# Making Complex Decisions

**Outline**

- ➤ Sequential decision problems
- ➤ Value iteration
- ➤ Policy iteration
- ➤ Decision-theoretic agent design

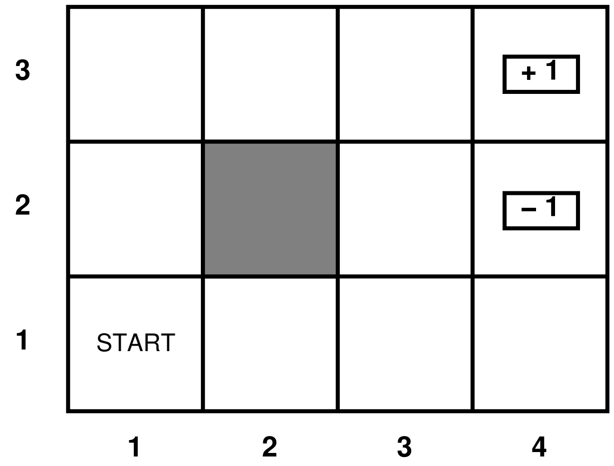


**Example.** Sequence of unreliable actions

- ➤ Available actions: North, South, East, West
- ➤ Transition model: $M_{ij}^{a}$ the probability of reaching state $j$ if action $a$ is done in state $i$.

  Unreliable actions: 0.8/0.1/0.1
- ➤ Utility function on sequence of states:
  terminal state's value minus 1/25th the length of the sequence.
- ➤ Example: location (3,2)/action sequence [North,East] reaches (4,3) with probability 0.64.

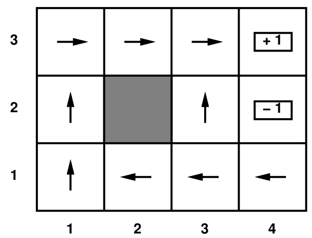







## Sequential Decision Problems

- ➤ A sequence as an action: committing before executing
- ➤ Policy: mapping from states to actions
- ➤ Markov decision problem:
  find an optimal policy (highest expected utility) in an accessible environment (percepts identify the state).
- ➤ Markov property: transition probabilities depend only on the state (not on the history).
- ➤ Solving MDP: value iteration, policy iteration
- ➤ Inaccessible environment: partially observable MDP




An optimal policy

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 3 | 0.812 | 0.868 | 0.912 | +1 |
| 2 | 0.762 | | 0.660 | −1 |
| 1 | 0.705 | 0.655 | 0.611 | 0.388 |

The utilities of the states



## Value Iteration

- Idea: compute first utility for each state and then select optimal actions.
- Assume additive utility function: $U_h([s_0, s_1, \ldots, s_n]) = R(s_0) + U_h([s_1, \ldots, s_n])$
- optimal action: $policy^*(i) = \arg \max_a \sum_j M_{ij}^a U(j)$
- Utility: $U(i) = R(i) + \max_a \sum_j M_{ij}^a U(j)$
- Dynamic programming (n steps)



### Value iteration algorithm

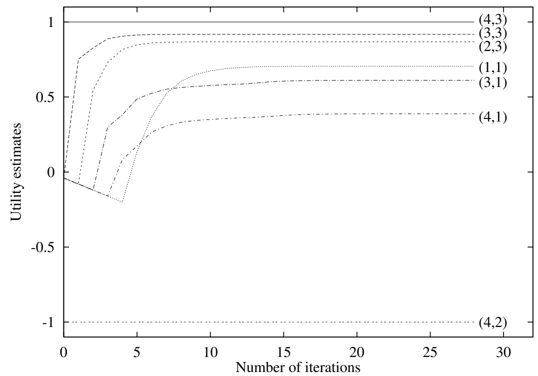
(No bound on the sequences is assumed)

**function** VALUE-ITERATION($M$, $R$) **returns** a utility function
  **inputs**: $M$, a transition model
    $R$, a reward function on states
  **local variables**: $U$, utility function, initially identical to $R$
    $U'$, utility function, initially identical to $R$

  **repeat**
    $U \leftarrow U'$
    **for each** state $i$ **do**
      $U'[i] \leftarrow R[i] + \max_a \sum_j M_{ij}^a U[j]$
    **end**
  **until** CLOSE-ENOUGH($U$, $U'$)
  **return** $U$



### Convergence



## Policy Iteration

- Idea: choose a policy, calculate utilities given the policy, update policy.
- Value determination (utility values) simpler when given a policy.

  $$U_{t+1}(i) \leftarrow R(i) + \sum_j M_{ij}^{Policy(i)} U_t(j)$$

  Convergence?
- Values from a set of equations:

  $$u_{(1,1)} = r_{(1,1)} + 0.8u_{(1,2)} + 0.1u_{(1,1)} + 0.1u_{(2,1)}$$



**function** POLICY-ITERATION($M$, $R$) **returns** a policy
  **inputs**: $M$, a transition model
    $R$, a reward function on states
  **local variables**: $U$, a utility function, initially identical to $R$
    $P$, a policy, initially optimal with respect to $U$

  **repeat**
    $U \leftarrow$ VALUE-DETERMINATION($P, U, M, R$)
    *unchanged?* $\leftarrow$ true
    **for each** state $i$ **do**
      **if** $\max_a \sum_j M_{ij}^a U[j] > \sum_j M_{ij}^{P[i]} U[j]$ **then**
        $P[i] \leftarrow \arg\max_a \sum_j M_{ij}^a U[j]$
        *unchanged?* $\leftarrow$ false
    **end**
  **until** *unchanged?*
  **return** $P$



## Decision-Theoretic Agent Design

### Decision cycle

**function** DECISION-THEORETIC-AGENT( *percept*) **returns** *action*

  calculate updated probabilities for current state based on
    available evidence including current percept and previous action
  calculate outcome probabilities for actions
    given action descriptions and probabilities of current states
  select *action* with highest expected utility
    given probabilities of outcomes and utility information
  **return** *action*



### Updating current state estimate

- State variables: $\mathbf{X}_t$ indexed by time $t$
- $Bel(\mathbf{X}_t)$: $\mathbf{P}(\mathbf{X}_t|\mathbf{E}_1, \ldots, \mathbf{E}_t, A_1, \ldots, A_{t-1})$
- Assumptions:

  $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_1, \ldots, \mathbf{X}_{t-1}, A_1, \ldots, A_{t-1}) =$
  $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1}, A_{t-1})$ (Markovian)

  $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_1, \ldots, \mathbf{X}_t, A_1, \ldots, A_{t-1}, \mathbf{E}_1, \ldots, \mathbf{E}_{t-1}) =$
  $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$

  $\mathbf{P}(A_{t-1}|A_1, \ldots, A_{t-2}, \mathbf{E}_1, \ldots, \mathbf{E}_{t-1}) =$
  $\mathbf{P}(A_{t-1}|\mathbf{E}_1, \ldots, \mathbf{E}_{t-1})$

**Computing** $Bel(\mathbf{X}_t)$: (Generaliztion of Kalman filtering)

Prediction phase: $\widehat{Bel}(\mathbf{X}_t) =$

$$\sum_{\mathbf{x}_{t-1}} \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{x}_{t-1}, A_{t-1}) Bel(\mathbf{X}_{t-1} = \mathbf{x}_{t-1})$$

Estimation phase: $Bel(\mathbf{X}_t) = \alpha \mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)\widehat{Bel}(\mathbf{X}_t)$
where $\alpha$ is a normalization constant.

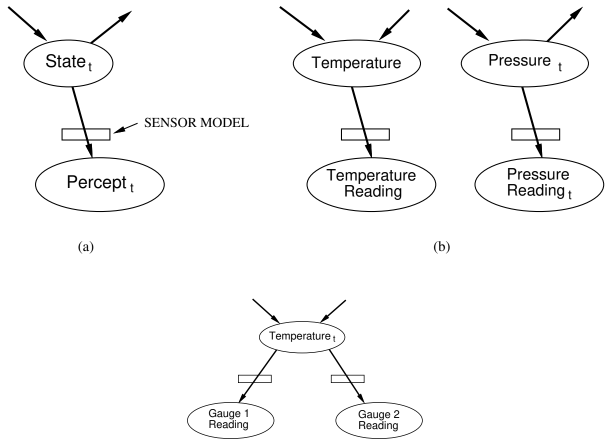
- action model: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1}, A_{t-1})$
- sensor model: $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$
  usually stationary: $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t) = \mathbf{P}(\mathbf{E}|\mathbf{X})$
  Noise and errors: probabilities of incorrect readings
  sensor/data fusion: multiple sensors for a variable



**function** DECISION-THEORETIC-AGENT($E_t$) **returns** an action
**inputs**: $E_t$, the percept at time $t$
**static**: $BN$, a belief network with nodes **X**
$Bel(\mathbf{X})$, a vector of probabilities, updated over time

$\widehat{Bel}(\mathbf{X}_t) \leftarrow \sum_{\mathbf{X}_{t-1}} \mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1}=\mathbf{x}_{t-1}, A_{t-1})\ Bel(\mathbf{X}_{t-1}=\mathbf{x}_{t-1})$
$Bel(\mathbf{X}_t) \leftarrow \alpha\ \mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_t)\ \widehat{Bel}(\mathbf{X}_t)$
$action \leftarrow \arg\max_{A_t} \sum_{\mathbf{X}_t} \left[ Bel(\mathbf{X}_t=\mathbf{x}_t) \sum_{\mathbf{X}_{t+1}} P(\mathbf{X}_{t+1}=\mathbf{x}_{t+1} \mid \mathbf{X}_t=\mathbf{x}_t, A_t)\ U(\mathbf{x}_{t+1}) \right]$
**return** *action*

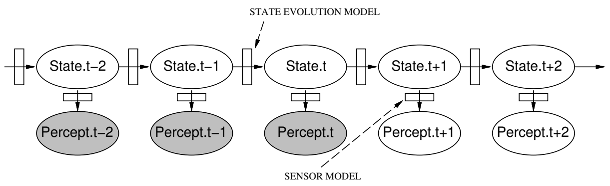







## Dynamic Belief Networks

- Representing the evolution of the state
- Stationarity assumption:
  $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1}, A_{t-1})$ same for all $t$.
- Assumption: the agent is passively monitoring and predicting a changing environment (no actions).
- Markov chain: sequence of $\mathbf{X}_t$ values determined by $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$
  ☞ Dynamic Belief Network (DBN)



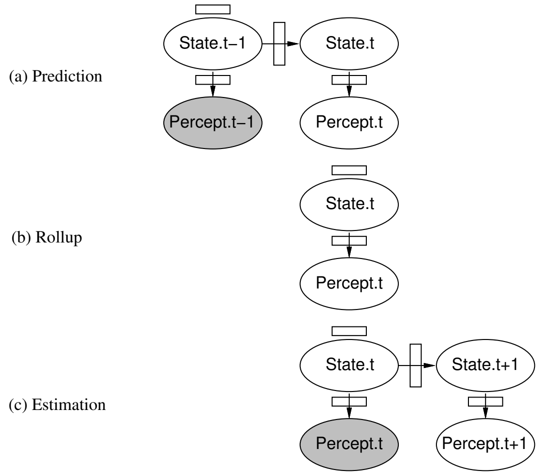



**Prediction-estimation process with belief networks**

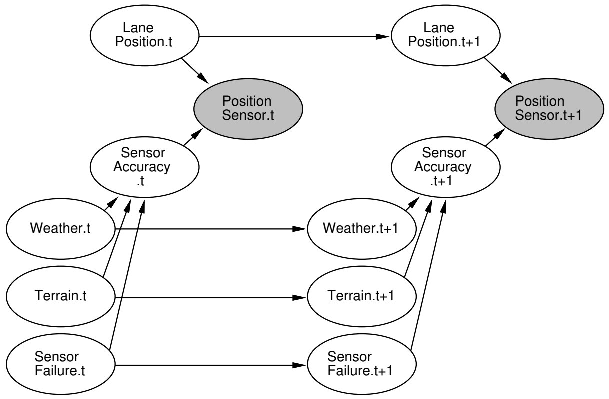
Two time steps are enough

- Prediction: we have $Bel(\mathbf{X}_{t-1})$, evidence up to $\mathbf{E}_{t-1}$; calculate $\widehat{Bel}(\mathbf{X}_t)$
- Rollup: remove slice $t-1$ (add priors $\widehat{Bel}(\mathbf{X}_t)$).
- Estimation: add $\mathbf{E}_t$, calculate $Bel(\mathbf{X}_t)$ and then add slice $t+1$.







# Dynamic Decision Networks

- DDN: DBN + utility nodes + decision nodes
- Utility of a decision sequence: weighted sum of utilities using the possible percept sequences
  weight: probability of the percept sequence given the decision sequence.