

Graphical Probability Models for Inference and Decision Making

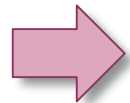
Unit 8: Graphical Models for Planning and Decision Making

Instructor: Kathryn Blackmond Laskey
Spring 2012

Learning Objectives

- Construct an influence diagram or decision tree to model a decision problem
- Construct a decision model for a decision involving whether to gather information
- Construct a collection of BN fragments to represent knowledge for a planning problem
 - Knowledge about goals of planning agent
 - Knowledge about how the state of the world evolves, including uncertainty and persistence of world states
 - Knowledge about how actions influence the state of the world
- Describe some approaches to selecting action policies for a planner

Unit 8 Outline



- Modeling decision problems with decision graphs
- Planning and temporal models

The Elements of a Decision

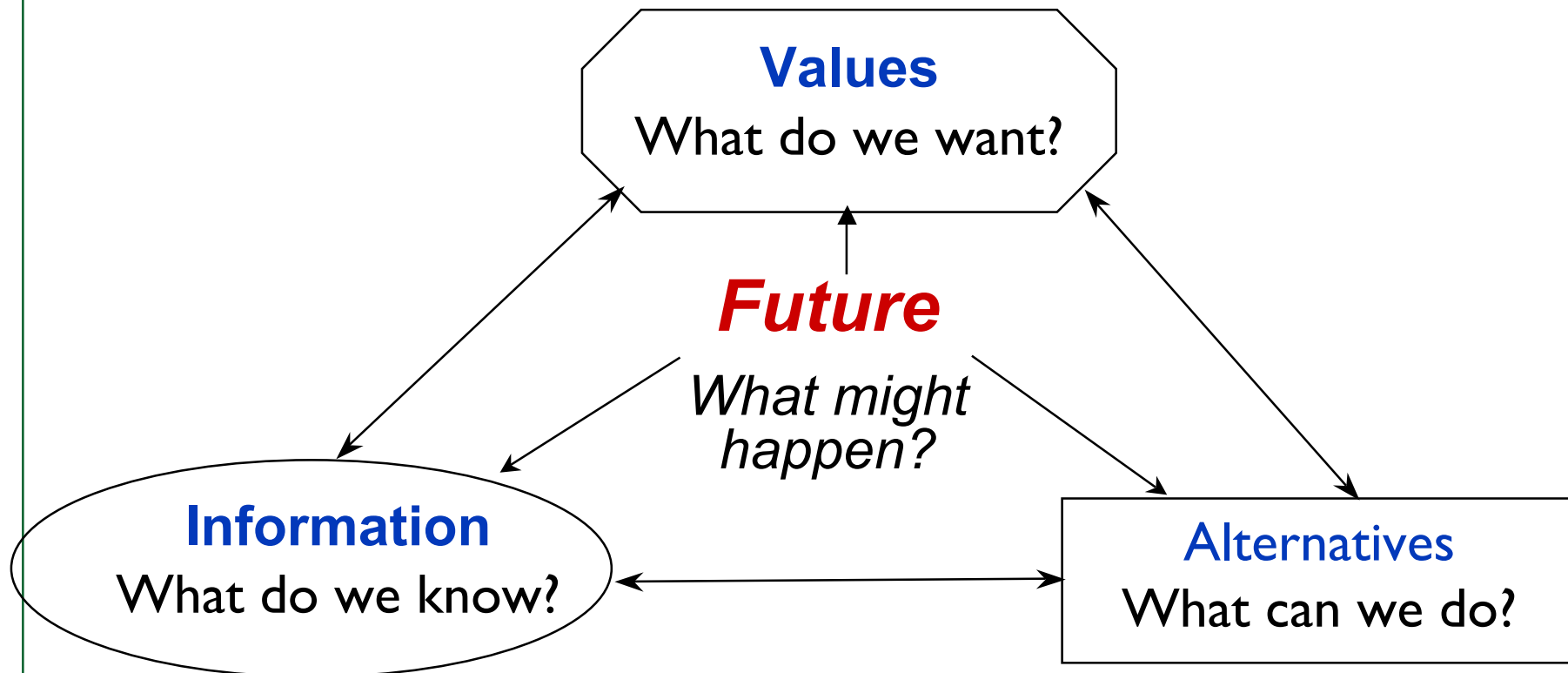
- Objectives
- Alternatives to choose between
- Uncertainty in Events and Outcomes
- Consequences of the Decision



What makes a decision hard to make?

- Competing objectives
- Uncertainty
- Politics
- Lack of information
- Differing perspectives of Decision makers
- Disagreement over what is to be accomplished
- Complexity

Critical Components



Understanding the future is critical to all three elements. Use strategic values to identify future decision opportunities.

Decision Theory

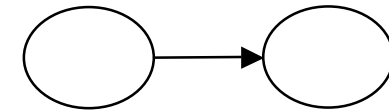
- Elements of a decision problem
 - Possible actions: $\{a\}a \in A$
 - States of the world (usually uncertain): $\{s\}s \in S$
 - Possible consequences: $\{c\}c \in C$
 - » Consequences $c(s,a)$ are functions of states and actions
- Question: What is the best action?
- Ingredients of a decision theoretic model
 - Utility function $u(c)$ expresses preference for consequences
 - Probability $p(s)$ expresses knowledge/uncertainty about state of world
 - Best action maximizes expected utility: $a^* = \arg \max_a \{E[u(c) | a]\}$
- Decision maker may need to trade off
 - Different dimensions of value
 - Value now against value in future
 - Value for self against value for others
- Outcomes may be uncertain

Influence Diagrams / Decision Graphs

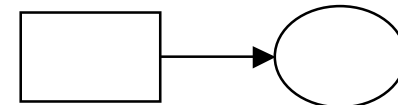
- An influence diagram (aka decision graph) is a DAG that depicts relationships among variables in a decision problem.
- An influence diagram has 3 types of node:
 - Aleatory or chance nodes
 - » Chance nodes are drawn as circles or ovals. (Often a double circle represents a chance node which is a deterministic function of other chance variables -- “deterministic chance node”)
 - » Chance nodes represent uncertain variables impacting the decision problem
 - Decision nodes
 - » Decision nodes are drawn as boxes
 - » Decision nodes represent choices open to the decision maker
 - Value nodes
 - » Value nodes are drawn as triangles, hexagons or rounded boxes (depending on the author and/or the computer draw package)
 - » Value nodes represent attributes the decision maker cares about

Types of Influence

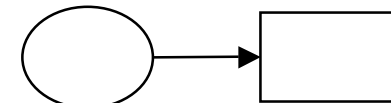
- Arcs into chance nodes from other chance nodes are called *relevance arcs*. A relevance arc indicates that the value of one variable is relevant to the probability distribution of the other variable.
- Arcs from decision nodes into chance nodes are called *influence arcs*. An influence arc means that the decision affects, or influences, the outcome of the chance node.
- Arcs into decision nodes are called *information arcs*. An information arc means that the quantity will be known at the time the decision is made.
 - Decision nodes are ordered in time
 - In standard influence diagrams, decision node and all its information predecessors are (implicit) information predecessors to all future decision nodes (no forgetting)
- Arcs from chance or decision nodes into value nodes represent *functional links*. Value nodes may not be parents to decision or chance nodes.



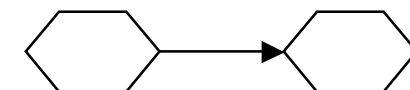
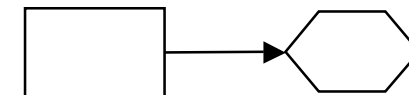
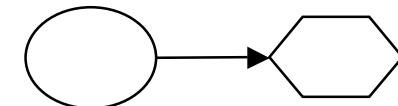
Relevance Arc



Influence Arc



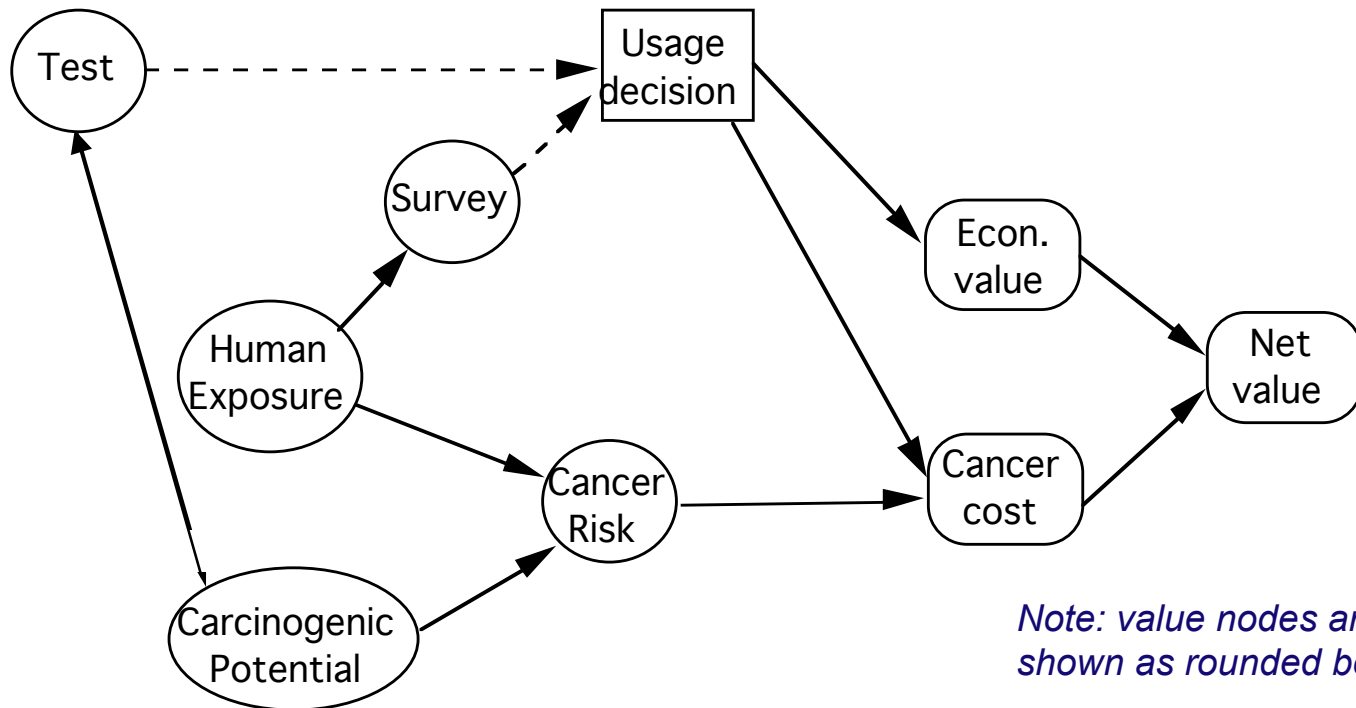
Information Arc



Value Arcs

Notation for value nodes varies. Some packages use rounded boxes, others diamonds, others hexagons.

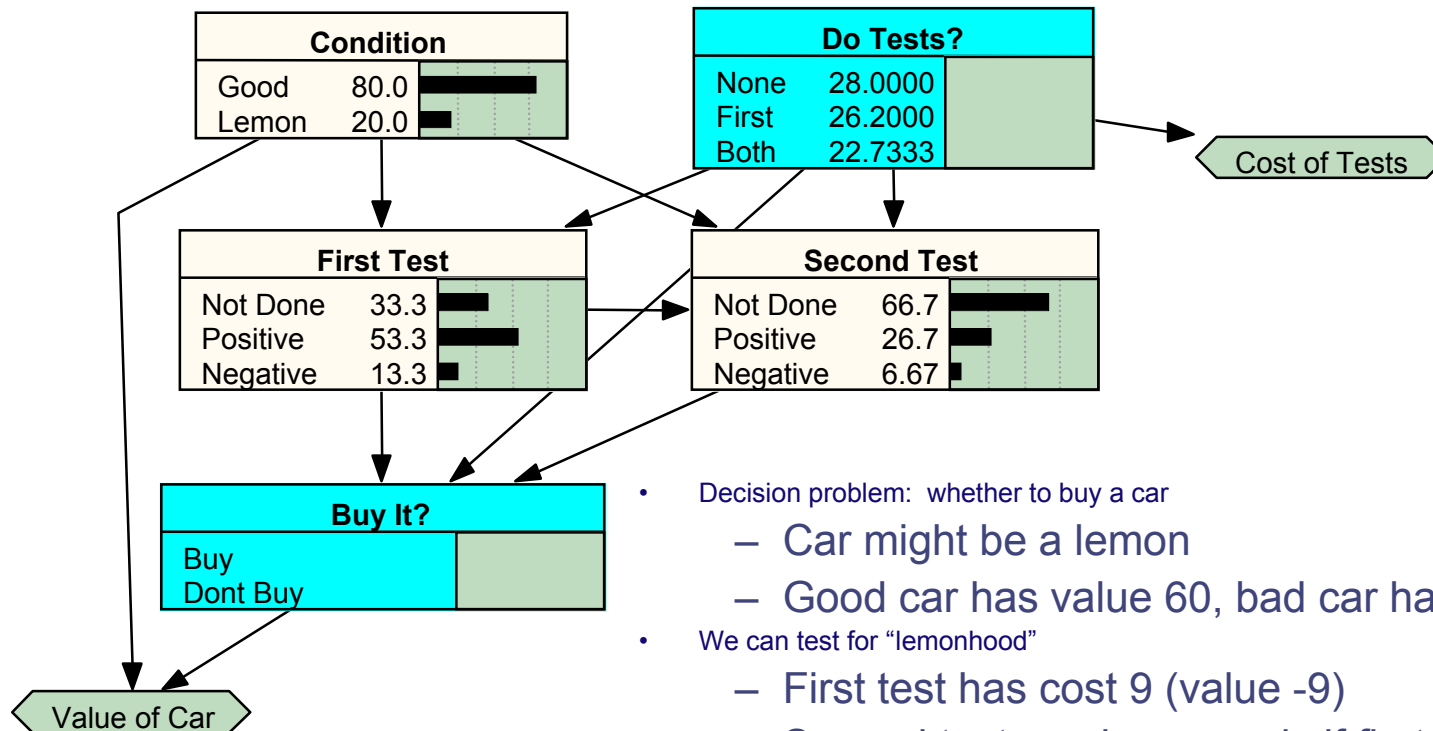
Example: Regulatory Decision



Policy question: permit or ban chemical?

- Need to trade off economic value against potential health effects
- Information is available to inform the decision

Example: Car Buyer

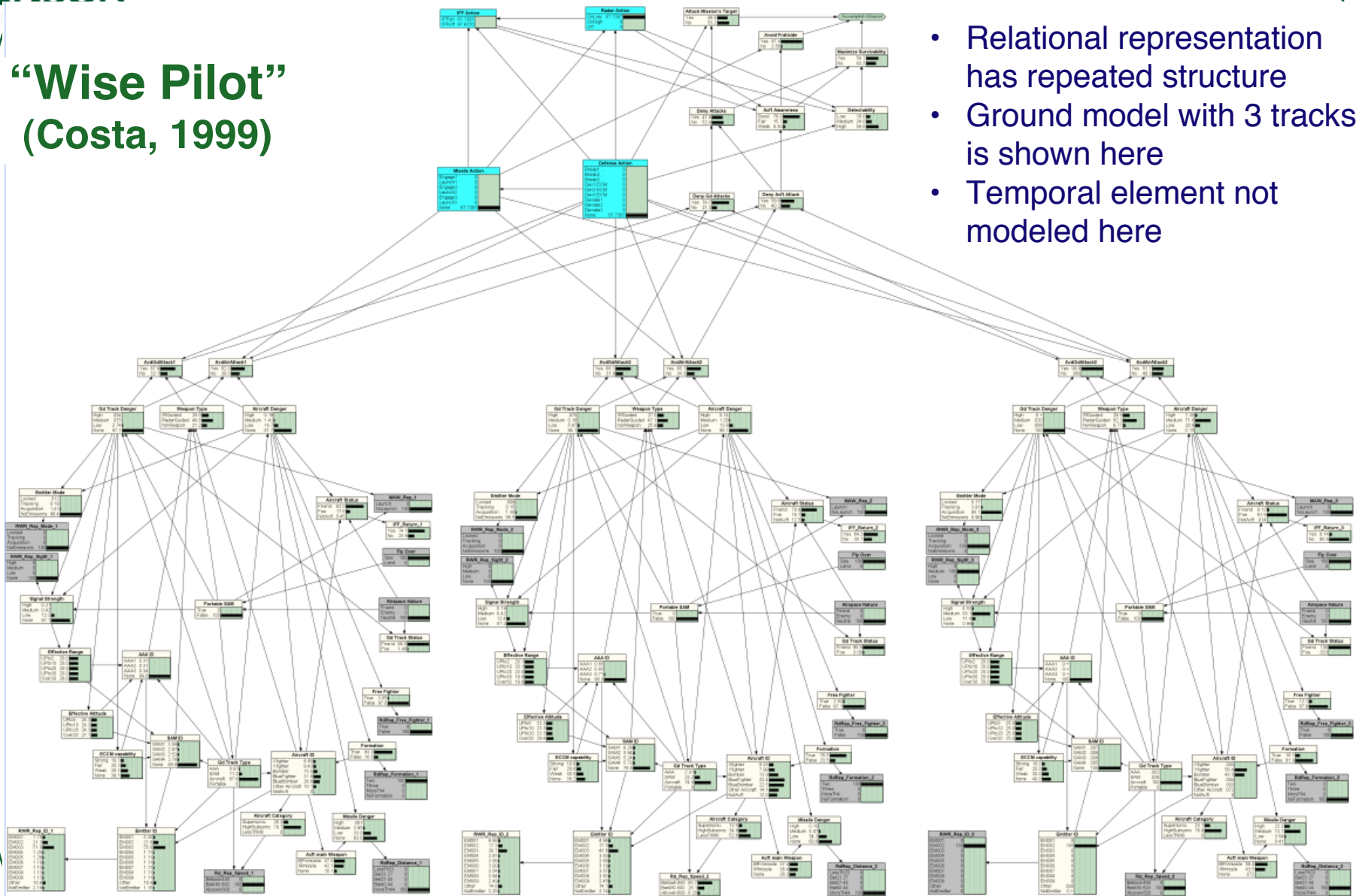


- Decision problem: whether to buy a car
 - Car might be a lemon
 - Good car has value 60, bad car has value -100
- We can test for "lemonhood"
 - First test has cost 9 (value -9)
 - Second test can be run only if first test is run
 - Second test has cost 4 (value -4)
- This example is provided with Netica
 - In Netica utilities of multiple terminal value nodes are assumed to be additive
- Netica is not the best tool for modeling decisions

Information Collection Decisions

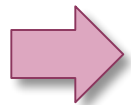
- Sometimes it is possible to collect information about relevant variables
- If information is free it cannot decrease your expected utility (and often increases your expected utility)
 - Why is this true?
- Expected value of perfect information (EVPI)
 - Add a decision “buy information” to the influence diagram. If “buy information” is selected uncertainty is resolved prior to decision.
 - The difference between expected utility of “buy information” and expected expected utility of next-highest branch is EVPI
 - Buy information if its cost (in utility units) is less than EVPI
- If information is costly it should be "purchased" only if it might change your decision
 - Why is this true?
- Expected value of imperfect information
 - The “buy information” decision gives only probabilistic information about the event in question
- Collecting imperfect information is sub-optimal if the cost exceeds EVPI
- In information collection decisions "cost of information" is a component of value

- Relational representation has repeated structure
- Ground model with 3 tracks is shown here
- Temporal element not modeled here



Unit 8 Outline

- Modeling decision problems with decision graphs



- Planning and temporal models

Planning

- A planner is designed to construct a sequence of actions intended to satisfy a goal
- Planning problem can be decomposed into:
 - Plan identification
 - Plan evaluation
- Plan identification typically involves heuristic rules designed to nominate “good” plans, e.g:
 - Heuristic rules for nominating actions that move planner closer to goal
 - Heuristic rules for breaking goal into subgoals
 - Heuristic “conflict resolution” strategies for selecting a rule to apply when conditions of multiple rules are met
 - Heuristic “problem space expansion” operators to apply when an impasse is reached (no rules can fire)
- Planning under uncertainty
 - Early AI planners assumed no uncertainty
 - Planning under uncertainty is an active research area
 - -Ways of responding to unexpected contingencies
 - » Contingency planning tries to pre-plan for all contingencies (computationally infeasible for all but the smallest problems)
 - » Replanning reinvokes planner when unexpected contingency occurs (may not be feasible in real time)
 - » Reactive planning pre-compiles stereotypic, fast reactions to contingencies (no time to develop good response)

Decision Theoretic Planning

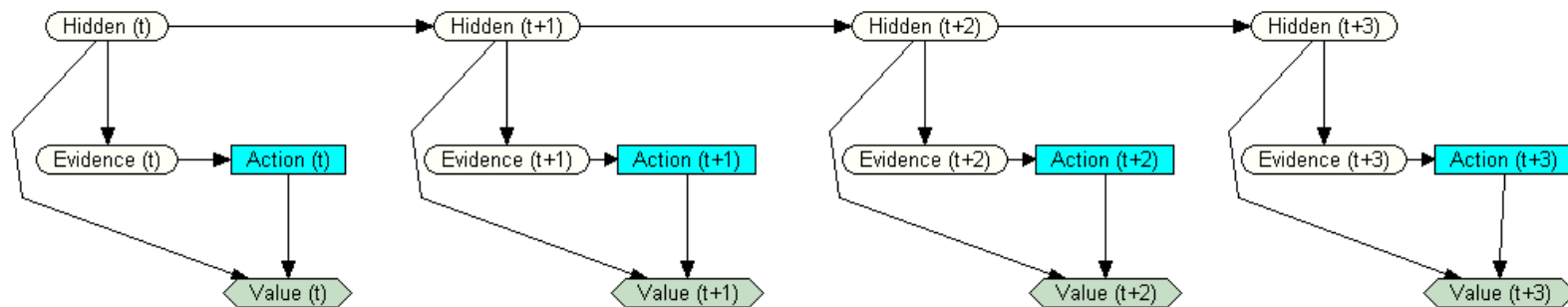
- Decision theoretic planning is an active area of research
- Key ideas in decision theoretic approach to plan evaluation
 - Goals are represented as attributes of utility
 - » Achievement of goal may not be all-or-nothing. There can be degrees of goal attainment
 - » One goal may be traded off against another
 - » Value of plan outcome is measured by utility
 - » Planner's utility function trades off degree of attainment among all goals
 - Uncertainty about plan outcomes is represented by probability
 - » Plans are evaluated prospectively by expected utility
 - Actions change the probabilities of outcomes
 - » A plan is a set of instructions for taking actions
 - » Actions may be conditioned on the state of the world at the time the action is taken
 - » The best plan is the one with the highest expected utility

Policies

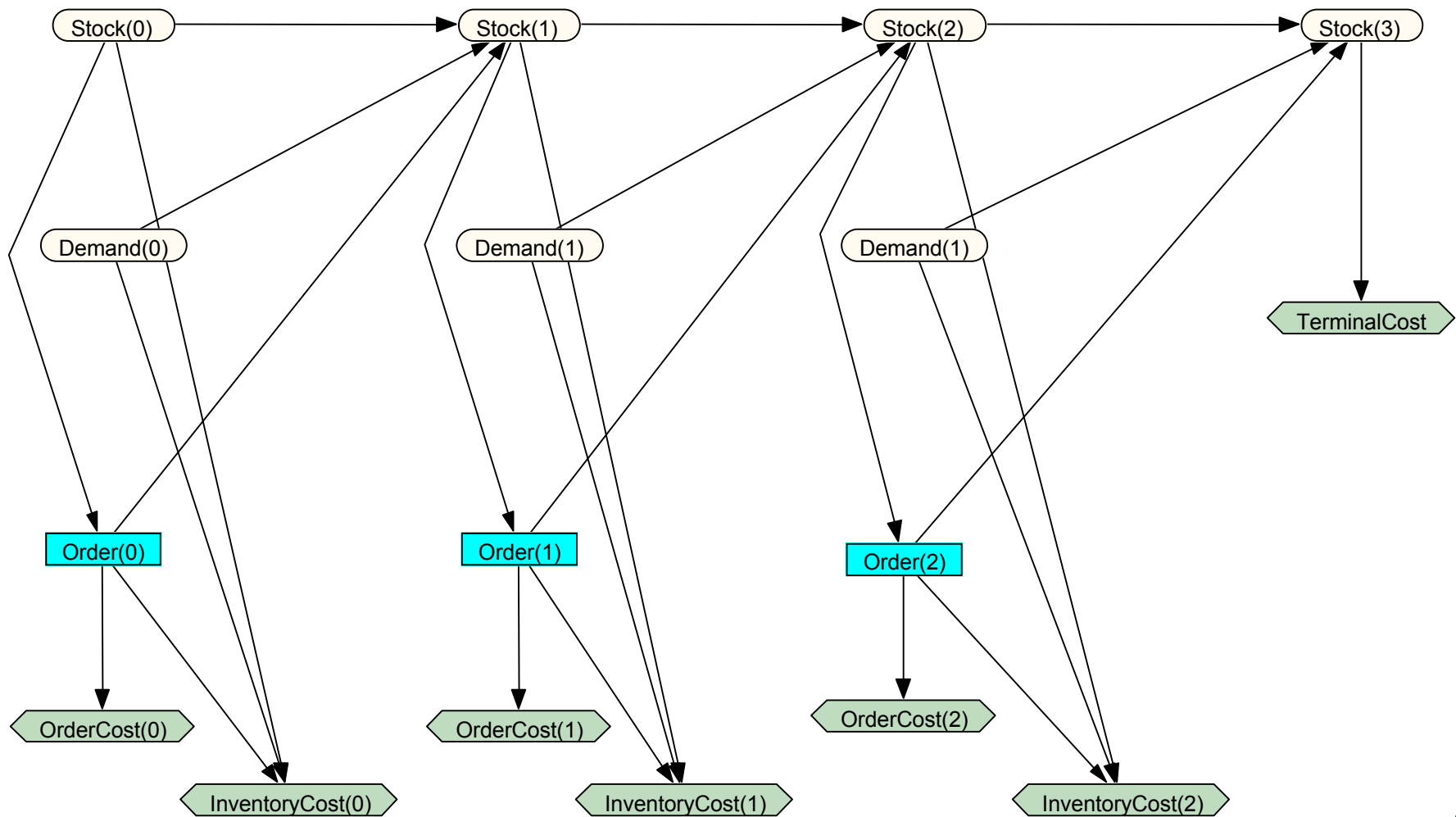
- A local policy for a decision node in an influence diagram is a function from the decision node's information predecessors to its action set
 - What you do at a decision node can depend only on what is known to you at the time you make the decision as represented by the information predecessors
 - Sometimes policies are randomized
- A policy for an influence diagram is a set of local policies
 - A policy determines (up to possible randomization) what the decision maker will do at every contingency
 - An optimal policy maximizes expected utility
 - Influence diagram solution algorithms find an optimal policy
- Solving an influence diagram
 - The best decision to take today depends on outcomes tomorrow which will depend on how I decide tomorrow
 - “Folding back” or dynamic programming algorithm (finite time horizon)
 - » Find the optimal policy for the decision farthest into the future
 - » Do until done: Find the optimal policy for a decision given that all decisions coming after it are taken optimally
- When there are many decisions, finding the optimal policy may be intractable
 - A policy can depend on all past actions and all information predecessors of past actions
 - We can design agents that forget all but a summary of their past history
 - These agents optimize on a restricted policy space (policy is allowed to depend on past actions only through history)

Markov Decision Processes

- MDPs are becoming a common representation for decision theoretic planning
- MDP represents temporal sequence of decisions
- Partially observable Markov decision process (POMDP) is MDP in which state is only partially observable
- POMDP has:
 - State that evolves in time with Markov transitions
 - Actions that are under control of decision maker
 - Values (or costs) at each time step
- Usually costs are assumed additive
- Dynamic decision network (DDN)
 - Augments DBN with decision and value nodes
 - Factored representation for POMDP state



MDP Example: Inventory Control



Inventory Example Continued

- $\text{Stock}(i)$ - amount of inventory in stock at time period i
 - Observed before order is placed
- $\text{Order}(i)$ - amount ordered for next time period
- $\text{Demand}(i)$ - amount of items requested by customers in time period i
- $\text{OrderCost}(i)$ - cost of order placed
 - Depends on how many units are ordered
- $\text{InventoryCost}(i)$ - cost of excess inventory and/or unmet demand
 - Depends on difference between stock plus incoming order and demand
- $\text{TerminalCost}(i)$ - cost of excess inventory at end of time horizon

The Dynamic Programming Optimality Equation

$$T_k(s_k) = \min_{a_k} \left\{ E[v_k(s_k, a_k) + T_{k+1}(s_{k+1}) | s_i, a_i, i \leq k] \right\}$$

- s_k is the state of the system at time k
- $T_k(s_k)$ is the optimal total payoff from time k to the end of the process
- a_k is the action taken at time k
- v_k is the single-period value at time k

This equation forms the basis for exact and approximate algorithms for solving MDPs and POMDPs

Solving a Dynamic Program

- For a finite-horizon problem with additive utility we can solve a dynamic program exactly using a recursive algorithm
 - Begin at terminal period: terminal costs are known and there is only one possible policy (remain terminated)
 - Go to previous period
 - For each policy and each state, compute current costs + future costs of following optimal policy from next period to end
 - Set optimal policy for state equal to minimum of policies computed under step 3 for that state, and set optimal cost to cost of that policy
 - If initial period we are done, else go to 2
- For infinite horizon problem this does not work
- Even for finite horizon problems this algorithm may be intractable

$$T_k(s_k) = \min_{a_k} \left\{ E[v_k(s_k, a_k) + T_{k+1}(s_{k+1}) \mid s_i, a_i, i \leq k] \right\}$$

Inventory Control Again (Example 3.2, Bertsekas, 1995, p. 23)

- Notation:
 - Time index k - varies from 0 to horizon N
 - State variable x_k - stock available at beginning of k th period
 - Control variable u_k - stock ordered (and immediately delivered) at beginning of k^{th} period
 - Random noise w_k - demand during the k^{th} period with given (constant) probability distribution
 - Purchasing cost cu_k - cost of purchasing u_k units at c per unit
 - Holding/shortage cost $r(x_k, u_k, w_k)$ - cost of excess inventory or unmet demand
 - Terminal cost $R(x_N)$ - cost of having x_N units left at end of N periods
- State evolution (excess demand is lost):
 - $x_{k+1} = \max\{0, x_k + u_k - w_k\}$
- “Closed loop” optimization: we know inventory x_k at the time order is made, so $u_k(x_k)$ can depend on system state
- Total cost of closed-loop policy $p = \{u_k(x_k)\}_{k=0, \dots, N-1}$:

$$J_{\pi}(x_0) = E \left[R(x_N) + \sum_{k=0}^N (r(x_k) + cu_k(x_k)) \right]$$

Specific Parameters and Constraints

- Inventory and demand are non-negative integers ranging from 0 to 2
- Upper bound on storage capacity: $x_k + u_k \leq 2$
- Holding/storage cost is quadratic: $r(x_k) = (x_k + u_k - w_k)^2$
- Per-unit purchasing cost is 1
- No terminal cost: $R(x_N) = 0$
- Demand distribution:
 - $P(w_k=0) = 0.1$
 - $P(w_k=1) = 0.7$
 - $P(w_k=2) = 0.2$
- Assume initial stock $x_0 = 0$

DP Solution: Periods 3 and 2

Period 2		Order Cost	Demand = 0	Demand = 1	Demand = 2	Total Cost
$x_2=0$	$u_2=0$	0	$0.1(0+0.0)$	$0.7(1+0.0)$	$0.2(4+0.0)$	1.5
	$u_2=1$	1	$0.1(1+0.0)$	$0.7(0+0.0)$	$0.2(1+0.0)$	1.3
	$u_2=2$	2	$0.1(4+0.0)$	$0.7(1+0.0)$	$0.2(0+0.0)$	3.1
$x_2=1$	$u_2=0$	0	$0.1(1+0.0)$	$0.7(0+0.0)$	$0.2(1+0.0)$	0.3
	$u_2=1$	1	$0.1(4+0.0)$	$0.7(1+0.0)$	$0.2(0+0.0)$	2.1
	$u_2=2$	2	Infeasible	Infeasible	Infeasible	--
$x_2=2$	$u_2=0$	0	$0.1(4+0.0)$	$0.7(1+0.0)$	$0.2(0+0.0)$	1.1
	$u_2=1$	1	Infeasible	Infeasible	Infeasible	--
	$u_2=2$	2	Infeasible	Infeasible	Infeasible	--

Order cost
Holding/storage cost
Previous period optimal cost
Demand probability
Total cost

Period 3: Total cost = Terminal cost = 0

DP Solution: Period 1

Period 1		Order Cost	Demand = 0	Demand = 1	Demand = 2	Total Cost
$x_1=0$	$u_1=0$	0	$0.1(0+1.3)$	$0.7(1+1.3)$	$0.2(4+1.3)$	2.8
	$u_1=1$	1	$0.1(1+0.3)$	$0.7(0+1.3)$	$0.2(1+1.3)$	2.5
	$u_1=2$	2	$0.1(4+1.1)$	$0.7(1+1.3)$	$0.2(0+1.3)$	3.68
$x_1=1$	$u_1=0$	0	$0.1(1+0.3)$	$0.7(0+1.3)$	$0.2(1+1.3)$	1.5
	$u_1=1$	1	$0.1(4+1.1)$	$0.7(1+0.3)$	$0.2(0+1.3)$	2.68
	$u_1=2$	2	Infeasible	Infeasible	Infeasible	--
$x_1=2$	$u_1=0$	0	$0.1(4+1.1)$	$0.7(1+0.3)$	$0.2(0+1.3)$	1.68
	$u_1=1$	1	Infeasible	Infeasible	Infeasible	--
	$u_1=2$	2	Infeasible	Infeasible	Infeasible	--

Order cost
Holding/storage cost
Previous period optimal cost
Demand probability
Total cost

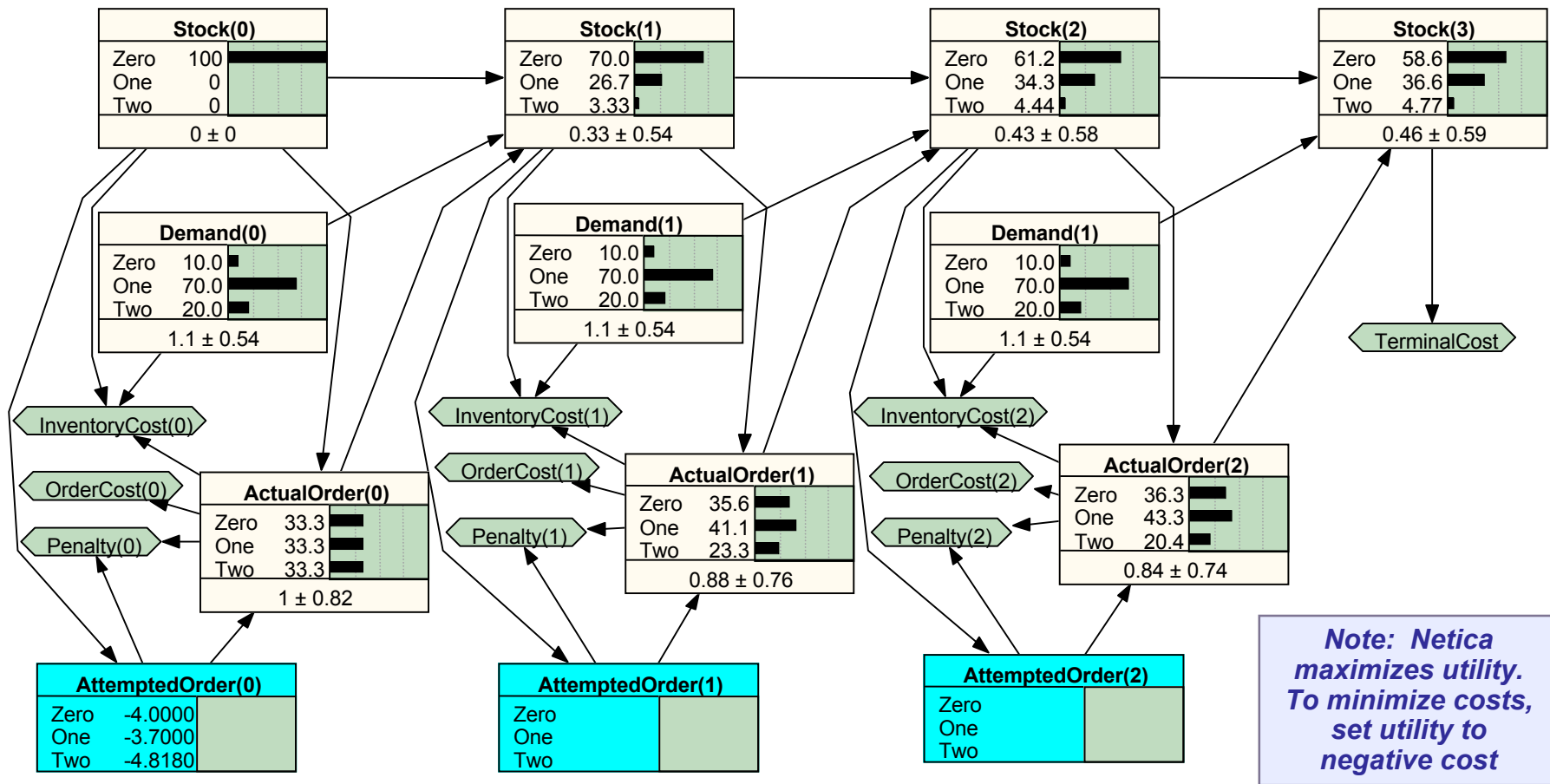
DP Solution: Period 0

Period 0		Order Cost	Demand = 0	Demand = 1	Demand = 2	Total Cost
$x_2=0$	$u_2=0$	0	$0.1(0+2.5)$	$0.7(1+2.5)$	$0.2(4+2.5)$	4.0
	$u_2=1$	1	$0.1(1+1.5)$	$0.7(0+2.5)$	$0.2(1+2.5)$	3.7
	$u_2=2$	2	$0.1(4+1.68)$	$0.7(1+1.5)$	$0.2(0+2.5)$	4.818
$x_2=1$	$u_2=0$	0	XXX	XXX	XXX	XXX
	$u_2=1$	1	XXX	XXX	XXX	XXX
	$u_2=2$	2	Infeasible	Infeasible	Infeasible	--
$x_2=2$	$u_2=0$	0	XXX	XXX	XXX	XXX
	$u_2=1$	1	Infeasible	Infeasible	Infeasible	--
	$u_2=2$	2	Infeasible	Infeasible	Infeasible	--

Order cost
Holding/storage cost
Previous period optimal cost
Demand probability
Total cost

If we assume initial stock $x_2=0$ we don't need to compute optimal policy for other values of x_2

Netica Model



Implementing constrained policies in Netica:

- Unconstrained “attempted decision” node
- “Actual decision” node is equal to attempted decision if feasible; otherwise is set to any feasible choice
- Penalty node adds a penalty if actual decision \neq attempted decision

Infinite Horizon Problems

- Infinite horizon problems pose challenges
 - Need to analyze limiting behavior as horizon tends to infinity
 - To achieve a bounded total cost we often use discounting (costs k periods in the future count only δ^k as much as current costs)
- Standard classes of problems:
 - Stochastic shortest path problems - Horizon is finite but uncertain and may be affected by policy
 - Discounted problems with bounded cost per stage - The total cost is well-defined because it is bounded above by a geometric series $M + \delta M + \delta^2 M + \dots$, where M is the upper bound of the per-stage cost
 - Problems with unbounded cost per stage - These are mathematically challenging because some policies may have infinite cost
 - Average cost problems - If we don't discount then all policies may have infinite cost, but the limit as the average cost may be well-defined for each finite horizon and may have a well-defined limit as the horizon tends to zero
- We may approximate a finite-horizon problem with an infinite-horizon problem for which the solution is well-defined and tractable
- Typically we assume stationarity
 - System evolution equation, per-stage costs, and random disturbance do not change from one stage to the next
 - Typically optimal policy is also stationary

Value Iteration Algorithm

- Assume:
 - Problem is stationary (therefore optimal policy is stationary)
 - Finite state space (infinite state spaces require more sophisticated analysis)
- The algorithm:
 - Initialize:
 - » Set $k=0$
 - » Begin with an initial guess for the optimal cost function $T_0(s)$
 - Do until termination criterion is met:
 - » Increment k
 - » Apply DP iteration to find optimal cost $T_k(s)$ for each s
 - Terminate when change in value is less than a threshold
- When state space is finite and problem is stationary value iteration converges to the optimal policy

- The DP iteration:
$$T_k(s) = \min_a \left\{ E \left[v(s,a) + \sum_{s'} p(s'|s,a) T_{k-1}(s') \right] \right\}$$

Policy Iteration Algorithm

- Assume:
 - Problem is stationary (therefore optimal policy is stationary)
 - Finite state space (infinite state spaces require more sophisticated analysis)
- The algorithm:
 - Initialize:
 - » Set $k=0$
 - » Begin with an initial stationary policy a_0
 - Do until termination criterion is met:
 - » Increment k
 - » Apply policy evaluation
 - » Apply policy improvement to find optimal policy $a(s)$ for each s
 - Terminate when policy does not change
- When state space is finite and problem is stationary policy iteration converges to the optimal policy in a finite number of steps
- Policy evaluation - solve for $T(s) = E \left[v(s, a_k(s)) + \sum_{s'} p(s'|s, a_k(s)) T(s') \right]$
 - (system of linear equations)
- Policy improvement: $a_k(s) = \arg \min_a \left\{ E \left[v(s, a) + \sum_{s'} p(s'|s, a) T_{k-1}(s') \right] \right\}$

Causality and Action

- We have discussed 2 types of action
 - Actions that change the world (intervening actions)
 - Actions that provide information about the state of the world (non-intervening actions)
- Intervening action
 - Normal time evolution of variable is “interrupted”
 - Agent sets the value of variable to desired value
 - Variables causally downstream are affected by change
 - Variables causally upstream are unaffected
- Non-intervening action
 - Variable is information predecessor to future actions
 - No other changes
- Every action has aspects of both
 - When we take an action our knowledge is updated to include the fact that we took the action (but we may forget actions we took)
 - Observing the world changes it (we often do not model the impact of actions whose impact is negligible)

Summary and Synthesis

- Graphical modeling approach can be extended to compact and composable representations for goals, actions and decision policies
- Decision theoretic planning involves selecting action policies that achieve high expected utility for agents
- Planning and decision making models tend to be intractable and approximation methods are needed

References for Unit 8

- Decision Theoretic Planning
 - Special issue of Artificial Intelligence July 2003 on decision theoretic planning
 - Dean, M. and Wellman, M.P. *Planning and Control*. Morgan Kaufmann, 1991.
- Dynamic Programming and Markov Decision Processes
 - Bertsekas, D.P. *Dynamic Programming and Optimal Control* (volumes 1 and 2), Prentice-Hall, 1995
 - Howard, R.A. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA, 1960.
- Decision Analysis
 - Clemen, R. *Making Hard Decisions*, PWS Kent, 2000.
 - Von Winterfeldt, D. and Edwards, W., *Decision Analysis and Behavioral Research*. Cambridge University Press, Cambridge, U.K., 1986.
- Influence Diagrams and Value of Information
 - Howard, R. Information value theory. *IEEE Transactions on Systems Science and Cybernetics* 2, 22-26, 1966.
 - Howard, R. and Matheson, J., Influence diagrams. In R.A. Howard and J.E. Matheson (eds) *Readings on the Principles and Applications of Decision Analysis*, Volume 2, Chapter 37, pp. 719-762. Strategic Decisions Group, Menlo Park, CA, 1984.
- Wise Pilot
 - Costa, P.C.G. *The Fighter Aircraft's Autodefense Management Problem: A Dynamic Decision Network Approach*, MS Thesis, IT&E, George Mason University, 1999.