Methods of Artificial Intelligence midterm recap

...

General remarks

Yesterday, only 15 people had filled out the test exam. The test exam provides a good overview of the questions that will be asked in the midterm exam, so please take time to test your knowledge!

CSP consistency

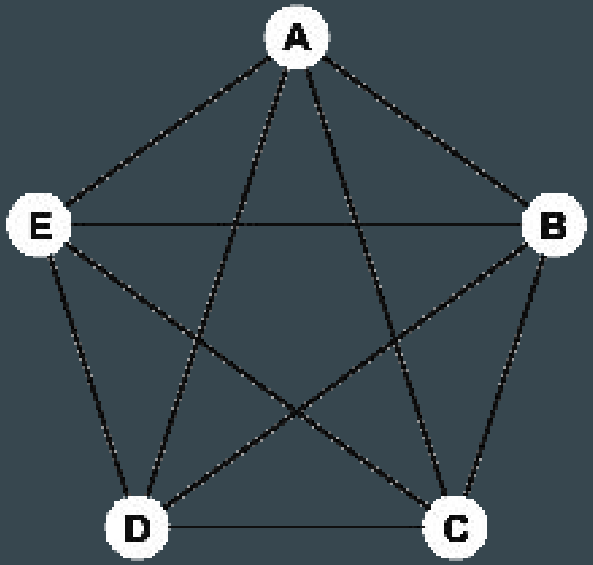
Consistency types

- Node consistency
- Arc consistency
- Path consistency
- k-consistency

Node consistency

suppose the domain {1,2,3,4,5}

with the constraint node E != 3



Node consistency

Consider the domain D = {1,2,3,4,5}

with the constraint of an arbitrary node N != 3

Since 3 $\in$ D, the graph is Node inconsistent!

Deleting 3 from the domain would make the graph node-consistent again:

D = {1,2,4,5}

Arc consistency

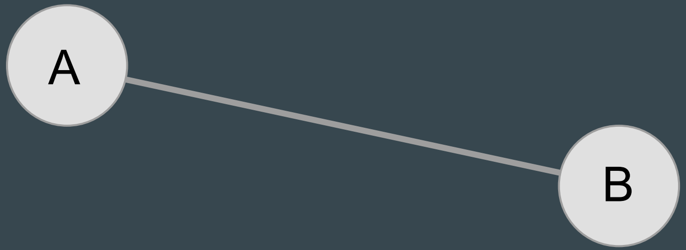
Suppose the set {1,2} and a graph G

if:

A = 1

B = 1

The graph is arc-inconsistent!



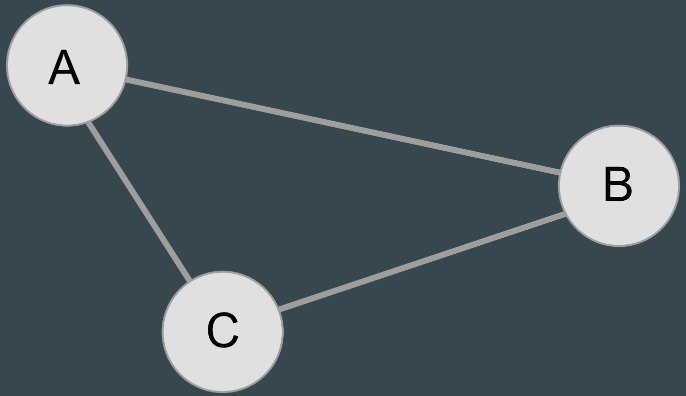
Path consistency

Suppose the Domain D = {1,2,3}:

We establish **arc-consistency** between A and B by assigning A = 1 and B = 2

Nodes A and B are path consistent with node C only if we assign C = 3, making it arc consistent with A and B!

(path consistency is arc consistency between an already arc consistent pairing of nodes with another node)



k-consistency

Node consistency = 1-consistent

Arc consistency = 2-consistent

path-consistency = 3-consistent

strong k consistent: consider a $j \in \{1,2,\ldots,k\}$. A graph is strong k-consistent if it is consistent for every $j \leq k$

A graph is k-consistent if it can be extended from a (k-1)-consistent state to a k-consistent state.

A k-consistent graph DOES NOT need to be consistent in general! i.e. with k+1 nodes

VIPS questions concerning consistency

- Every arc consistent CSP is node consistent. 🚫
- Every node consistent CSP is consistent. 🚫
- Every node consistent CSP is arc consistent. 🚫
- Every strong k-consistent CSP is strong (k-1)-consistent. 🙂
- Every strong (k-1)-consistent CSP is strong k-consistent. 🚫
- Every consistent CSP is node consistent. 🚫

Local search

Hill climbing vs Simulated annealing

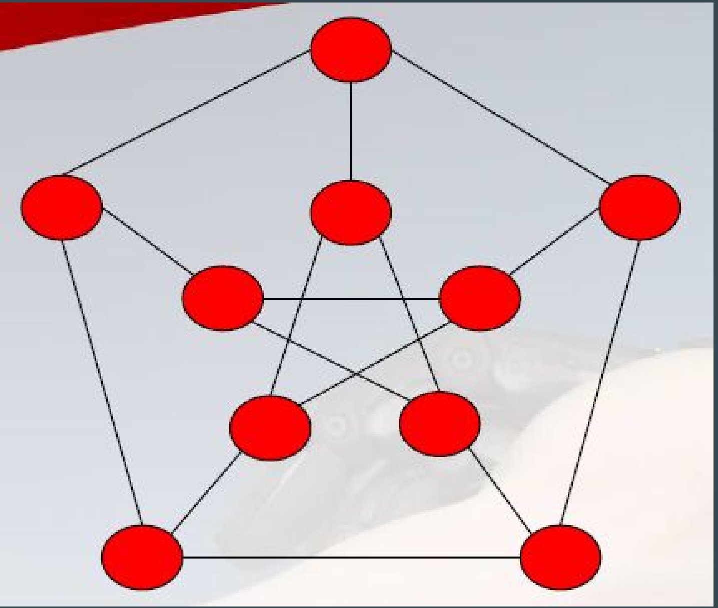
Stochastic Hill Climbing can make choices that decrease the objective function.

->wrong

```
current ← select random initial state
do
   neighbor ← random neighbor of current with higher value
   if neighbor = null
      return current
   current←neighbor
until termination condition is met
```

Solving CSPs with Local Search

Suppose that the CSP has n variables and m constraints.



n = 10

m =

domain = {1,2,3}

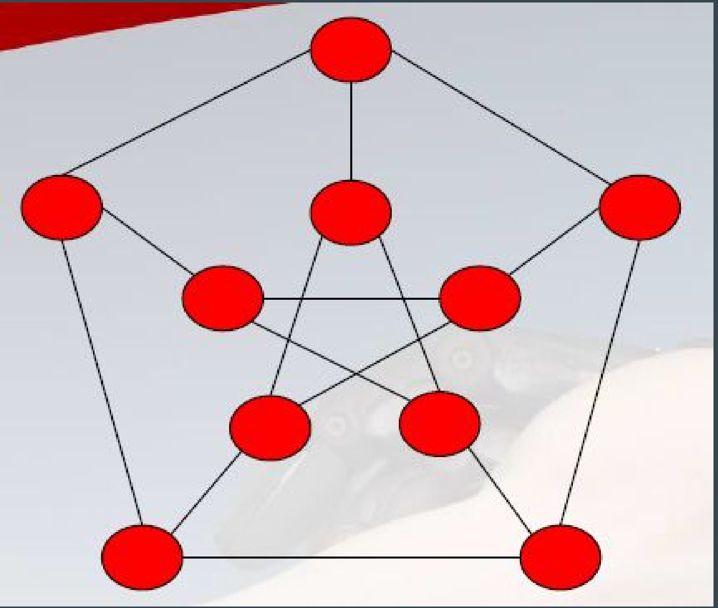
a state is a complete assignment. In this example: (1,1,1,1,1,1,1,1,1,1)

#neighbors = #variables * #values in domain other than the current one

=10 * 2 = 20

Solving CSPs with Local Search

Suppose that the CSP has n variables and m constraints.



Objective function:
value of state is defined as the number of constraints that are satisfied in the state

$n = 10$

$m = 15$

$f(s) = 0$

Solving CSPs with Local Search

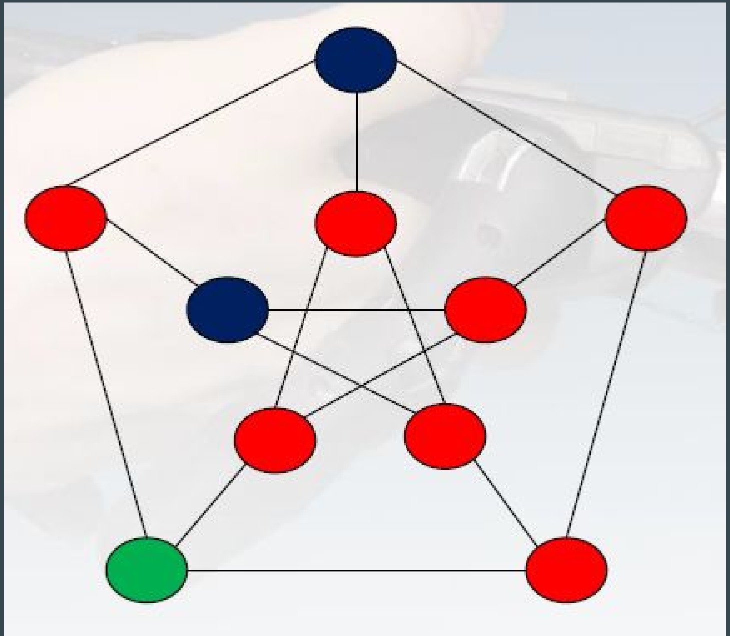
Suppose that the CSP has n variables and m constraints.

$n = 10$, $m = 15$

1. When our local search algorithm finds a solution with value n, it found a consistent assignment and the CSP is consistent.
2. When our local search algorithm finds a solution with value m, it found a consistent assignment and the CSP is consistent.
3. When our local search algorithm finds a solution with value less than n, there is no consistent assignment and the CSP is inconsistent.
4. When our local search algorithm finds a solution with value less than m, there is no consistent assignment and the CSP is inconsistent.

Solving CSPs with Local Search

Suppose that the CSP has n variables and m constraints.



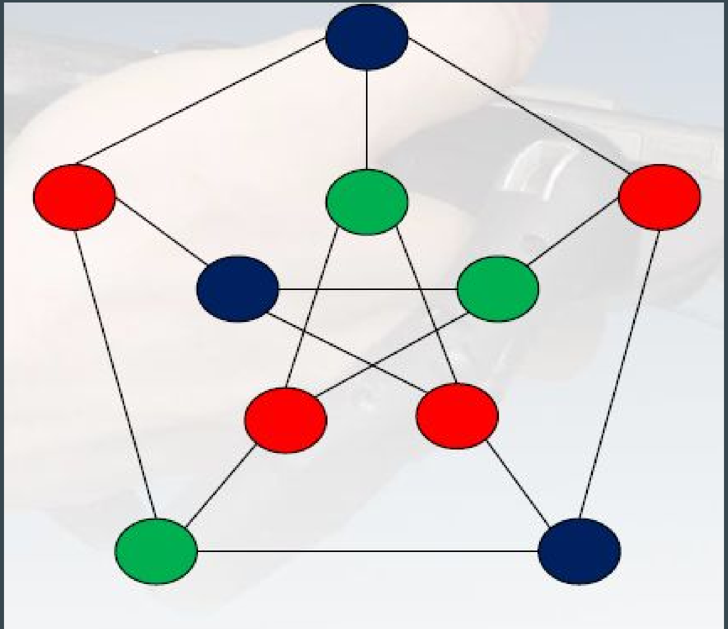
n = 10

m = 15

f(s) = 9

Solving CSPs with Local Search

Suppose that the CSP has n variables and m constraints.



n = 10

m = 15

f(s) = 15

Genetic Algorithms

Uniform-Order-Crossover

Child 1 is created by using genes of parent 1 at 1-positions. Fill gaps with the remaining elements according to the order given by parent 2.
Child 2 is created by switching the roles of parent 1 and 2.

parent 1: 456123
parent 2: 312546

template: 010101

child 1: ______
child 2: ______

Uniform-Order-Crossover

Child 1 is created by using genes of parent 1 at 1-positions. Fill gaps with the remaining elements according to the order given by parent 2.
Child 2 is created by switching the roles of parent 1 and 2.

parent 1: 4~~5~~6~~1~~2~~3~~ leftover: 4,6,2
parent 2: 3~~1~~2~~5~~4~~6~~ leftover: 3,2,4

template: 010101

child 1: _5_1_3
child 2: _1_5_6

Uniform-Order-Crossover

Child 1 is created by using genes of parent 1 at 1-positions. Fill gaps with the remaining elements according to the order given by parent 2.
Child 2 is created by switching the roles of parent 1 and 2.

parent 1: 4~~5~~6~~1~~2~~3~~ leftover: 4,6,2
parent 2: 3~~1~~2~~5~~4~~6~~ leftover: 3,2,4

template: 010101

child 1: _5_1_3
child 2: _1_5_6

Uniform-Order-Crossover

Child 1 is created by using genes of parent 1 at 1-positions. Fill gaps with the remaining elements according to the order given by parent 2.
Child 2 is created by switching the roles of parent 1 and 2.

parent 1: 4~~5~~6~~1~~2~~3~~ leftover: 4,6,2
parent 2: 3~~1~~2~~5~~4~~6~~ leftover: 3,2,4

template: 010101

child 1: 254163
child 2: 412536

Theorem Proving

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) “≤” 3 Human(X)

Mammal(johannes) “ = “ “+” 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) "≤" 3 Human(X)

Mammal(johannes) " = " "+" 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) "≤" 3 Human(X)

Mammal(johannes) " = " "+" 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) "≤" 3 Human(X)

Mammal(johannes) " = " "+" 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) “≤” 3 Human(X)

Mammal(johannes) “ = “ “+” 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) “≤” 3 Human(X)

Mammal(johannes) “ = “ “+” 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) “≤” 3 Human(X)

Mammal(johannes) “ = “ “+” 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates Mammal(A) -> Animal(A) "≤" 3 Human(X)

Mammal(johannes) " = " "+" 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

socrates   Mammal(A) -> Animal(A)   "≤"   3   Human(X)

Mammal(johannes)   " = "   "+"   4+Z   Human   2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

Terms:

socrates Mammal(A) -> Animal(A) "≤" 3 Human(X)

Mammal(johannes) " = " "+" 4+Z Human 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

Terms: socrates 3 Human 4+Z

Mammal(A) -> Animal(A) "≤" Human(X)

Mammal(johannes) " = " "+" 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

Terms: socrates 3 Human 4+Z

Predicate symbols:

Mammal(A) -> Animal(A) "≤" Human(X)

Mammal(johannes) " = " "+" 2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

Terms: socrates   3 Human   4+Z

Predicate symbols: Animal(…), Human(…), Mammal(…), “≤”, “ = “

Mammal(A) -> Animal(A)

Human(X)

Mammal(johannes)

2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

Terms: socrates 3 Human 4+Z

Predicate symbols: Human(...), Mammal(...), “≤”, “ = “

Formulas:

Mammal(A) -> Animal(A)

Human(X)

Mammal(johannes)

2+3=5

First-Order Predicate Logic

Terms, Predicates, Formulas, Function Symbol, Relation Symbol, Constants, Variables

Terms: socrates 3 Human 4+Z

Predicate symbols: Human(...), Mammal(...), "≤", " = "

Formulas: Mammal(A) -> Animal(A), Human(X), Mammal(johannes), 2+3=5

Normal Clause Form

$\exists x(P(x) \rightarrow \forall y Q(y))$

Steps for conversion:

(1) Remove Implications.
(2) Reduce scope of negations.
(3) Skolemization (remove existential quantifiers).
(4) Standardize (rename) variables.
(5) Prenex-form.
(6) Conjunctive normal form (CNF).
(7) Eliminate conjunctions and rename variables if necessary. Each conjunct should have a different set of variables.
(8) Eliminate universal quantifiers.

Normal Clause Form

$\exists x(P(x) \rightarrow \forall y Q(y))$

(1) $\exists x(\neg P(x) \vee \forall y Q(y))$

Steps for conversion:

(1) Remove Implications.
(2) Reduce scope of negations.
(3) Skolemization (remove existential quantifiers).
(4) Standardize (rename) variables.
(5) Prenex-form.
(6) Conjunctive normal form (CNF).
(7) Eliminate conjunctions and rename variables if necessary. Each conjunct should have a different set of variables.
(8) Eliminate universal quantifiers.

Normal Clause Form

$\exists x(P(x) \rightarrow \forall y Q(y))$

(1) $\exists x(\neg P(x) \vee \forall y Q(y))$

(3) $\neg P(c_x) \vee \forall y Q(y)$

Steps for conversion:

(1) Remove Implications.
(2) Reduce scope of negations.
(3) Skolemization (remove existential quantifiers).
(4) Standardize (rename) variables.
(5) Prenex-form.
(6) Conjunctive normal form (CNF).
(7) Eliminate conjunctions and rename variables if necessary. Each conjunct should have a different set of variables.
(8) Eliminate universal quantifiers.

Normal Clause Form

$\exists x(P(x) \rightarrow \forall y Q(y))$

(1) $\exists x(\neg P(x) \vee \forall y Q(y))$

(3) $\neg P(c_x) \vee \forall y Q(y)$

(8) $\neg P(c_x) \vee Q(y)$

Steps for conversion:

(1) Remove Implications.
(2) Reduce scope of negations.
(3) Skolemization (remove existential quantifiers).
(4) Standardize (rename) variables.
(5) Prenex-form.
(6) Conjunctive normal form (CNF).
(7) Eliminate conjunctions and rename variables if necessary. Each conjunct should have a different set of variables.
(8) Eliminate universal quantifiers.

Markov decision processes

Basic idea

For a given problem, we want to find the most optimal (or good enough) solution-strategy

if there exists a strategy: it is to be evaluated

if there exists no strategy: it is to be found

Policy iteration

consider an MDP with (S, A, p, r):

S = set of states

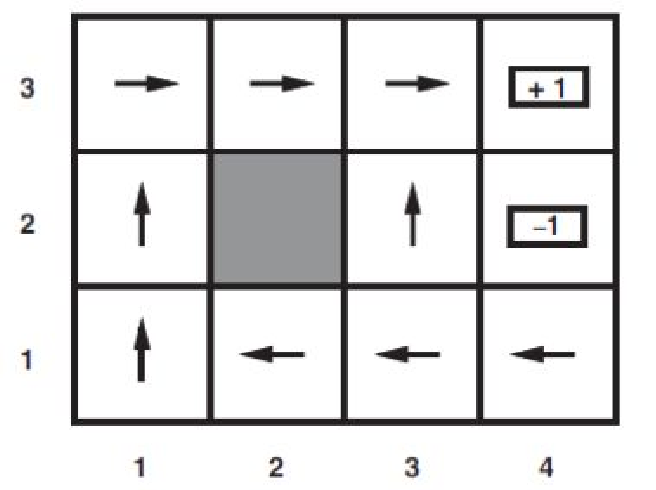
A = set of actions

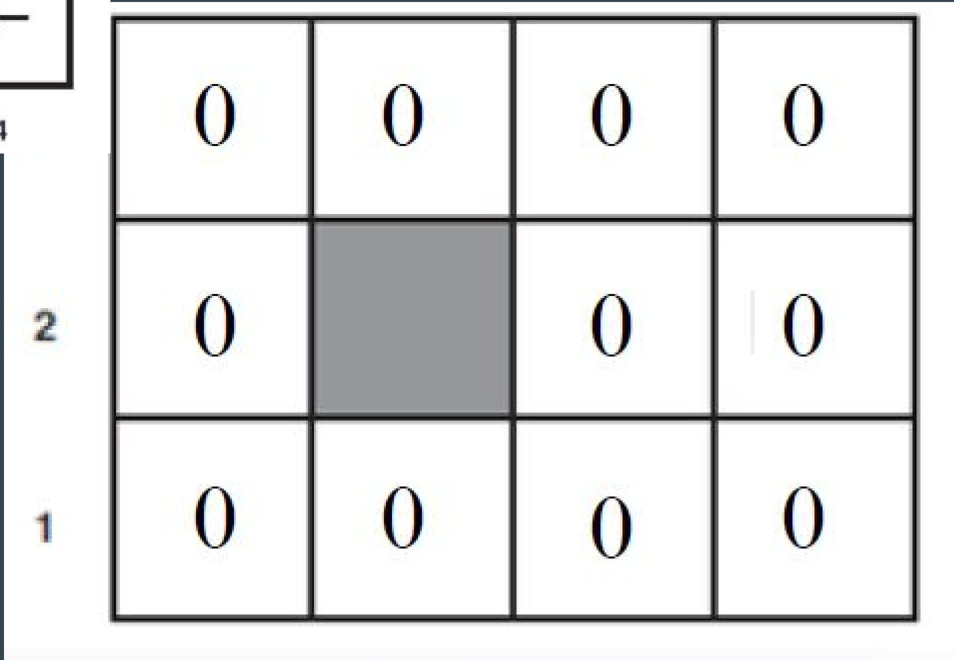
p = probability that action *a* in state *s* will lead to states *s1*

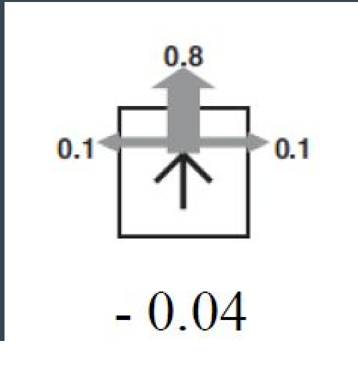
r = expected rewards from going from *s* to *s1*

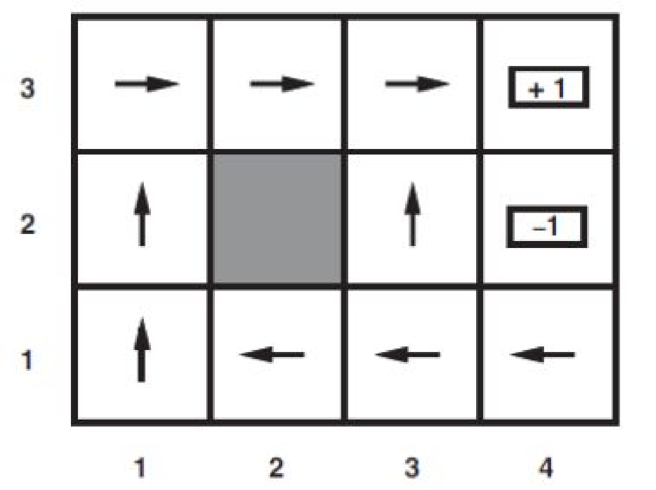
Discount factor γ

deterministic policy π


3
2
1
+ 1
–1
1
2
3
4

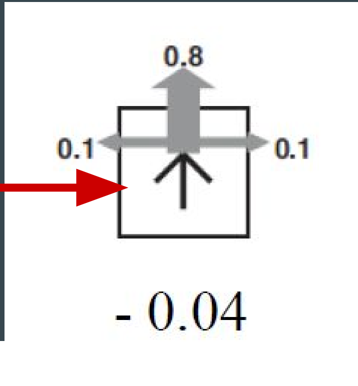

0 0 0 0
2 0 0 0
1 0 0 0 0


0.8
0.1
0.1
- 0.04



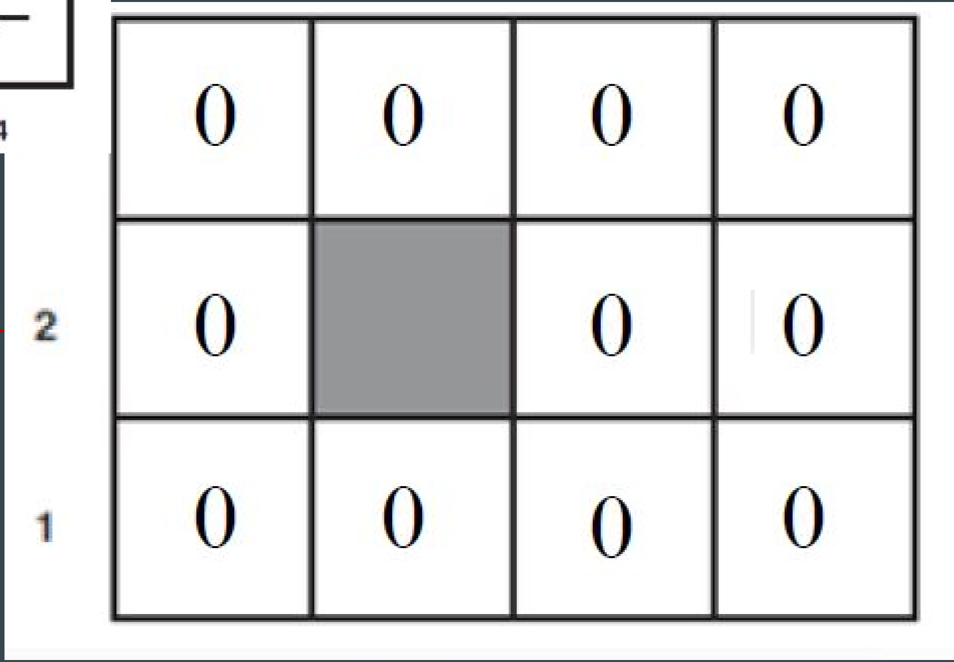
Initial policy

Probability of state-action function



expected reward per move

Initial values



Initialize value estimate v and policy π arbritrarily

do

  v ← evaluate π *(perform policy evaluation)*

  for each s in S

    π(s) ← greedy action with respect to v

until policy is stable (does not change anymore)

return π

3	-0.04	-0.04	-0.04	1
2	-0.04		-0.04	-1
1		-0.04	-0.04	-0.04

$$\begin{array}{c} -0.04 \\ + 0.8 * -0.04 \\ + 0.1 * -0.04 \\ + 0.1 * -0.04 \\ \hline = -0.08 \end{array}$$

Let’s continue on the blackboard
(if there is time)

Convergence

Convergence is guaranteed under some assumptions such as:

γ != 1!

why?

Convergence

Convergence is guaranteed under some assumptions such as:

γ != 1!

why?

Because if $\gamma = 1$, the algorithm will search for a reward after infinite steps, this may lead to a jumping in between equally optimal policies

A possible solution to this is to simply let the algorithm terminate after a number of iterations rather than when it falls under a certain ε value.

Good luck in the exam!