

Implementing Value Iteration

1. Make it work
2. Make it right
3. Make it fast

Implementing Value Iteration

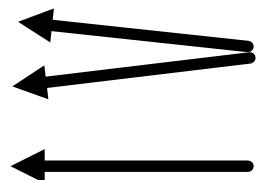
1. Make it work
2. Make it right
3. Make it fast



Implementing Value Iteration

1. Make it work ← Problem 4
2. Make it right ← Problem 4
3. Make it fast ← Problem 5

Implementing Value Iteration

1. Make it work
 2. Make it right
 3. Make it fast
- 
- Problem 4
- Problem 5

First step for making it fast (in any language, not just julia):

Implementing Value Iteration

1. Make it work
 2. Make it right
 3. Make it fast
-
- ```
graph LR; A[Problem 4] --> B[2. Make it right]; A --> C[3. Make it fast];
```

First step for making it fast (in any language, not just julia):

Find out what is slow (by profiling)!

# Bellman Operator

$$U' = B[U]$$

$$B[U](s) = \max_a \underbrace{\left( R(s, a) + \gamma \sum_{s'} T(s' | s, a) U(s') \right)}_{Q(s, a)}$$

# Bellman Operator

$$U' = B[U]$$

$$B[U](s) = \max_a \underbrace{\left( R(s, a) + \gamma \sum_{s'} T(s' | s, a) U(s') \right)}_{Q(s, a)}$$

$i$  = index of  $s$ ;  $j$  = index of  $s'$

Naive implementation:

$$U'[i] = \max_a \left( R[a][i] + \gamma \sum_j T[a][i, j] U[j] \right)$$

$\nwarrow$  Dict of vectors     $\curvearrowright$  Dict of matrices

# Bellman Operator

*i<sub>s</sub>, j<sub>s</sub>, values*

$$U' = B[U]$$

$$B[U](s) = \max_a \underbrace{\left( R(s, a) + \gamma \sum_{s'} T(s' | s, a) U(s') \right)}_{Q(s, a)}$$

$i$  = index of  $s$ ;  $j$  = index of  $s'$

Naive implementation:

$$U'[i] = \max_a \underbrace{\left( R[a][i] + \gamma \sum_j T[a][i, j] U[j] \right)}_{Q_a[i]}$$

$$Q_a = R[a] + \gamma T[a] U$$

$$y = Mx$$

$$y[i] = \sum_j M[i, j] x[j]$$

$$\begin{bmatrix} y \\ \vdots \end{bmatrix} = \begin{bmatrix} M & \\ & I \end{bmatrix} \begin{bmatrix} x \\ \vdots \end{bmatrix}$$

Diagram illustrating matrix multiplication:  $y = Mx$ . On the left, a column vector  $y$  is shown. An equals sign follows. To the right is a block matrix multiplication:  $y$  is multiplied by a block matrix where the top-left block is labeled  $M$  and the bottom-right block is labeled  $I$ . The result is a column vector  $x$ .