mdp_solution = struct('v',v,'q',q,'p',p);
9
```

value function
Q function
policy

linearvalueiteration.m

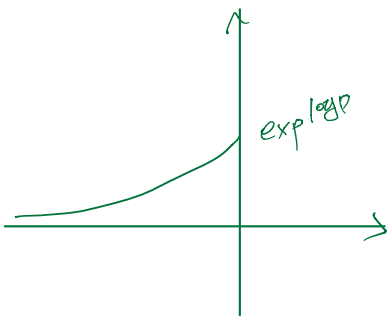

```
% Run value iteration to solve a linear MDP.
function [v,q,p,logp] = linearvalueiteration(mdp_data,r,vinit)

states = mdp_data.states;
actions = mdp_data.actions;
VITR_THRESH = 1e-4;
%VITR_THRESH = 1e-10;

% Compute log state partition function V.
if (nargin == 3)
    v = vinit;
else
    v = zeros(states,1);
end;
diff = 1.0;
while diff >= VITR_THRESH,
    % Initialize new v.
    vp = v;

    % Compute q function.
    q = r + mdp_data.discount*sum(mdp_data.sa_p.*vp(mdp_data.sa_s),3);

    % Compute softmax.
    v = maxentsoftmax(q);
    diff = max(abs(v-vp));
end;

% Compute policy.
logp = q - repmat(v,1,actions);
p = exp(logp);
```



$= R_{t+1} + \gamma \cdot \sum_{s'} (P(s'|s,a) \cdot V(s'))$

s에서 a를 취했을 때 s'으로 이동할 확률. $P(s'|s,a)$

s에서 a를 취했을 때 s'의 value $V(s')$



q와 v의 차이가 가장 적은 action을 최적의 policy로

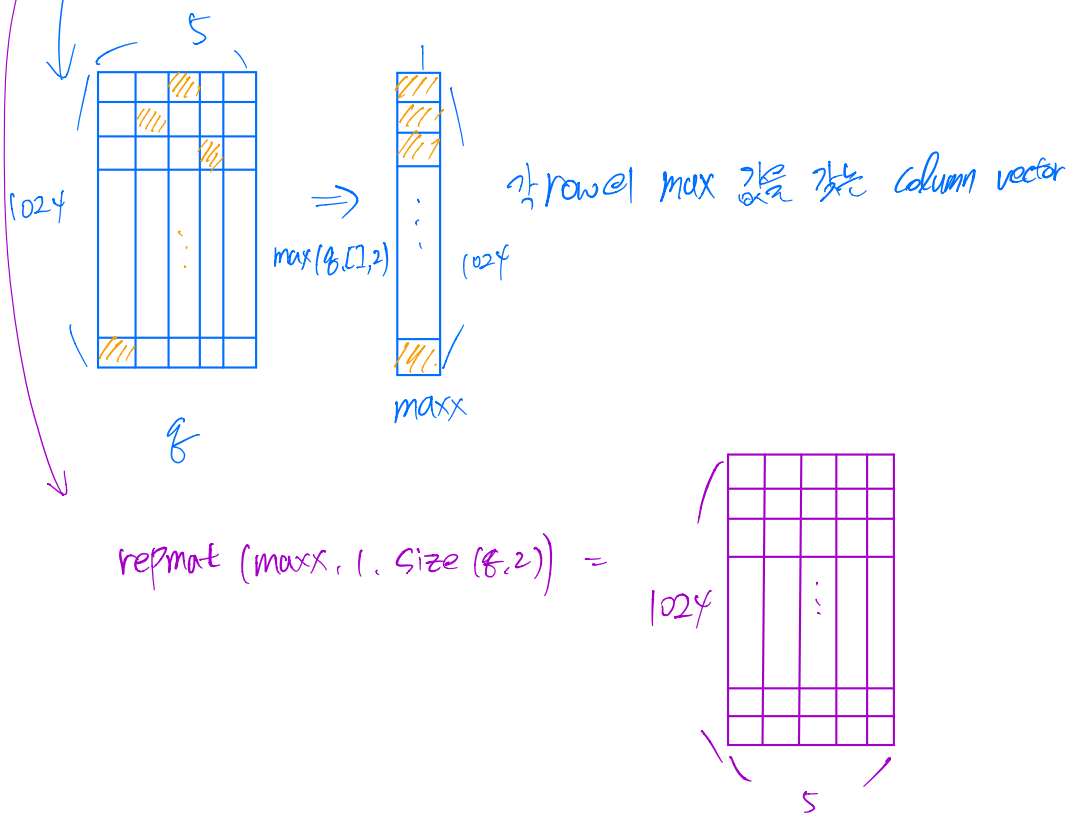
⇒ 실제로는 확률분포로 나타낸다.

maxentsoftmax.m


```
1 % Compute column-wise softmax of a matrix.
2 function v = maxentsoftmax(q)
3
4 % Find maximum elements.
5 maxx = max(q,[],2);
6
7 % Compute safe softmax.
8 v = maxx + log(sum(exp(q - repmat(maxx,1,size(q,2))),2));
9
```

$R_{t+1} + \gamma \sum_{s'} \left(P(s', 1 | s, a) \cdot V(s') \right)$

$\max_a E\left(R_{t+1} + \gamma \sum_{s'} \left(P(s', 1 | s, a) V(s') \right) \right)$



Sampleexamples.m


```
% Sample example tranjectories from the state space of a given MDP.
function example_samples = sampleexamples(mdp_model,mdp_data,mdp_solution,test_params)

% Allocate training samples.
N = test_params.training_samples; → sample 수
T = test_params.training_sample_lengths; → sample의 길이
example_samples = cell(N,T); → 각 cell에는 [state, action]이 저장

% Sample trajectories.
for i=1:N,
    % Sample initial state.
    s = ceil(rand(1,1)*mdp_data.states);

    % Run sample trajectory.
    for t=1:T,
        % Compute optimal action for current state.
        a = feval(strcat(mdp_model,'action'),mdp_data,mdp_solution,s); mdp_solution에 따라 state s에서의
                                                                       optimal action 선택.
        % Store example.
        example_samples{i,t} = [s;a]; → state와 action sample.

        % Move on to next state.
        s = feval(strcat(mdp_model,'step'),mdp_data,mdp_solution,s,a);
    end;
end;
```

linearmdpaction.m


```
1 % Return optimal action given the mdp solution.
2 function a = linearmdpaction(mdp_data,mdp_solution,s)
3
4 samp = rand(1,1);
5 total = 0;
6 for a=1:mdp_data.actions,
7     total = total+mdp_solution.p(s,a);
8     if total >= samp,
9         return;
10    end;
11 end;
12
```

rand(1,1)이 내놓은 optimal action을 취할 것인가?

mdp_solution.p

0.1	0.15	0.6	0.1	0.05

5

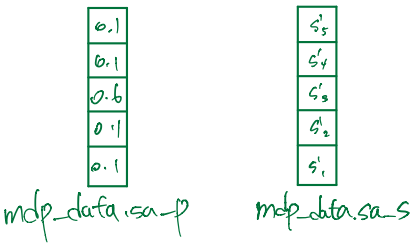
ex) samp = 0.8

→ a=3

```
1 % Take a single step with the specified action.
2 function s = linearmdpstep(mdp_data,~,s,a)
3
4 % Random sample for stochastic step.
5 r = rand(1,1);
6 sm = 0;
7 for k=1:size(mdp_data.sa_p,3),
8     sm = sm+mdp_data.sa_p(s,a,k);
9     if sm >= r,
10        s = mdp_data.sa_s(s,a,k);
11        return;
12    end;
13 end;
14
15 % Should never reach here.
16 fprintf(1,'ERROR: MDP data specifies transition distribution for state %i action %i that does not
sum to 1!Wn',...
17     s,a);
18 s = -1;
19
```

1024x5x5

State 와 action a를 취했을 때

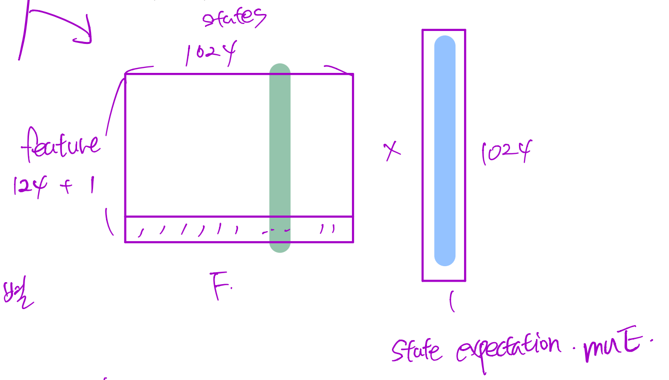


anrun.m


```
1 % Abbeel & Ng algorithm implementation (projection version).
2 function irl_result = anrun(algorithm_params,mdp_data,mdp_model,..
3     feature_data,example_samples,true_features,verbosity)
4
5 % algorithm_params - parameters of the MMP algorithm:
6 %       seed (0) - initialization for random seed
7 %       all_features (0) - use all features as a basis
8 %       true_features (0) - use true features as a basis
9 % mdp_data - definition of the MDP to be solved
10 % example_samples - cell array containing examples
11 % irl_result - result of IRL algorithm, generic and algorithm-specific:
12 %       r - inferred reward function
13 %       v - inferred value function.
14 %       q - corresponding q function.
15 %       p - corresponding policy.
16 %       time - total running time
17
18 % Fill in default parameters.
19 algorithm_params = andefaultparams(algorithm_params);
20 seed=0, all_features=1, true_features=0     = struct()
21 % Set random seed.
22 rand('seed',algorithm_params.seed);
23 tic;
24
25 % Initialize variables.
26 [states,actions,transitions] = size(mdp_data.sa_p);  = [1024 5 5]
27 [N,T] = size(example_samples); = [16 8]          1024x5x5.
28 mdp_solve = str2func(strcat(mdp_model,'solve'));
29                                  linearmdpsolve.
30 % Build feature membership matrix.
31 if algorithm_params.all_features,
32     F = feature_data.splittable;
33     % Note that we add a row of 1s to the feature matrix to ensure that we
34     % can control the reward at every state.
35     F = horzcat(F,ones(states,1));  1024×125
36 elseif algorithm_params.true_features,
37     F = true_features;
38 else
39     F = eye(states);
40 end;
41
42 % Count features.
43 features = size(F,2); =125
44 F = F';
45
46 % Construct state expectations.
47 muE = zeros(states,1);  1024×1
48 ex_s = zeros(N,T);    16×8
49 ex_a = zeros(N,T);    16×8
50 for i=1:N,
```

```
    for t=1:T,
        ex_s(i,t) = example_samples{i,t}(1);
        ex_a(i,t) = example_samples{i,t}(2);
        muE(ex_s(i,t)) = muE(ex_s(i,t)) + mdp_data.discount^(t-1);
    end;
end;
muE = muE/N;
Fmu = F*muE;

% Generate random policy.
w = rand(features,1);
r = repmat(F'*w,1,actions);
soln = standardmdpsolve(mdp_data,r);
weights = {w};
solutions = {soln};
mus = {};
mu_bars = {};
itr = 1;

% Initialize t.
t = 100.0;
told = 0.0;

while 1,
    told = t;
    % Compute feature expectations under the last policy.
    om = standardmdpfrequency(mdp_data,solutions{itr});
    mu = F*om;

    mus{itr} = mu;
    if itr == 1,
        mu_bars{itr} = mu;
    end;

    % Increment iteration count.
    itr = itr + 1;

    % Compute t and w using projection.
    if itr == 2,
        % use existing mu.
        mu_bar = mu_bars{itr-1};
        w = Fmu - mu_bar;
        t = norm(Fmu - mu_bar);
    else
        mu_bar_prev = mu_bars{itr-2};
        num = (mu-mu_bar_prev)'*(Fmu-mu_bar_prev);
        denom = (mu-mu_bar_prev)'*(mu-mu_bar_prev);
        ratio = num/denom;
        mu_bar = mu_bar_prev + ratio*(mu-mu_bar);

```

$\mu_E = \frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{\infty}\gamma^t \phi(s_t^{(i)})$



w = rand(features,1); → 125×1

$R = w^T \phi$ → 각 state 별 reward .

random 한 w에 대해 reward를 구하고, 이를 이용해 mdp를 풀어서 value function, Q function, policy를 찾는다. ($\pi^{(0)}$)

$i = 1$

225	225	225	257	257	257	257	257

$t=1$: $\phi(s_0^{(1)}) = \phi(225) = F(:,225)$

$muE(225) = muE(225) + 0.9^0 = 1$

$t=2$: $muE(225) = muE(225) + 0.9^1 = 1.9$

$t=3$: $muE(225) = muE(225) + 0.9^2 = 1.9 + 0.81 = 2.71$

$t=4$: $muE(257) = muE(257) + 0.9^3 = 0.9^3$

⋮

$t=8$: $muE(257) = 0.9^3 + 0.9^4 + 0.9^5 + 0.9^6 + 0.9^7$

주어진 policy에 대해 feature expectation을 구함.

state expectation ? $E[\sum_{t=0}^{\infty}\gamma^t]$?

논문에서의 $\bar{\mu}^{(0)} = \mu^{(0)}$

$\bar{\mu}^{(0)} = \mu^{(0)}$

$\mu^{(i-2)}$

$(\mu^{(i-1)} - \bar{\mu}^{(i-2)})^T(\mu_E - \bar{\mu}^{(i-2)})$

$(\mu^{(i-1)} - \bar{\mu}^{(i-2)})^T(\mu^{(i-1)} - \bar{\mu}^{(i-2)})$

$$\bar{\mu}^{(i-1)} = \bar{\mu}^{(i-2)} + \frac{(\mu^{(i-1)} - \bar{\mu}^{(i-2)})^T(\mu_E - \bar{\mu}^{(i-2)})}{(\mu^{(i-1)} - \bar{\mu}^{(i-2)})^T(\mu^{(i-1)} - \bar{\mu}^{(i-2)})}(\mu^{(i-1)} - \bar{\mu}^{(i-2)})$$

```
101         w = Fmu - mu_bar;
102         t = norm(Fmu - mu_bar);
103         mu_bars{itr-1} = mu_bar;
104     end;
105
106     % Recompute optimal policy using new weights.
107     r = repmat(F'*w,1,actions);
108     soln = standardmdpsolve(mdp_data,r);
109     weights{itr} = w;
110     solutions{itr} = soln;
111
112     % Check convergence.
113     if (abs(t-told) <= 0.0001),
114         break;
115     end;
116
117     % Print t.
118     if verbosity ~= 0,
119         fprintf(1,'Completed IRL iteration, t=%f₩n',t);
120     end;
121 end;
122
123 % Compute mu for last policy.
124 om = standardmdpfrequency(mdp_data,solutions{itr});
125 mu = F*om;
126 mus{itr} = mu;
127
128 % Construct matrix.
129 mu_mat = zeros(features,itr);
130 for i=1:itr,
131     mu_mat(:,i) = mus{i};
132 end;
133
134 % Solve optimization to determine lambda weights.
135 cvx_begin
136     if verbosity ~= 0,
137         cvx_quiet(false);
138     else
139         cvx_quiet(true);
140     end;
141     variable mu(features);
142     variable lambda(itr);
143     minimize(sum_square(mu-Fmu));
144     subject to
145         mu == mu_mat*lambda;
146         lambda >= zeros(itr,1);
147         lambda'*ones(itr,1) == 1;
148 cvx_end
149
150 % In Abbeel & Ng's algorithm, we should use the weights lambda to construct
```

```
% a stochastic policy. However, here we are evaluating IRL algorithms, so
% we must return a single reward. To this end, we'll simply pick the reward
% with the largest weight lambda.
[~,idx] = max(lambda);
w = weights{idx};

% Compute reward.
r = repmat(F'*w,1,actions);

% Compute policies.
soln = mdp_solve(mdp_data,r);
v = soln.v;
q = soln.q;
p = soln.p;
r_itr = cell(1,1);
tree_r_itr = cell(1,1);
p_itr = cell(1,1);
tree_p_itr = cell(1,1);
wts_itr = cell(1,1);
r_itr{1} = r;
tree_r_itr{1} = r;
p_itr{1} = soln.p;
tree_p_itr{1} = p_itr{1};
wts_itr{1} = w;
time = toc;

% Construct returned structure.
irl_result = struct('r',r,'v',v,'p',p,'q',q,'r_itr',{r_itr},'model_itr',{wts_itr},...
    'model_r_itr',{tree_r_itr},'p_itr',{p_itr},'model_p_itr',{tree_p_itr},...
    'time',time);

```

andefaultparams.m


```
% Fill in default parameters for the Abbeel & Ng algorithm.
function algorithm_params = andefaultparams(algorithm_params)

% Create default parameters.
default_params = struct(...
    'seed',0,...
    'all_features',1,...
    'true_features',0);

% Set parameters.
algorithm_params = filldefaultparams(algorithm_params,default_params);
```

입력된 algorithm_params 중 비어있는 field 값을 default 값으로 채워줌.

standardmdpsolve.m


```
1 % Solve a standard MDP and return value function, Q function, and policy.
2 function mdp_solution = standardmdpsolve(mdp_data,r)
3
4 % Run value iteration to compute the value function.
5 v = stdvalueiteration(mdp_data,r);
6
7 % Compute Q function and policy.
8 [q,p] = stdpolicy(mdp_data,r,v);
9
10 % Return solution.
11 mdp_solution = struct('v',v,'q',q,'p',p);
12
```

1024x5.

stdvalueiteration.m


```
1 % Run value iteration to solve a standard MDP.
2 function v = stdvalueiteration(mdp_data,r,vinit)
3
4 % Allocate initial value function & variables.
5 diff = 1.0;
6 if (nargin == 3)
7     vn = vinit;
8 else
9     vn = zeros(mdp_data.states,1);
10 end;
11
12 % Perform value iteration.
13 while diff >= 0.0001,
14     vp = vn;
15     vn = max(r + sum(mdp_data.sa_p.*vp(mdp_data.sa_s),3)*mdp_data.discount,[],2);
16     diff = max(abs(vn-vp));
17 end;
18
19 % Return value function.
20 v = vn;
21
```

1024

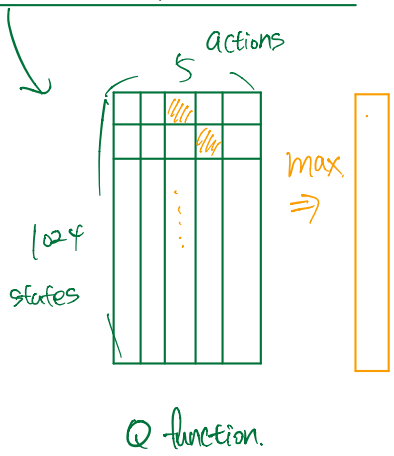
$w^T\phi$

$R_{t+1} + (P(s'|s,a) * V(s')) \cdot \gamma$

1024x5x5 1024x5x5

각 state 마다 value 값이 가장 큰 action 의 value 를 value function 으로 취한다.

```
1 % Given reward and value functions, solve for q function and policy.
2 function [q,p] = stdpolicy(mdp_data,r,v)
3
4 % Compute Q function.
5 q = r + sum(mdp_data.sa_p.*v(mdp_data.sa_s),3)*mdp_data.discount;
6
7 % Compute policy.
8 [~,p] = max(q,[],2);
9
```



```
1 % Compute the occupancy measure of the MDP given a policy.
2 function x = standardmdpfrequency(mdp_data,mdp_solution)
3
4 % Build flow constraint matrix.
5 A = sparse([],[],[],mdp_data.states,mdp_data.states,...
6     mdp_data.states*size(mdp_data.sa_p,3));
7 for s=1:mdp_data.states,
8     A(s,s) = A(s,s)+1;
9     a = mdp_solution.p(s);
10    for k=1:size(mdp_data.sa_p,3),
11        sp = mdp_data.sa_s(s,a,k);
12        A(sp,s) = A(sp,s) - mdp_data.discount*mdp_data.sa_p(s,a,k);
13    end;
14 end;
15
16 % Solve linear system to get occupancy measure.
17 x = A\((1/mdp_data.states)*ones(mdp_data.states,1));
18
```

1024 × 1024 크기 sparse matrix를 생성한다. 이때 0이 아닌 element의 수는 1024×5를 넘지 않는다.

⇒ S에서 policy π에 따라 action a를 취했을때

$$A\alpha = \frac{1}{1024}\begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix}_{1024}$$

Occupancy Frequencies

The occupancy frequency $f(s,a)$ expresses the total discounted probability of being in state s and taking action a

Given $f(s,a)$ we can recover the policy π

$$\pi(s,a) = \frac{f^{\pi}(s,a)}{\sum_{a'} f^{\pi}(s,a')}$$

The value of optimal π^* corresponds to the value of the optimal f^*

$$V_r^{\pi^*} = r \cdot f^*$$

evaluateirl.m


```
% Evaluate result returned by IRL algorithm on given MDP.
function test_result = evaluateirl(irl_result,true_r,example_samples,...
    mdp_data,mdp_params,mdp_solution,mdp,~,test_models,test_metrics,...
    feature_data,true_feature_map)

% test_result - data structure encapsulating generic test results:
%   metric_scores - array of results from each metric, for each MDP type
%   irl_result - copy of result from IRL algorithm
%   true_r - true reward function
%   example_samples - copy of example samples
%   mdp_data - copy of MDP data
%   mdp_params - copy of MDP parameters
%   mdp_solution - copy of true MDP solution
%   feature_data - copy of feature data
%   mdp - type of MDP used

metric_scores = cell(length(test_models),length(test_metrics));
for m=1:length(test_models),     standardmdp, linearmdp 8
    % Evaluate for each test model.
    cur_model = test_models{m};
    mdp_solve = str2func(strcat(cur_model,'solve'));
    irl_soln = mdp_solve(mdp_data,irl_result.r);
    mdp_soln = mdp_solve(mdp_data,true_r);
    irl_r = irl_result.r;

    % Evaluate each metric.
    for k=1:length(test_metrics),
        cur_metric = test_metrics{k};
        metric_scores{m,k} = feval(strcat(cur_metric,'score'),mdp_soln,...
            true_r,irl_soln,irl_r,feature_data,true_feature_map,...
            mdp_data,mdp_params,cur_model);
    end;
end;

% Build results structure.
test_result = struct(...
    'metric_scores',{metric_scores},...
    'irl_result',irl_result,...
    'true_r',true_r,...
    'example_samples',{example_samples},...
    'test_models',{test_models},...
    'test_metrics',{test_metrics},...
    'mdp_data',mdp_data,...
    'mdp_params',feval(strcat(mdp,'defaultparams'),mdp_params),...
    'mdp_solution',mdp_solution,...
    'feature_data',feature_data,...
    'mdp',mdp);
```