Markov Decision Processes

Wha Sook Jeon

Mobile Computing & Communications Lab.

Seoul National University

Contents

Markov Process

- Stochastic Process (Ransom Process)
  - Collection of random variables, X_t , which represents the system state at time t: $\{X_t, t = 0, 1, \dots\}$
  - Describes the evolution through time of physical precess
- Makovian Property

$$\Pr(X_{t+1} = s_k \mid X_0 = s_0 , X_1 = s_1, \dots, X_t = s_i)$$
$$= \Pr(X_{t+1} = s_k \mid X_t = s_i) = p_{ik}$$

- Markov process
  - Stochastic process having Markovian property
  - defined as a tuple $(\boldsymbol{S}, \boldsymbol{P})$
    - $\boldsymbol{S}$: Set of feasible states
    - $\boldsymbol{P}$: State transition matrix $[p_{ik}]$
  - is used to evaluate the system performance

γ

Markov processes with rewards

- A MP with rewards is a tuple ($\boldsymbol{S}$, $\boldsymbol{P}$, $\boldsymbol{R}$)
  - State space: $\boldsymbol{S} = \{s_1, s_2, \ldots, s_N\}$
  - Transition probability matrix: P

    $$P_{ij} = \Pr(X_{t+1} = s_j \mid X_t = s_i)$$

  - Reward: $\boldsymbol{R} = (r_1, r_2, \ldots, r_N)$
    - Each state s_i has a reward r_i

γ

Discounted Rewards

- A reward in the future is not worth as much as a reward now.
- Discounting factor: γ
- Expected discounted sum of future rewards

$$\sum_{t=0}^{\infty} \gamma^t R_t$$

  - R_t: reward in time t
  - R_0: immediate (now) reward

γ

Expected Reward Sum: $J^*(s_i)$

- $J^*(s_i)$: the expected discounted sum of future rewards, starting in state s_i

$$J^*(s_i) = r_i + \gamma \sum_{j=1}^{N} P_{ij} J^*(s_j)$$

- Matrix Inversion for solving $J^*(s_i)$
  - Using the vector (matrix) notation

$$J^* = \begin{bmatrix} J^*(s_1) \\ J^*(s_2) \\ \vdots \\ J^*(s_N) \end{bmatrix} \qquad R = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_3 \end{bmatrix} \qquad P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix}$$

$$\boldsymbol{J^*} = \boldsymbol{R} + \gamma \boldsymbol{PJ^*}$$

- Then, solve $\boldsymbol{J^*}$ using the matrix inversion

Value Iteration for solving $J^*(s_i)$ (1)

- $J^k(s_i)$: the expected discounted sum of rewards during next k steps, starting at s_i

$$J^0(s_i) \leftarrow r_i$$

$$J^1(s_i) \leftarrow r_i + \gamma \sum_{j=1}^{N} p_{ij} J^0(s_i)$$

$$\vdots$$

$$J^k(s_i) \leftarrow r_i + \gamma \sum_{j=1}^{N} p_{ij} J^{k-1}(s_j)$$

$$\vdots$$

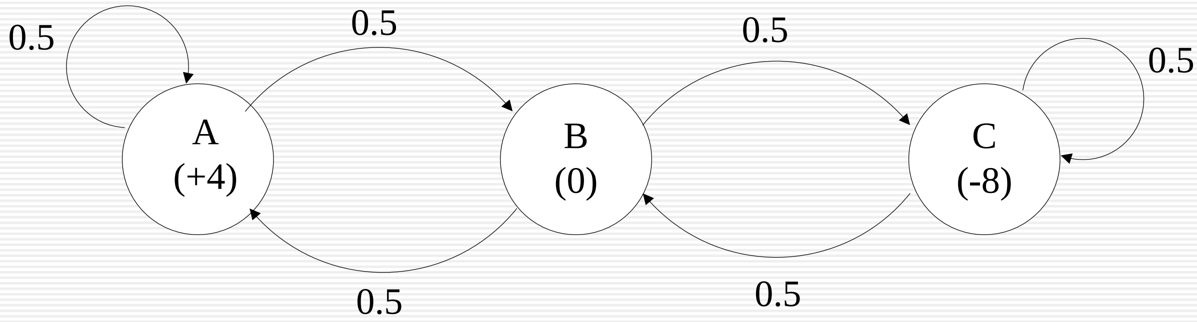
$$\lim_{k \to \infty} J^k(s_i) = J^*(s_i)$$

$$\Downarrow$$

$$\text{until } \underset{\text{i}}{\text{Max}} \left| J^{k+1}(s_i) - J^k(s_i) \right| < \zeta$$

Value Iteration for solving $J^*(s_i)$ (2)

- Example



$\gamma = 0.5$

k	J^k(A)	J^k(B)	J^k(C)
0	4	0	-8
1	5	-1	-10
2	5	-1.25	-10.75
3	4.94	-1.44	-11

Markov Decision Process

Markov Decision Process (1)

- MDPs provide a mathematical framework for modeling decision-making
  - in situation where outcomes are partly random and partly under the control of the decision maker
- MDPs are useful for studying a wide range of optimization problems via dynamic programming
- A variety of areas including robotics, automated control, economics, etc.

Markov Decision Process (2)

- A discrete time stochastic control process
- Markov chain with rewards and actions
- defined as a tuple (***S***, ***A***, ***P***, ***R***)
  - State space: ***S*** = $\{s_1, s_2, \ldots, s_N\}$
  - Action space: **A**
  - Transition probability matrix: P

    $$P_a(i,j) = \Pr(X_{t+1} = s_j \mid X_t = s_i, a_t = a)$$
  - Reward: ***R*** = $(R(s_1), R(s_2), \ldots, R(s_N))$
- A policy is a mapping from states to actions
- What's an optimal policy?

Finding the optimal policy: Value Iteration (1)

- Computing the optimal value function using value iteration.
- Optimal policy is the actions for the optimal value function
- Optimal Value function: $J^*(s_i)$
  - the expected discounted sum of future rewards, starting at state s_i, when the optimal policy is assumed to be used.
- Computing the optimal value function
  - $J^k(s_i)$: the maximum possible expected discounted sum of rewards we can get, after k time steps starting at s_i
  - $\lim_{k \to \infty} J^k(s_i) = J^*(s_i)$

Value Iteration (2)

- Bellman's Equation

$$J^k(s_i) = \max_a [r_i + \gamma \sum_{j=1}^{N} p_a(i,j)\, J^{k-1}(s_i)]$$

- Using the dynamic programming

$$k = 0$$

$$J^0(s_i) \leftarrow r_i \quad \text{for all } s_i$$

repeat

$$k \leftarrow k+1$$

$$J^k(s_i) \leftarrow \max_a [r_i + \gamma \sum_{j=1}^{N} p_a(i,j)\, J^{k-1}(s_j)] \quad \text{for all } s_i$$

$$\text{until } (\max_{\mathrm{i}} \left| J^k(s_i) - J^{k-1}(s_i) \right| < \zeta)$$

Value Iteration (3)

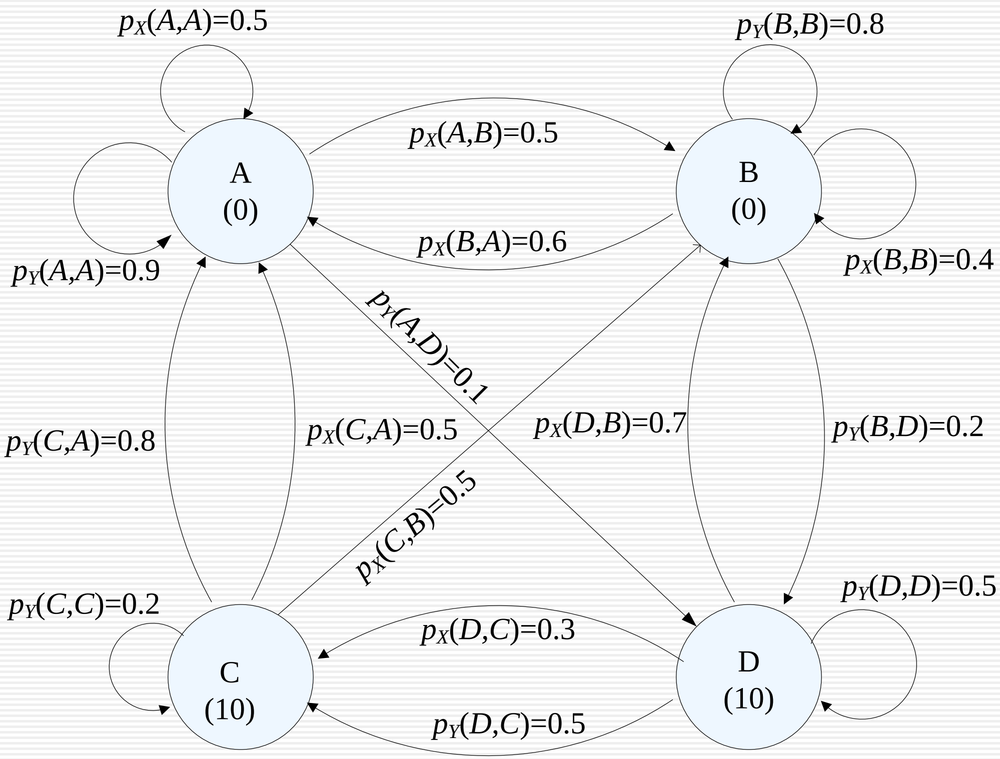
- Finding the optimal policy
  - Compute $J^*(s_i)$ for all s_i
  - Then, we can obtain the best action in state s_i

⇨ Optimal policy

$$\arg\max_a [r_i + \gamma \sum_{j=1}^{N} p_a(i,j) J^*(s_j)]$$

Value Iteration (4)

- Example



k	$J^k(A)$	$J^k(B)$	$J^k(C)$	$J^k(D)$
1	0	0	10	10
2	0.9	1.8	11.8	19
3	2.439			
4				
5				
6				

Action set = {X, Y} γ = 0.9

Partially Observable MDP

POMDP (1)

- defined as a six-tuple (***S***, ***A***, ***P***, ***O***, ***Q***, ***R***)
- Core process
  - A finite state Markov chain $\{X_t, t \in I\}$, where $I = \{0, 1, ...\}$.
    - State space: $S = \{1, 2, \ ..., N\}$
    - Transition probability matrix: $p_{ij} = \Pr\{X_{t+1} = j \mid X_t = i\}$
  - cannot be directly observable
- Observation process: $\{Y_t, t \in I\}$, where $I = \{0, 1, ...\}$.
  - By observing Y_t at time *t*, information regarding the true value of X_t is obtained
- The probabilistic relationship between the core process and observation process when action *a* is chosen: $q_{ij}(a) = \Pr\{Y_t = j \mid X_t = i, \ a_{t-1} = a\}$

POMDP (2)

- Random variables
  - m_t : the observable value of Y_t

    a_t : the action taken at time t

    d_t : the data available for decision making at time t

    $$d_t = (\pi(0), m_1, a_1, m_2, a_2, \cdots, a_{t-1}, m_t)$$
- Information vector: $\pi(t) = (\pi_1(t), \pi_2(t), \cdots, \pi_N(t))$
  - $\pi_i(t) = \Pr\{X_t = i \mid d_t\}$
- Transformation of information vector
  - $$\begin{aligned} \pi_i(t+1) &= \Pr\{X_{t+1} = i \mid d_{t+1} = (d_t, a_t, m_{t+1} = j)\} \\ &= \mathrm{T}_i\left[\pi(t), a_t, j\right] \\ &= \frac{q_{ij}(a_t) \sum_{k \in S} \pi_k(t) p_{ki}(a_t)}{\sum_{l \in S} q_{lj}(a_t) \sum_{k \in S} \pi_k(t) p_{kl}(a_t)} \end{aligned}$$

POMDP (3)

- Immediate reward:
  - $$r_i(a_t) = \sum_{j \in S} \sum_{k \in \Theta} R(i, j, k, a) p_{ij}(a) q_{jk}(a)$$
    - $R(i, j, k, a)$: immediate reward when action *a* is taken, the core process is in state *i*, moves to state *j*, and observation is *k*
- Value function
  - $$V_\beta^n(\pi) = \max_{a \in A} \left\{ \pi \cdot r(a) + \beta \sum_{j \in S} V_\beta^{n-1}\left(T\left[\pi, j, a\right]\right) \eta(j \mid \pi, a) \right\}$$
    - $\eta(j \mid \pi, a) = \Pr\{Y_{t+1} = j \mid \pi(t), a_t = a\}$

An Example of POMDP

A POMDP-based
Cognitive Radio Senor Networks

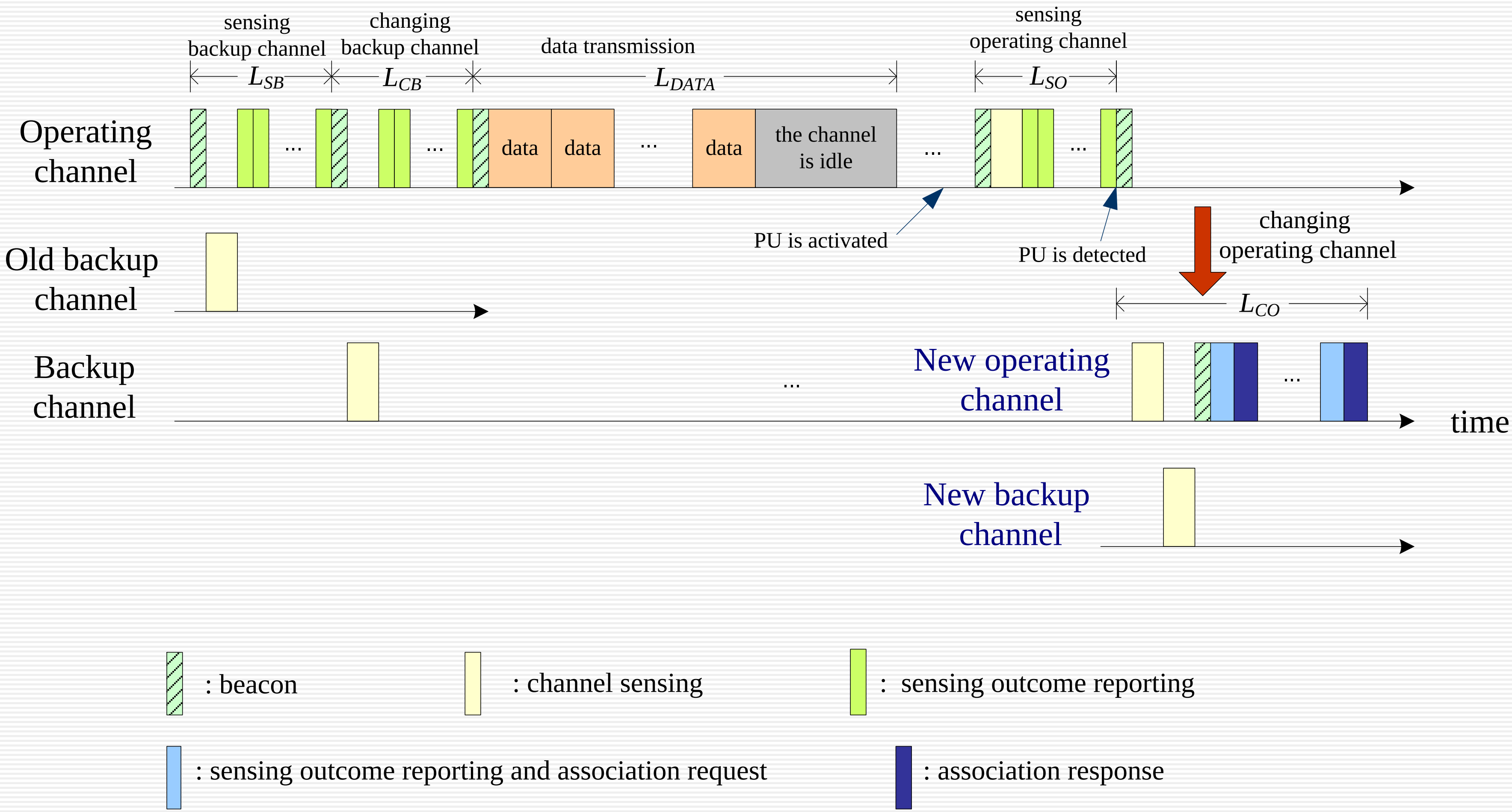
System Description

- Channel pool
  - each channel is licensed to a primary user (PU)
- CR sensor network
  - Cluster with star topology
    - One cluster header (CH) and (N-1) cluster members (CMs)
  - The sensor nodes (CM) opportunistically access to a vacant channel under the control of CH
    - CRSN control of CH: POMDP-based Decision
  - One operating channel and one backup channel

Operation Modes of CRSN

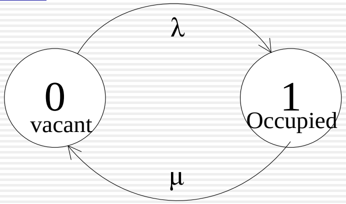
- *DATA* mode
  - The sensor nodes transmit data to CH
    - according to the transmission schedule given by CH
- *SO* mode
  - Sense the operating channel and report the sensing result to CH
- *SB* mode
  - Sense the backup channel and report the sensing result to CH
- *CO* mode
  - Switch to the backup channel (new operating channel)
  - Sense the new operating channel and the randomly selected backup channel
  - Report the sensing result and send new association message
- *CB* mode
  - CH randomly chooses new backup channel
  - All sensor nodes sense the backup channel and report their results

An Example Scenario



Sensing Model

- PU activation model on a channel:



- Operating channel: channel 1

  Backup channel: channel 2
- Sensing model: Energy detection
  - Each cluster member reports the received energy to CH
  - $s_t^{(m)}$: Sum of the sensing results on the channel *m* at decision epoch *t*
    - *Chi-square* distribution
    - Quantize $s_t^{(m)}$ into *K* levels with thresholds $\gamma_0, \gamma_1, \cdots, \gamma_K$
  - Probability that the quantized value (observation value) is *k*
    - H_0: Channel *m* is empty
      - $v_0(k) \equiv \Pr\{\gamma_{k-1} < s_t^{(m)} < \gamma_k \mid H_0\}$
    - H_1: A PU exists on channel *m*
      - $v_1(k) \equiv \Pr\{\gamma_{k-1} < s_t^{(m)} < \gamma_k \mid H_1\}$

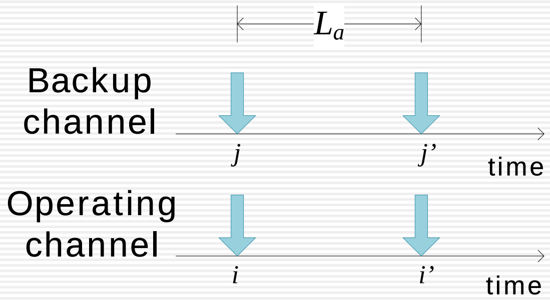
POMDP Model (1)

- A six-tuple (***X***, ***A***, ***O***, ***P***, ***Q***, ***R***)
  - ***X***: State space of the core process
    - $X_t = (x_t^{(1)}, x_t^{(2)})$
      - $x_t^{(i)} = 0$ (vacant) or 1 (occupied)
  - ***A***: Action space
    - $\{DATA,\ SO,\ SB,\ CO,\ CB\}$
  - ***O***: State space of the observation process
    - $O_t = (o_t^{(1)}, o_t^{(2)})$
      - If the channel m is sensed, $o_t^{(m)} \in \{1, \ldots, K\}$
      - Otherwise, $o_t^{(m)} = 0$
  - ***P***: Transition probability matrix

$$p_{(i,j)(i',j')}(a) = \Pr\{X_{t+1} = (i', j') \mid X_t = (i, j), A_t = a\}$$

POMDP Model (2)

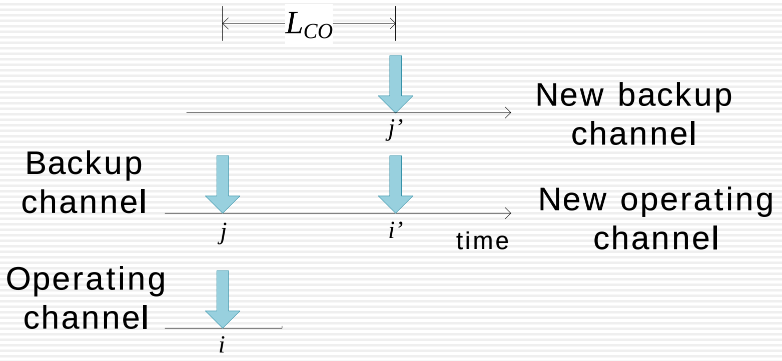
- Transition probability of core process
  - *DATA*, *SO*, *SB*
    - $p_{(i,j)(i',j')}(a) = u_{i,i'}(a) \times u_{j,j'}(a)$



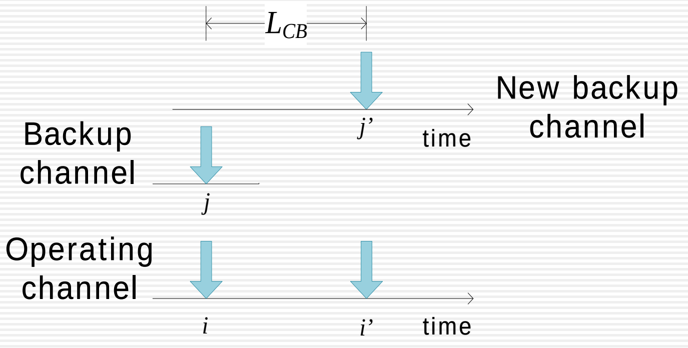
- $u_{i,i'}(a)$: prob. that a channel transits from state i to i'
- $w_i(a)$: prob. that a randomly selected channel is in state i

POMDP Model (3)

- ***P***: Transition probability of core process
  - *CO*: $p_{(i,j)(i',j')}(a) = u_{j,i'}(a) \times w_{j'}(a)$



  - *CB*: $p_{(i,j)(i',j')}(a) = u_{i,i'}(a) \times w_{j'}(a)$



POMDP Model (4)

- ***Q***: Probabilistic relation between ***X*** and ***O***
  - $q_{(i,j)(k,l)}(a) = \Pr\{O_t = (k,l) \mid X_t = (i,j), A_{t-1} = a\}$

Action a	$q_{(i,j)(k,l)}(a)$
DATA	$n(k) \times n(l)$
SO	$v_i(k) \times n(l)$
SB	$n(k) \times v_j(l)$
CO	$v_i(k) \times v_j(l)$
CB	$n(k) \times v_j(l)$

- $v_i(k)$: Prob. that $o_t^{(m)} = k$ when the channel m is sensed
- $n(k)$: Prob. that $o_t^{(m)} = k$ when the channel m is not sensed

POMDP Model (5)

- ***R***: Rewards
  - Control parameter for getting the required performance
    - Penalty on unnecessary energy consumption
      - ex) CO/CB by false alarm
    - Positive reward on protecting PU
  - $r_{(i,j)(i',j')}(a)$: reward by taking the action *a* in state (*i*,*j*) which results in the transition (*i'*,*j'*)
    - Example

$$r_{(1,\cdot)(\cdot,\cdot)}(DATA) = -10, \qquad r_{(0,\cdot)(\cdot,\cdot)}(DATA) = 10$$

$$r_{(0,\cdot)(\cdot,\cdot)}(SO) = r_{(\cdot,0)(\cdot,\cdot)}(SB) = -1$$

  - $R_{(i,j)}(a) = \sum_{i'=0}^{1}\sum_{j'=0}^{1} r_{(i,j)(i',j')}(a)\, p_{(i,j)(i',j')}(a)$

Decision Making (1)

- Information vector: $\Pi(t) = (\pi_{(0,0)}(t),\ \pi_{(0,1)}(t),\ \pi_{(1,0)}(t),\ \pi_{(1,1)}(t))$
  - $\pi_{(i,j)}(t)$: Prob. that the core process is in state (i,j) at decision epoch t.
  - summarizes all information required for the decision-making
  - Update of information vector
    - $$\begin{aligned} \pi_{(i,j)}(t+1) &\equiv T_{(i,j)}(\Pi(t), a, (k,l)) \\ &= \Pr\{X_{t+1} = (i,j) \mid \Pi(t), A_t = a, O_{t+1} = (k,l)\} \\ &= \frac{q_{(i,j)(k,l)}(a) \sum_{i'=0}^{1} \sum_{j'=0}^{1} p_{(i',j')(i,j)}(a) \pi_{(i',j')}(t)}{\sum_{\tilde{i}=0}^{1} \sum_{\tilde{j}=0}^{1} q_{(\tilde{i},\tilde{j})(k,l)}(a) \sum_{i'=0}^{1} \sum_{j'=0}^{1} p_{(i',j')(\tilde{i},\tilde{j})}(a) \pi_{(i',j')}(t)} \end{aligned}$$

Decision Making (2)

- Optimal value function
  - $$V^*(\Pi) = \max_{a \in \mathbf{A}} \left(\sum_{i=0}^{1} \sum_{j=0}^{1} \pi_{(i,j)} R_{(i,j)}(a) + \beta \sum_{k=0}^{K} \sum_{l=0}^{K} V^*(T(\Pi, a, (k,l)) \times \Pr\{(k,l) \mid \Pi, a\} \right)$$
- Optimal policy
  - $$\delta^*(\Pi) = \arg\max_{a \in \mathbf{A}} \left(\sum_{i=0}^{1} \sum_{j=0}^{1} \pi_{(i,j)} R_{(i,j)}(a) + \beta \sum_{k=0}^{K} \sum_{l=0}^{K} V^*(T(\Pi, a, (k,l)) \times \Pr\{(k,l) \mid \Pi, a\} \right)$$

Conclusion

- The solution to a MDP is an optimal policy, which gives the action to take for a given state
- When the action is fixed to each state, the resulting MDP behaves like a Markov process
- A POMDP is a generation of a MDP which permits uncertainty