# Markov Decision Process MDP

- Create a policy when acting under uncertainty
- 1-Markov assumption: next state only depends on current state
- **Components:**
  - Set of states: S    Set of actions A
  - Initial state: $s_0$
  - Transition model: T(s, a, s') —> [0, 1]    if in s and perform a, what prob of getting to s' $P(s' | s, a)$
  - Reward function: R(s) —> real #    How much value do you get for being in a state

Action UP

| | (1,1) | (2,1) | (2,2) | ··· | (3,3) | ··· | (1,2) ··· |
|---|---|---|---|---|---|---|---|
| (1,1) | 0.1 | 0.1 | | | (0.0) | | 0.8 |
| (2,1) | 0.1 | 0.8 | | | 0.1 | | 0.0 |
| (22) | | | | | | | |
| ⋮ | | | | | | | |
| (3,3) | | | | | | | |
| ⋮ | | | | | | | |
| (1,2) | | | | | | | |

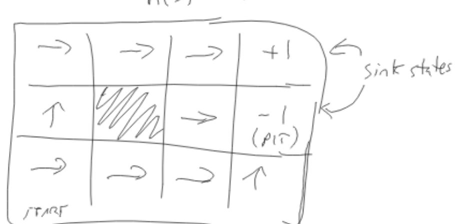
## MDP

$R(s) \rightarrow \mathbb{R}$

Reward defines what optimal is.

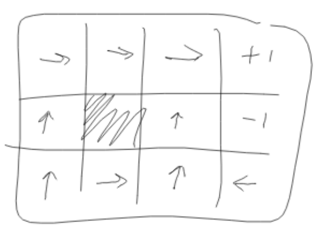
- Solution is a policy π: s —> a
- π(s) is the action to take from state s
- π*(s) is the optimal action to take from state s
  - Maximizes reward over time, giving the highest expected utility

$R(s) \leq -1.7$




Suicide

$-0.43 < R(s) < -0.06$



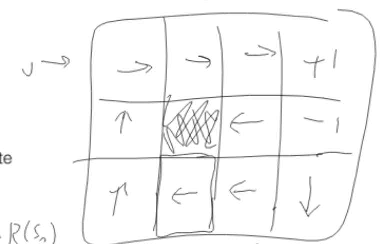

Risk taker

# Calculating utility

- What is it worth to the agent to be in a state?
- Utility of a state s is the amount of reward accumulated since starting in state $s_0$ before arriving in s
- Depends on how you got there
- Utility of a plan acts like a heuristic, indicating which route is better

$\cancel{U([s_0, s_1, s_2, \dots s_n])} = R(s_0) + R(s_1) + \dots + R(s_n)$

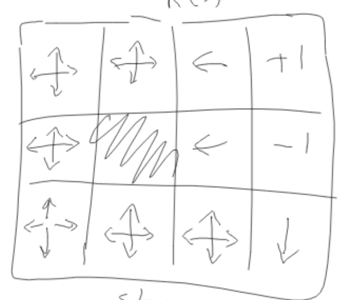
$-0.2 < R(s) < 0$




~~Paranoid~~

Conservative

$R(s) > 0$




Stoner

$R(s, a)$

# Calculating utility

- **Discounted reward:**
  - $U_h([s_0, s_1, \dots, s_n]) = \underline{R(s_0)} + \underline{\gamma R(s_1)} + \underline{\gamma^2 R(s_2)} + + \underline{\gamma^3 R(s_3)} \dots + \gamma^n R(s_n)$
  - $\gamma$ is discount factor, [0, 1]

- If you knew the utility of every state, the policy is just the action that moves the agent to the successor with the highest utility

| | | | |
|---|---|---|---|
| 0.812 | 0.868 | 0.918 | +1 |
| 0.762 | | 0.660 | −1 |
| 0.705 START | 0.655 | 0.611 | 0.388 |

U (all future sequences)

$$\left.\begin{array}{l} U(R) \\ U(U) \\ U(UR) \\ U(DUR) \end{array}\right\} \text{Max}$$

⋮

$$\pi^*(s) = \underset{a \in A}{\operatorname{argmax}} \left( \sum_{s' \in S} \left( T(s,a,s') \times U(s') \right) \right)$$

for all other states

probability of getting to that state if a is executed

utility of successor

?

- Need to compute U(s)
- Utility of state s is the immediate reward R(s) plus expected discounted utility of future states
- **Bellman Equation:**

?

$$U(s) = R(s) + \gamma \times \max_{a\in A} \left( \sum_{s'\in S} \left( T(s,a,s') \times U(s') \right) \right)$$

me

My reward

discount

probability of successor state

Recursive!

# Value Iteration

- Start with arbitrary utility values and iteratively update the values until they converge on true utility
    - Recall: utility of a state is a function on immediate reward and the expected utilities of all neighbors
    - $U_i(s)$ is the utility of state s at the ith iteration
- **Bellman update:**

$$U_{i+1}(s) = R(s) + \gamma \times \max_{a \in A} \left( \sum_{s' \in S} \left( T(s,a,s') \times U_i(s') \right) \right)$$

my utility at i+1

my reward

Utility of my successors at the previous iteration

MDP

States $S$
Actions $A$
initial state $s_0$
$T(s', a, s)$
$R(s)$ $(R(s,a))$

$\Rightarrow \pi(s) \to a$

Value Iteration

# Algorithm

- Set $U_0(s)$ to arbitrary starting value for all s
- Do until utilities converge (difference less than ε):
  - i = i +1
  - For each state s do:
    - Compute $U_i$ using the Bellman update (using $U_{i-1}$)