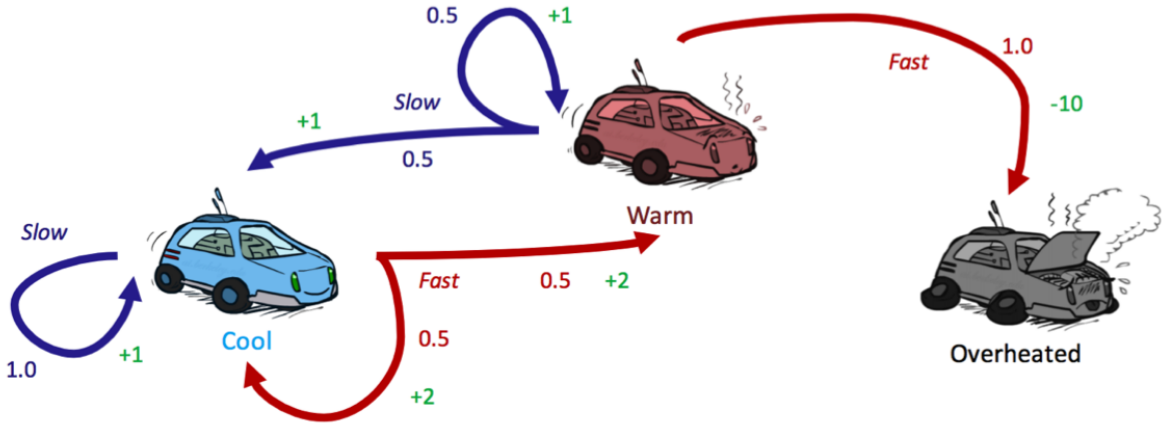
From CS188, UC Berkeley

Consider the motivating example of a racecar:



There are three possible states, $S = \{cool, warm, overheated\}$, and two possible actions $A = \{slow, fast\}$. Just like in a state-space graph, each of the three states is represented by a node, with edges representing actions. *Overheated* is a terminal state, since once a racecar agent arrives at this state, it can no longer perform any actions for further rewards (it's a *sink state* in the MDP and has no outgoing edges). Notably, for nondeterministic actions, there are multiple edges representing the same action from the same state with differing successor states. Each edge is annotated not only with the action it represents, but also a transition probability and corresponding reward. These are summarized below:

- **Transition Function**: $T(s,a,s')$
  - $T(cool, slow, cool) = 1$
  - $T(warm, slow, cool) = 0.5$
  - $T(warm, slow, warm) = 0.5$
  - $T(cool, fast, cool) = 0.5$
  - $T(cool, fast, warm) = 0.5$
  - $T(warm, fast, overheated) = 1$
- **Reward Function**: $R(s,a,s')$
  - $R(cool, slow, cool) = 1$
  - $R(warm, slow, cool) = 1$
  - $R(warm, slow, warm) = 1$
  - $R(cool, fast, cool) = 2$
  - $R(cool, fast, warm) = 2$
  - $R(warm, fast, overheated) = -10$

Let's see a few updates of value iteration in practice by revisiting our racecar MDP from earlier, introducing a discount factor of $\gamma = 0.5$:

	cool	**warm**	**overheated**
V_0	0	0	0
V_1			
V_2			

Recall that our ultimate goal in solving a MDP is to determine an optimal policy. This can be done once all optimal values for states are determined using a method called **policy extraction**. The intuition behind policy extraction is very simple: if you're in a state s, you should take the action a which yields the maximum expected utility. Not surprisingly, a is the action which takes us to the q-state with maximum q-value, allowing for a formal definition of the optimal policy:

$$\forall s \in S,\ \pi^*(s) = \underset{a}{\text{argmax}}\, Q^*(s,a) = \underset{a}{\text{argmax}} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

It's useful to keep in mind for performance reasons that it's better for policy extraction to have the optimal q-values of states, in which case a single argmax operation is all that is required to determine the optimal action from a state.

	cool	**warm**
π_0		
π_1		
π_2		

Because terminal states have no outgoing actions, no policy can assign a value to one.