Structure and Synthesis of Robot Motion

Motion Synthesis under Sensorimotor Uncertainty II

Subramanian Ramamoorthy
School of Informatics

13 February, 2012

What Would we Like to Tell our Robot?

Task specifications come in various forms

- Automata level logic-like specifications
- 'Global' level of motion planning
- Local feedback policies

The key issue:

You may want to deploy an autonomous robot to perform many different tasks – how to encode, for this whole family, the specification of how to perform each task instance?

What is the language? What is the 'program'?

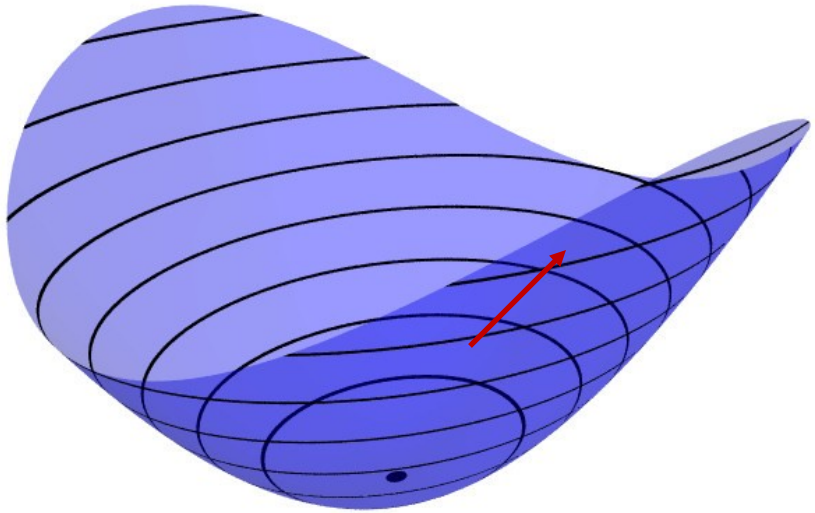
How to Encode a (local) Set of Trajectories? Potential Function

- Differentiable real-valued function,

$$U : \mathbb{R}^m \mapsto \mathbb{R}$$

- Treat the value as ‘energy’
- Then, gradient is the vector,

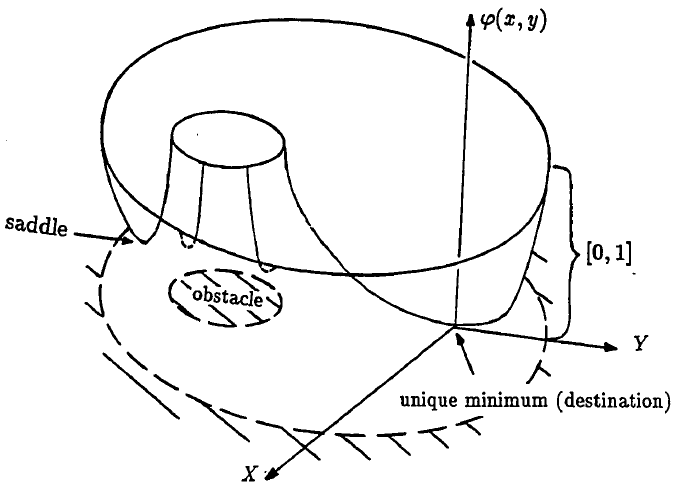
$$\nabla U(q) = DU(q)' = [\frac{\partial U}{\partial q_1}(q), \ldots, \frac{\partial U}{\partial q_m}(q)]'$$



- The gradient points in the direction that locally maximally increases U

Question

Potential/Navigation functions were designed to handle both path planning w.r.t. obstacles and actuator-level control in an integrated manner



Can we go further with this style of reasoning?

(How) can we encode a complex dynamically dexterous behaviour involving:

- Large unforeseen disturbances
- requiring some understanding of global dynamic behaviour
- With natural limits on sensing and actuation

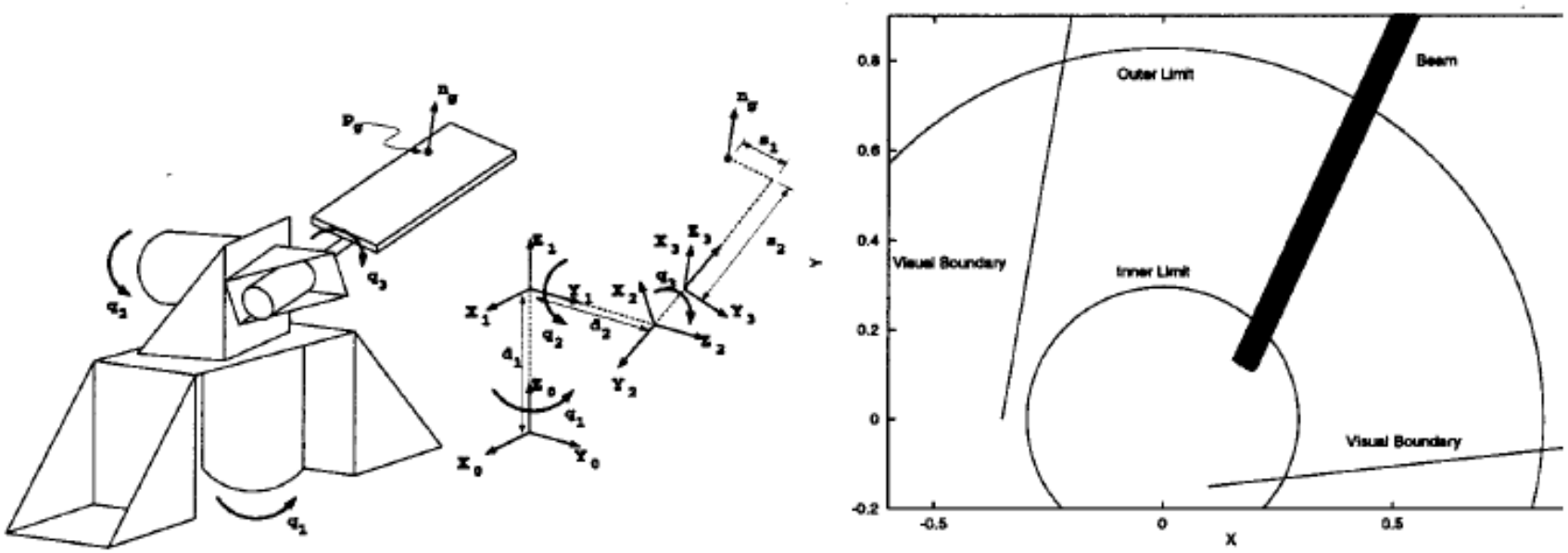
Sequential Composition of Dynamically Dexterous Behaviours [Burridge et al.]

- Explore control issues that arise from dynamical interactions between robot and environment
  - Active balancing, Hopping, Throwing and catching, Juggling
- Explore how such dexterous behaviors can be marshaled towards goals whose achievement requires rudiments of strategy
- Develop algorithms that are sufficiently tractable to allow correctness guarantees AND estimates of domain of attraction

A Task - Juggling

- Robot with flat paddle
  - required to strike repeatedly at thrown ball
  - until ball is brought to rest on the paddle at specified location
- Reachable workspace is disconnected
  - Ball and paddle can't remain in contact and approach goal location
  - Forces machine to *let go* for a time to bring the ball to desired state

The Buhgler Arm



Sequential Controller Composition

- Controller compositions guarantee that a ball introduced in the “safe workspace” remains there and is ultimately brought to the goal
- Partition of state space induced by a palette of pre-existing feedback controllers
- Each cell associated with a unique controller, chosen such that entry into a cell guarantees passage to successively “lower” cells until the “lowest” goal cell is reached

Behaviours

- Robotic implementations of user specified tasks
- Amenable to representation as state regulation (via feedback) to specified goal set, in the presence of obstacles
- Closed loop dynamics of a plant operating under feedback
- No single feedback algorithm will successfully stabilize the large range of initial conditions

Dynamical Pick and Place

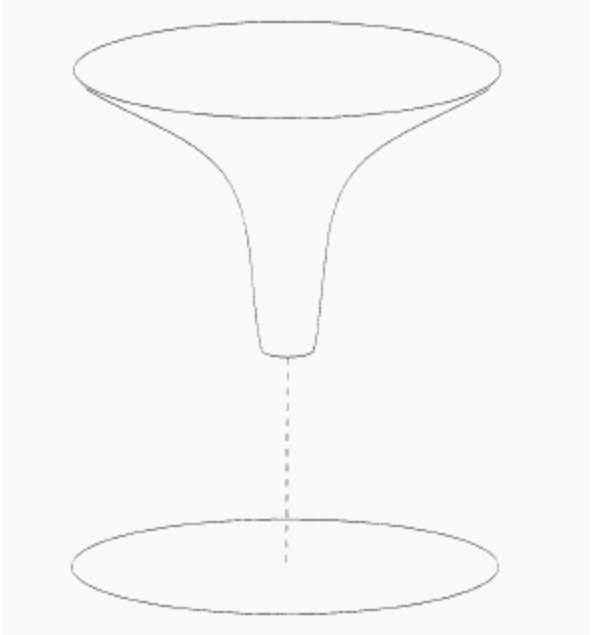
- Traditional motion planning finds an obstacle free path in 3D space

- This work can be considered as path planning in a higher dimensional state space (i.e., more constraints, expressible as differential equations)

- Feedback confers robustness. Use this principle at a higher level and attempt to construct basic strategies

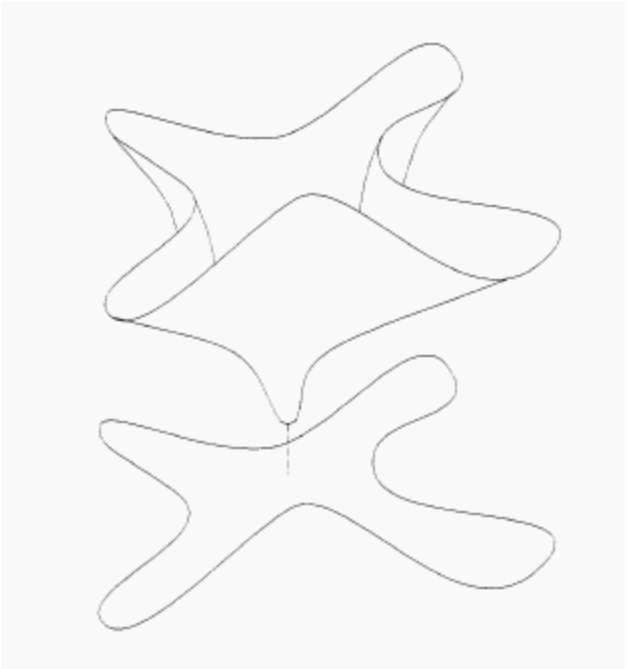
Feedback Strategies as Funnels

- For our purposes, feedback strategies result in invariant regions
- Strictly speaking, these invariant regions are characterized by *Lyapunov* functions or similar constructs

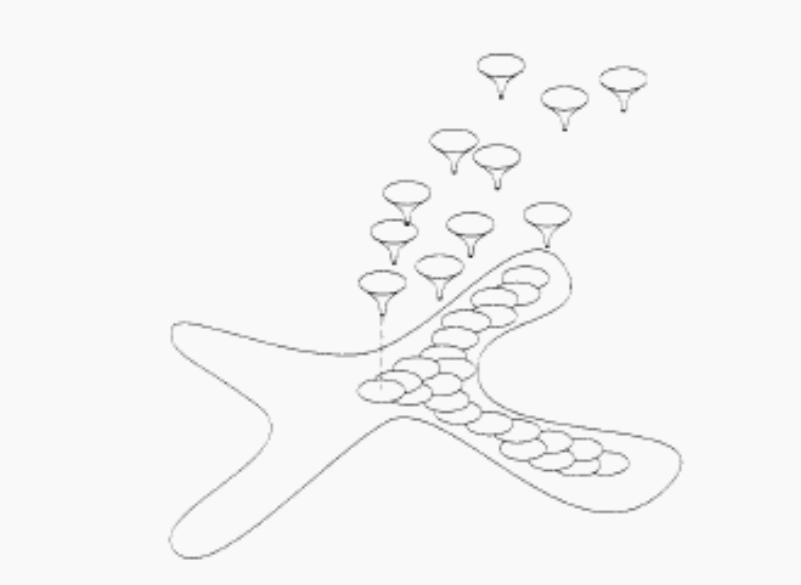


Feedback Strategies with Obstacles

- Most meaningful tasks include obstacles of one kind or another
- Obstacles tend to ‘warp’ the shape of the funnels
- Obstacles can result in “disjoint functions” in state space



Sequential Composition



Physical Setting

Hardware

- 3 DOF direct drive machine
- 2 cameras detect ball at 60 Hz
- Obstacle is a beam just above the paddle's height
- State space (tangent space notation)

$$Tb = (b, \dot{b}) \in TB$$
$$q = (\phi, \theta, \psi)$$
$$Tq = (q, \dot{q}) \in TQ$$

Physical Setting

Software

- Ball states, *Tb*, interpreted at 60 Hz by vision
- Vision data used by observer to estimate true *Tb*, interpolated at 330 Hz
- A memory-less transformation (mirror law) produces reference robot positions
- The reference robot positions fed to an inverse dynamics, joint-space controller

Discuss: Why do you **need** the "mirror law"?

The control system

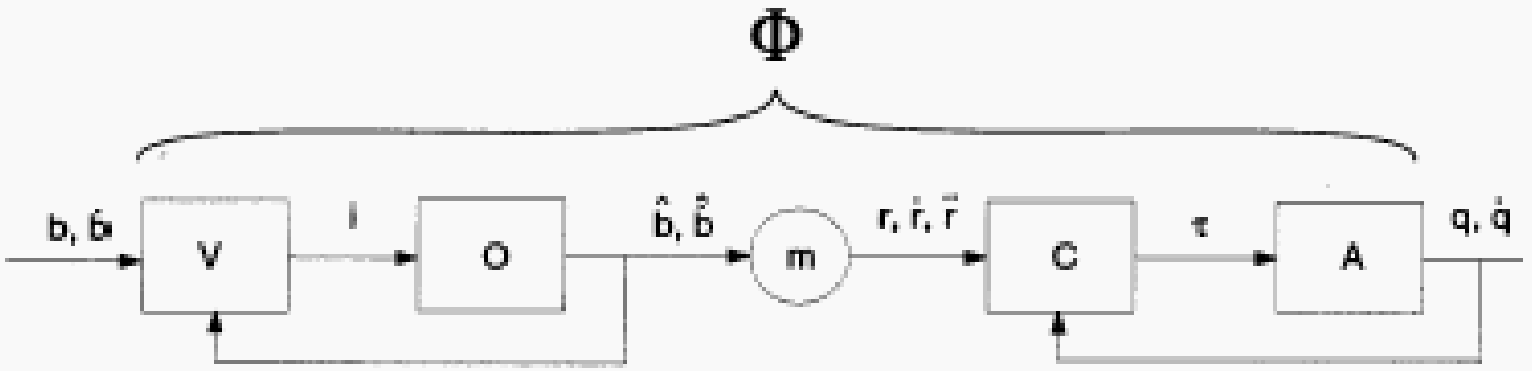


Fig. 6. Flow chart showing the various functional blocks of the system: vision, V; observer, O; mirror law, m; control, C; and actuation, A. The parameters of interest to this paper all reside in m.

The Closed Loop System

- Repetitive continuous trajectories represented as “return map”
- Discrete event sampled mapping of the periodic orbit
  - Analysis uses the notion of *Poincare section*

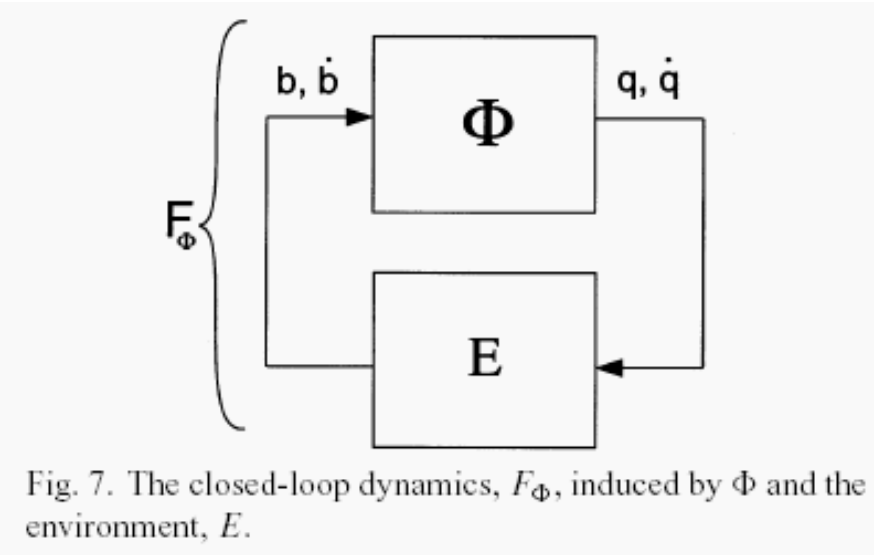
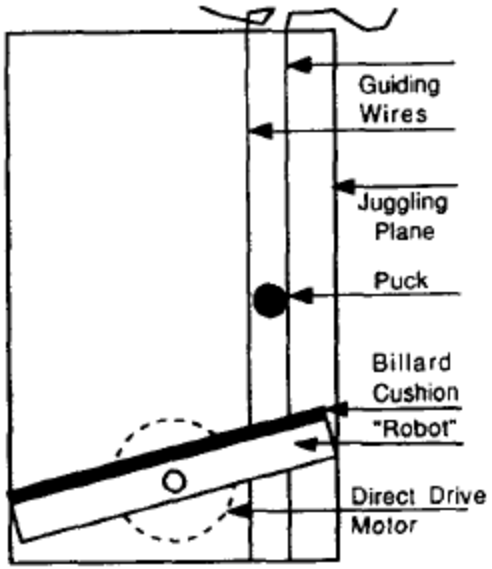


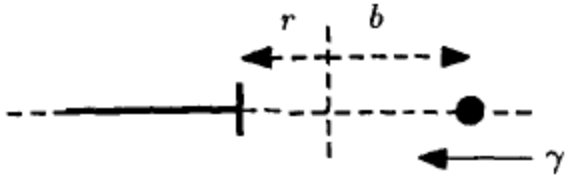
Fig. 7. The closed-loop dynamics, F_Φ, induced by Φ and the environment, E.

1-*dim* Mirror Law

Start with a line juggler



Task:



An open loop way would be to enforce post-contact vel.

$$\dot{b}' = -\alpha\dot{b} + (1+\alpha)\dot{r} = \mathbf{c}(\dot{b}, \dot{r}),$$

The free dynamics of the ball is simply:

$$\begin{bmatrix} b(t) \\ \dot{b}(t) \end{bmatrix} = \begin{bmatrix} b' + \dot{b}'t - \frac{1}{2}\gamma t^2 \\ \dot{b}' - \gamma t \end{bmatrix}.$$

This works but the result isn't very stable – small noise can have big effects

Mirror Law Control

Define a mapping from the phase space of ball to configuration space of robot arm

So, mirror law is based on getting the effector to $q(b) = (\phi_b, \theta_b, 0)$

1. $\phi_r = \phi_b$ causes the paddle to track under the ball at all times.
2. θ_r mirrors the vertical motion of the ball (as it evolves in θ_b): as the ball goes up, the paddle goes down, and vice-versa, meeting at zero height. Differences between the desired and actual total ball energy lead to changes in paddle velocity at impact.
3. Radial motion or offset of the ball causes changes in θ_r, resulting in a slight adjustment of the normal at impact, tending to push the ball back toward the set point.
4. Angular motion or offset of the ball causes changes in ψ_r, again adjusting the normal so as to correct for lateral position errors.

Mirror Law

- Defining the vertical energy and radial distance as:

$$\eta \triangleq \gamma b_z + \frac{1}{2}\dot{b}_z^2 \quad \text{and,} \quad \rho_b \triangleq \sin(\theta_b)s_b$$

- We can describe the mirror law as:

$$m(w) \triangleq \Big[\underbrace{\phi_b}_{\text{(i)}} \underbrace{-\frac{\pi}{2} - (\kappa_0 + \kappa_1(\eta - \bar{\eta}))\left(\theta_b + \frac{\pi}{2}\right)}_{\text{(ii)}} + \underbrace{\kappa_{00}(\rho_b - \bar{\rho}_b) + \kappa_{01}\dot{\rho}_b}_{\text{(iii)}} \underbrace{\kappa_{10}(\phi_b - \bar{\phi}_b) + \kappa_{11}\dot{\phi}_b}_{\text{(iv)}} \Big]$$

Note: Sophistication of these expressions is minimal… (PD, really) The controller itself is of low complexity!

Domain of m_j

- No closed form expression of return map
- Difficult to ascertain the shape of the boundaries of domain of attraction
- Use experimental data to formulate an approximation of *safe domain*
- To speed up deployment, create numerical simulation of the juggler and use it to determine domain of attraction

Domain of m_j : Experiments

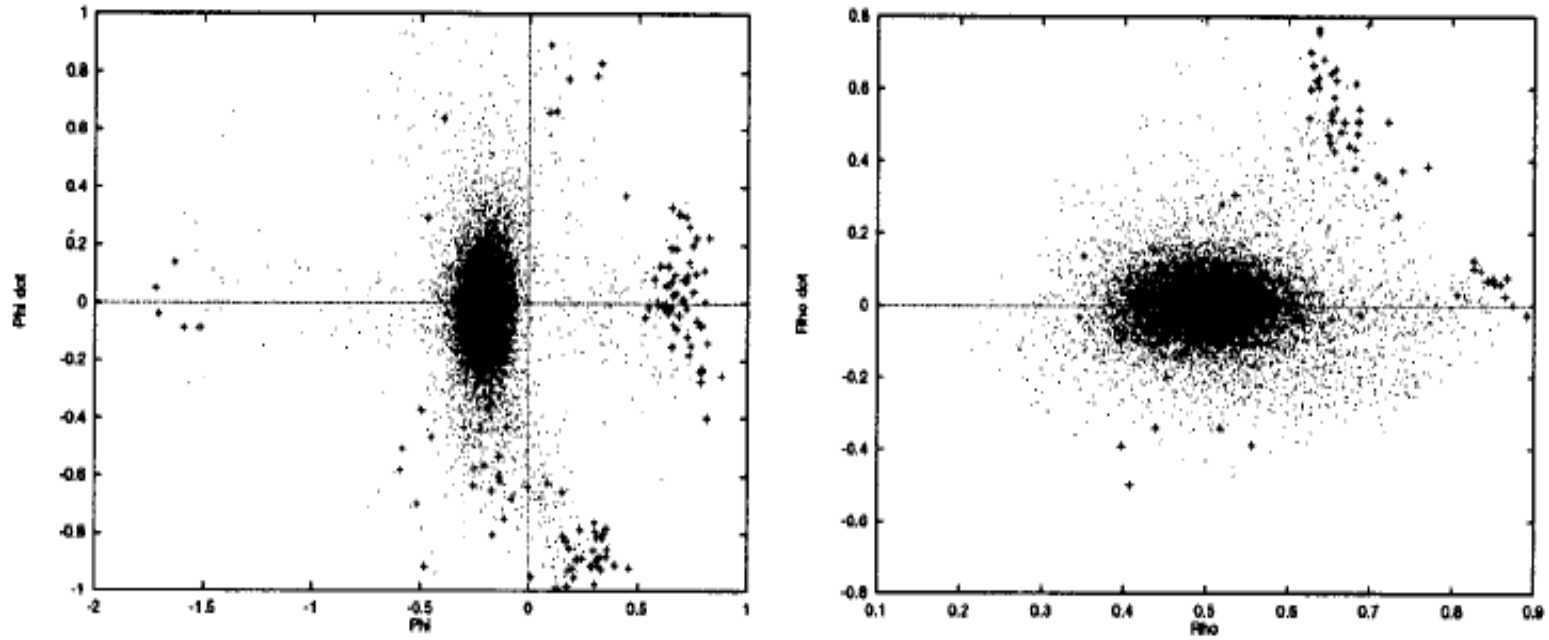


Fig. 8. Empirical data used to estimate the juggling domain, $\mathcal{D}(\Phi_J)$. Each dot (+ sign) represents in apex coordinates a ball trajectory that was successfully (unsuccessfully) handled under the action of Φ_J. Because of the properties of the vertical subsystem, most of these points are at nearly the same height, so only the horizontal coordinates are plotted.

Complete Control Strategy

- A set of controllers $\mathcal{U} = \{\Phi_1, ..., \Phi_N\}$ is designed to handle various scenarios
- Scenarios include:
  - Juggle (mirror law)
  - Palming
  - Catching

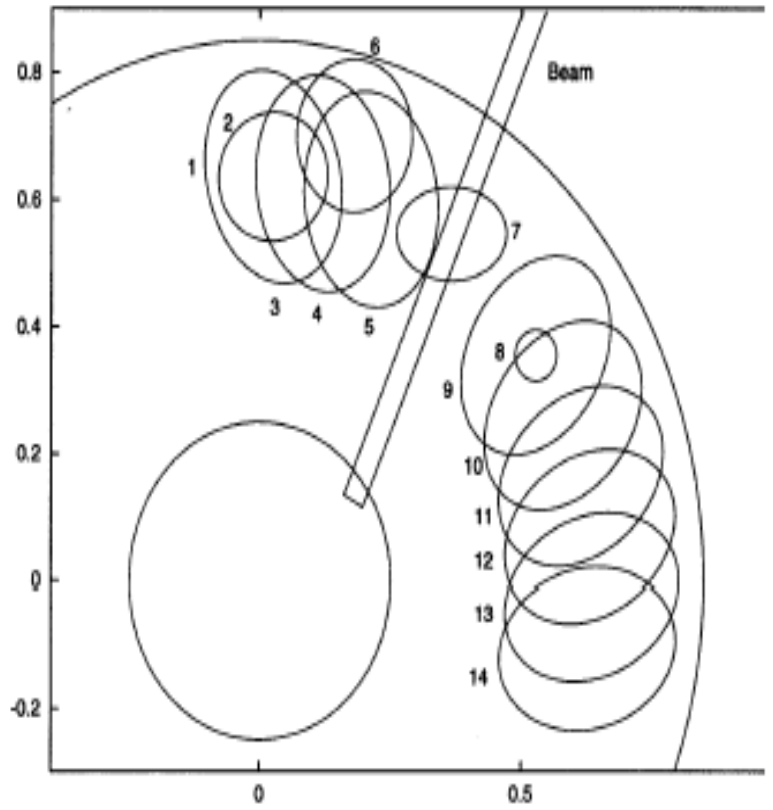
1. Let the queue contain Φ_1. Let $\mathcal{C}(\Phi_1) = \mathcal{D}(\Phi_1)$, $N = 1$, $\mathcal{U}'(1) = \{\Phi_1\}$, and $\mathcal{D}_1(\mathcal{U}') = \mathcal{D}(\Phi_1)$.
2. Remove the first element of the queue, and append the list of all controllers which *prepare* it to the back of the list.
3. While the first element of the queue has a previously defined cell, $\mathcal{C}$, remove the first element without further processing.
4. For Φ_j, the first unprocessed element on the queue, let $\mathcal{C}(\Phi_j) = \mathcal{D}(\Phi_j) - \mathcal{D}_N(\mathcal{U}')$. Let $\mathcal{U}'(N+1) = \mathcal{U}' \cup \{\Phi_j\}$, and $\mathcal{D}_{N+1}(\mathcal{U}') = \mathcal{D}_N(\mathcal{U}') \cup \mathcal{D}(\Phi_j)$. Increment N.
5. Repeat steps 2, 3, and 4 until the queue is empty.

Composition of Domains

Table 1. The Full Deployment, with Controller Types, Goal Points, and Domain Types[a]

Ellipses	Type	Goal: $\bar{\phi}$	Domain Type
1	Φ_P	0.3	$\mathcal{D}_P$
2	Φ_C	0.3	$\mathcal{D}_C$
3	Φ_J	0.3	$\mathcal{D}_0$
4	Φ_J	0.15	$\mathcal{D}_0$
5–8	Φ_J	0.0	$\mathcal{D}_1$
9	Φ_J	−0.64	$\mathcal{D}_0$
10	Φ_J	−0.81	$\mathcal{D}_0$
11	Φ_J	−0.97	$\mathcal{D}_0$
12	Φ_J	−1.12	$\mathcal{D}_0$
13	Φ_J	−1.26	$\mathcal{D}_0$
14	Φ_J	−1.4	$\mathcal{D}_0$

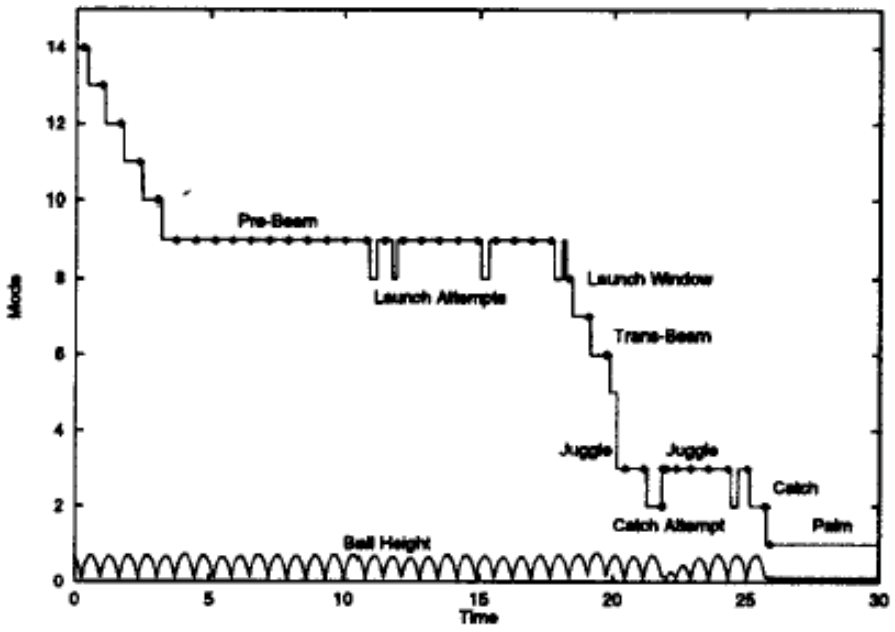
a. All goal points have the same radial component, $\bar{\rho}_0 = 0.6$, so we list here only the angular component, $\bar{\phi}$. Ellipse numbers correspond to those in Figure 16.

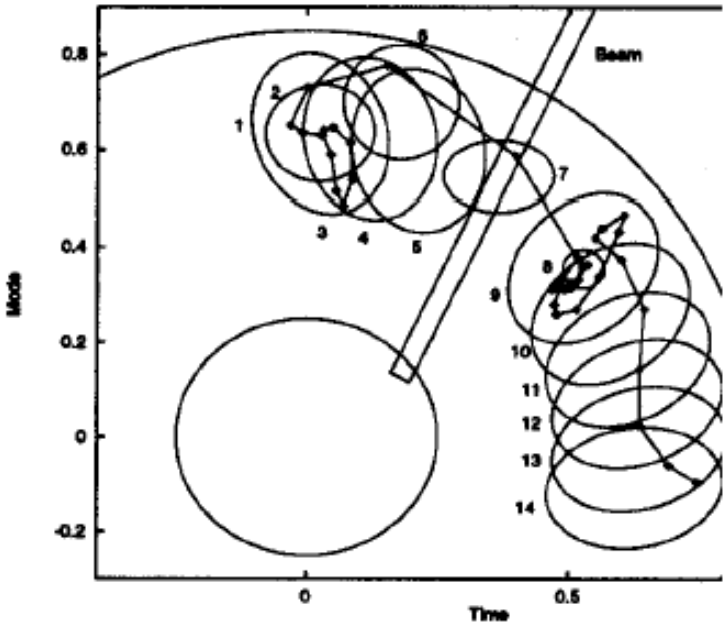


Problems faced / Assumptions

- Domain of attraction is experimentally determined and approximated by a fixed shape (ellipse)
- Occasionally, when earlier assumptions are violated, observer *hallucinates* that the ball is inside domain of attraction
- *Hallucination* leads to loss of global stability

A Typical Run





Results

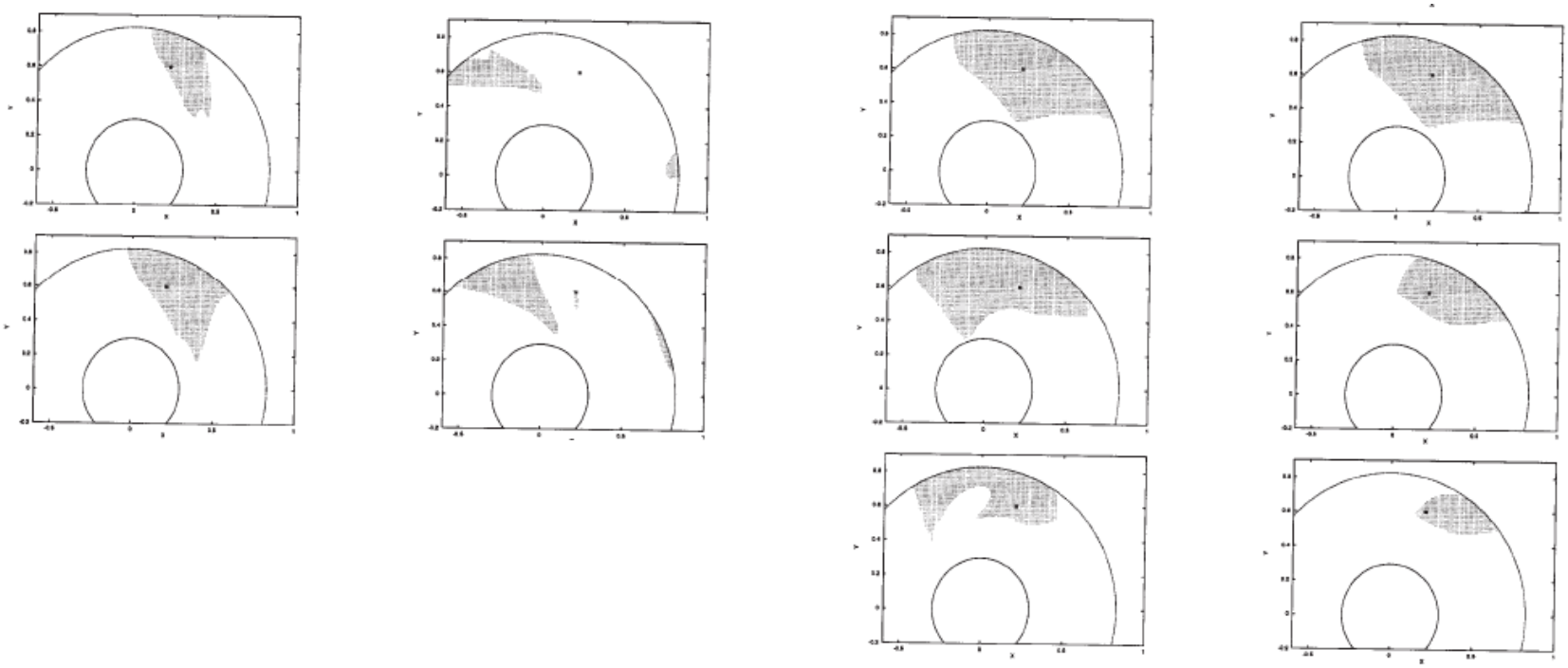
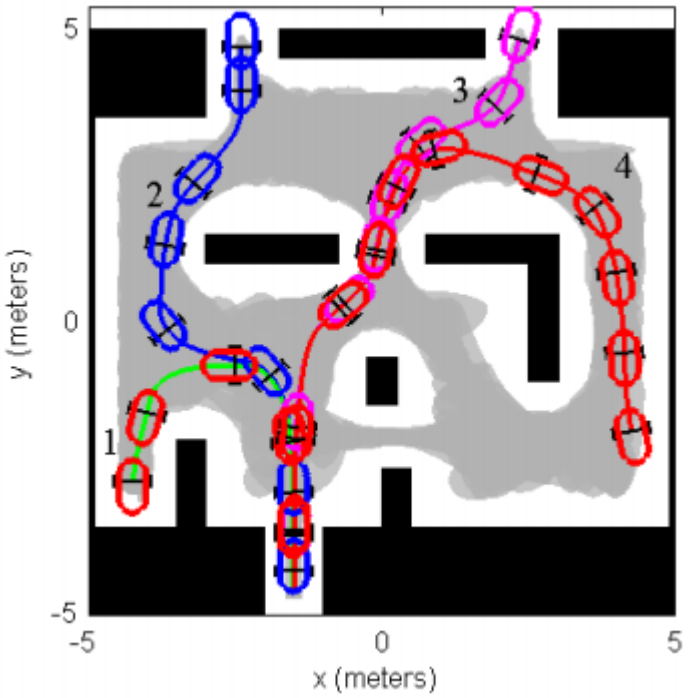


Fig. 11. Shaded regions denote initial conditions that were successfully contained in the workspace while being brought to the goal via iteration of $f_{\Phi,\gamma}$: varying initial $\dot{x}_i$ from negative (top) to positive (bottom) with $\dot{y}_i = 0$ (left column); varying initial $\dot{y}_i$ from negative to positive with $\dot{x}_i = 0$ (right column). The scale for all plots is meters.

Navigation in 2-dim Worlds [Conner et al.]

- Parameterized local feedback control policies
- Use in concert with discrete planning tools
  - (re)planning
  - analysis of safety, etc.

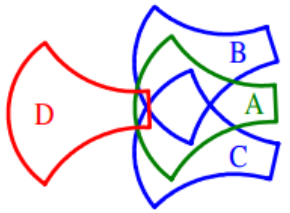
- Local policies are defined to properly handle issues such as non-holonomic dynamics



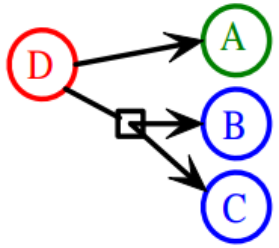
Key Ideas [Conner et al., Sec IV]

Design of a single cell:

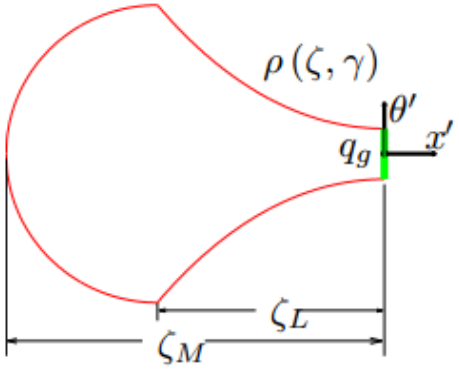
- Define a goal (q_g)
- Define a boundary
- Define a vector field in terms of level sets



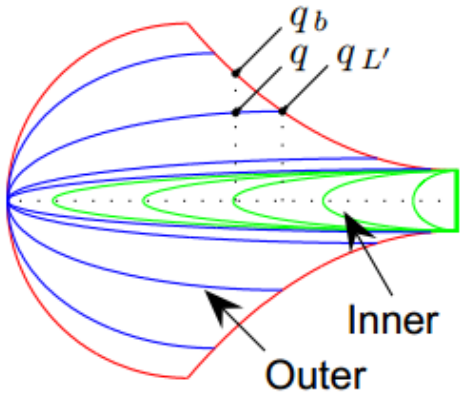
a) Prepares relationship between a collection of policies.



b) Graph representation of the induced discrete abstraction.



a) Cell Definition



b) Level Set Definition

In Coursework 2, implement a 'grid-cell version' of this

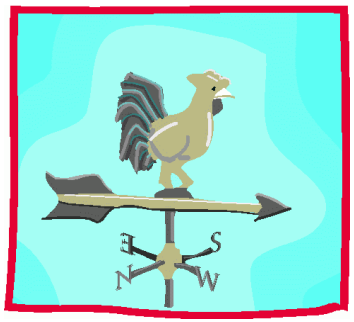
Explicitly Modelling Uncertainty

- Probability provides a mathematically sound basis for dealing with uncertainty.
- Combined with utilities, provides a basis for decision-making under uncertainty.

Simple Famous Example

Bob observes the weather forecast before deciding whether to carry an umbrella to work. Bob wishes to stay dry, but carrying an umbrella around is annoying.

Forecast



Setup of Decision Theory

- Set ***A*** of actions
  - **Umbrella**=*{true, false}*
- Set ***E*** of (unobserved) events
  - **Weather**=*{rain, sun}*
- Set ***O*** of observations
  - **Forecast**=*{rain, sun}*
- Probability distribution over
  - events ***P(E)***
  - observations given events ***P(O | E)***
- Utility function from actions and events to real numbers.

	Weather
sun	0.7
rain	0.3

	Forecast	
Weather	*sun*	*rain*
sun	0.6	0.4
rain	0.4	0.6

Weather	Umbrella	Utility
sun	*TRUE*	-10
sun	*FALSE*	100
rain	*TRUE*	100
rain	*FALSE*	-10

Choosing the Best Action

Let $U^a(\text{Bob} \mid e)$ be Bob's reward for taking action $a \in \mathbf{A}$ after event $e \in \mathbf{E}$ has occurred.

The expected utility for Bob after observing $o \in \mathbf{O}$ is

$$EU^a(\text{Bob} \mid o) = \sum_{e \in \mathbf{E}} P(e \mid o) \cdot U^a(\text{Bob} \mid e)$$

Optimal behavior — Given observation o choose the action that leads to maximal expected utility.

$$a^* = \text{argmax}_{a \in \mathbf{A}} EU^a(\text{Bob} \mid o)$$

Computing an Optimal Strategy for Bob

- A strategy for Bob must specify whether to take an umbrella for any possible value of the forecast.
- Suppose forecast predicts sun. What is Bob's expected utility for taking an umbrella ?

Computing Expected Utility for Bob for taking Umbrella

$$EU^{\mathsf{UM}}(\text{Bob} \mid \mathsf{F} = sun) = P(\mathsf{W} = sun \mid \mathsf{F} = sun) \cdot U^{\mathsf{UM}}(\text{Bob} \mid \mathsf{W} = sun) + P(\mathsf{W} = rain \mid \mathsf{F} = sun) \cdot U^{\mathsf{UM}}(\text{Bob} \mid \mathsf{W} = rain)$$

Weather	Umbrella	Utility
sun	*TRUE*	-10
sun	*FALSE*	100
rain	*TRUE*	100
rain	*FALSE*	-10

Marginal probability

$$\begin{aligned} P(\mathsf{F} = sun) =& P(\mathsf{F} = sun \mid \mathsf{W} = sun) \cdot P(\mathsf{W} = sun) + \\ & P(\mathsf{F} = sun \mid \mathsf{W} = rain) \cdot P(\mathsf{W} = rain) \\ =& 0.6 \cdot 0.7 + 0.4 \cdot 0.3 = 0.54 \end{aligned}$$

Bayes Rule

$$\begin{aligned} P(\mathsf{W} = sun \mid \mathsf{F} = sun) =& \frac{P(\mathsf{F} = sun \mid \mathsf{W} = sun) \cdot P(\mathsf{W} = sun)}{P(\mathsf{F} = sun)} \\ =& \frac{0.6 \cdot 0.7}{0.54} = 0.77 \end{aligned}$$

Computing Expected Cost

$$\begin{aligned} EU^{\mathsf{UM}}(\mathbf{Bob} \mid \mathsf{F} = sun) =& P(\mathsf{W} = sun \mid \mathsf{F} = sun) \cdot U^{\mathsf{UM}}(\mathbf{Bob} \mid \mathsf{W} = sun) + \\ & P(\mathsf{W} = rain \mid \mathsf{F} = sun) \cdot U^{\mathsf{UM}}(\mathbf{Bob} \mid \mathsf{W} = rain) \\ =& 0.77 \cdot (-10) + 0.23 \cdot 100 = 15.3 \end{aligned}$$

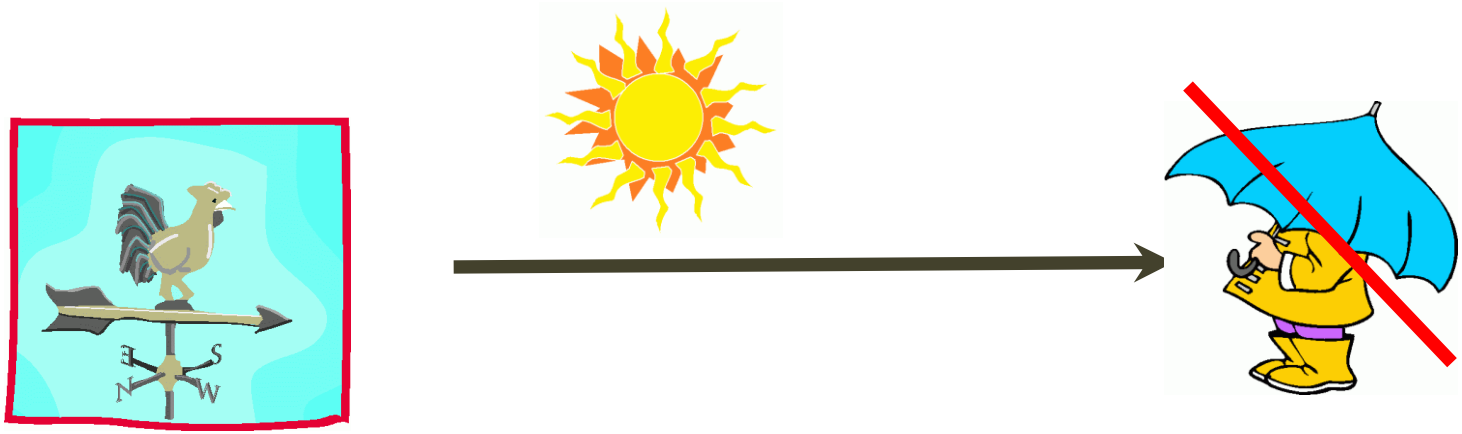
We now compute the expected utility for Bob for the case where Bob does not take an umbrella.

$$\begin{aligned} EU^{\overline{\mathsf{UM}}}(\mathbf{Bob} \mid \mathsf{F} = sun) =& P(\mathsf{W} = sun \mid \mathsf{F} = sun) \cdot U^{\overline{\mathsf{UM}}}(\mathbf{Bob} \mid \mathsf{W} = sun) + \\ & P(\mathsf{W} = rain \mid \mathsf{F} = sun) \cdot U^{\overline{\mathsf{UM}}}(\mathbf{Bob} \mid \mathsf{W} = rain) \\ =& 0.77 \cdot 100 + 0.23 \cdot (-10) = 74.7 \end{aligned}$$

Computing Bob's Best Action

$$EU^{\mathsf{UM}}(\mathrm{Bob} \mid \mathsf{F} = sun) \underset{(15.3)}{<} EU^{\overline{\mathsf{UM}}}(\mathrm{Bob} \mid \mathsf{F} = sun) \quad (74.7)$$

If the forecast predicts sun, then Bob should not take the umbrella



Computing Bob's Best Action

We now compute Bob's decision for the case where the forecast predicts rain. We have that

$$EU^{\mathsf{UM}}(\mathbf{Bob} \mid \mathsf{F} = rain) \overset{(34)}{<} EU^{\overline{\mathsf{UM}}}(\mathbf{Bob} \mid \mathsf{F} = rain) \overset{(56)}{}$$

We get the following strategy for Bob

	Forecast	
	rain	*sun*
Umbrella	*FALSE*	FALSE

Making Sequential Decisions

The newspaper forecast is more reliable, but costs money, decreasing Bob's utility by 10 units. There are now two decisions:

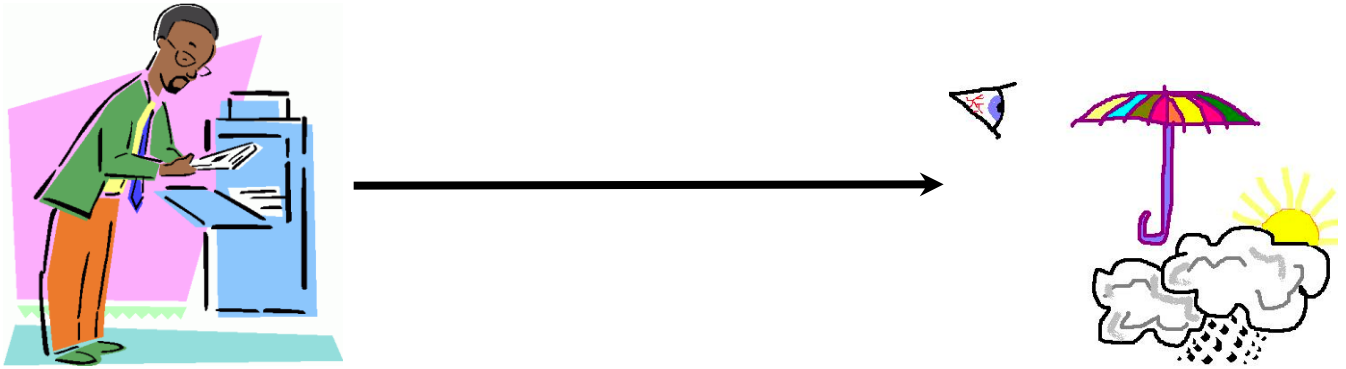
- Buying a newspaper
- Carrying an umbrella

	Forecast	
Weather	*sun*	*rain*
sun	0.8	0.2
rain	0.2	0.8

Weather	NP	Umbrella	Utility
sun	TRUE	*TRUE*	-20
sun	TRUE	*FALSE*	90
rain	TRUE	*TRUE*	90
rain	TRUE	*FALSE*	-20
...	....	*....*	....

Making Sequential Decisions

- Choosing the best action for one decision depends on the action for the other decision.
- How to weigh the tradeoff between these two decisions ?



Marginal probability

$$\begin{aligned} P^{\mathsf{NP}}(\mathsf{F} = sun) =& P^{\mathsf{NP}}(\mathsf{F} = sun \mid \mathsf{W} = sun) \cdot P(\mathsf{W} = sun) + \\ & P^{\mathsf{NP}}(\mathsf{F} = sun \mid \mathsf{W} = rain) \cdot P(\mathsf{W} = rain) \\ =& 0.8 \cdot 0.7 + 0.2 \cdot 0.3 = 0.62 \end{aligned}$$

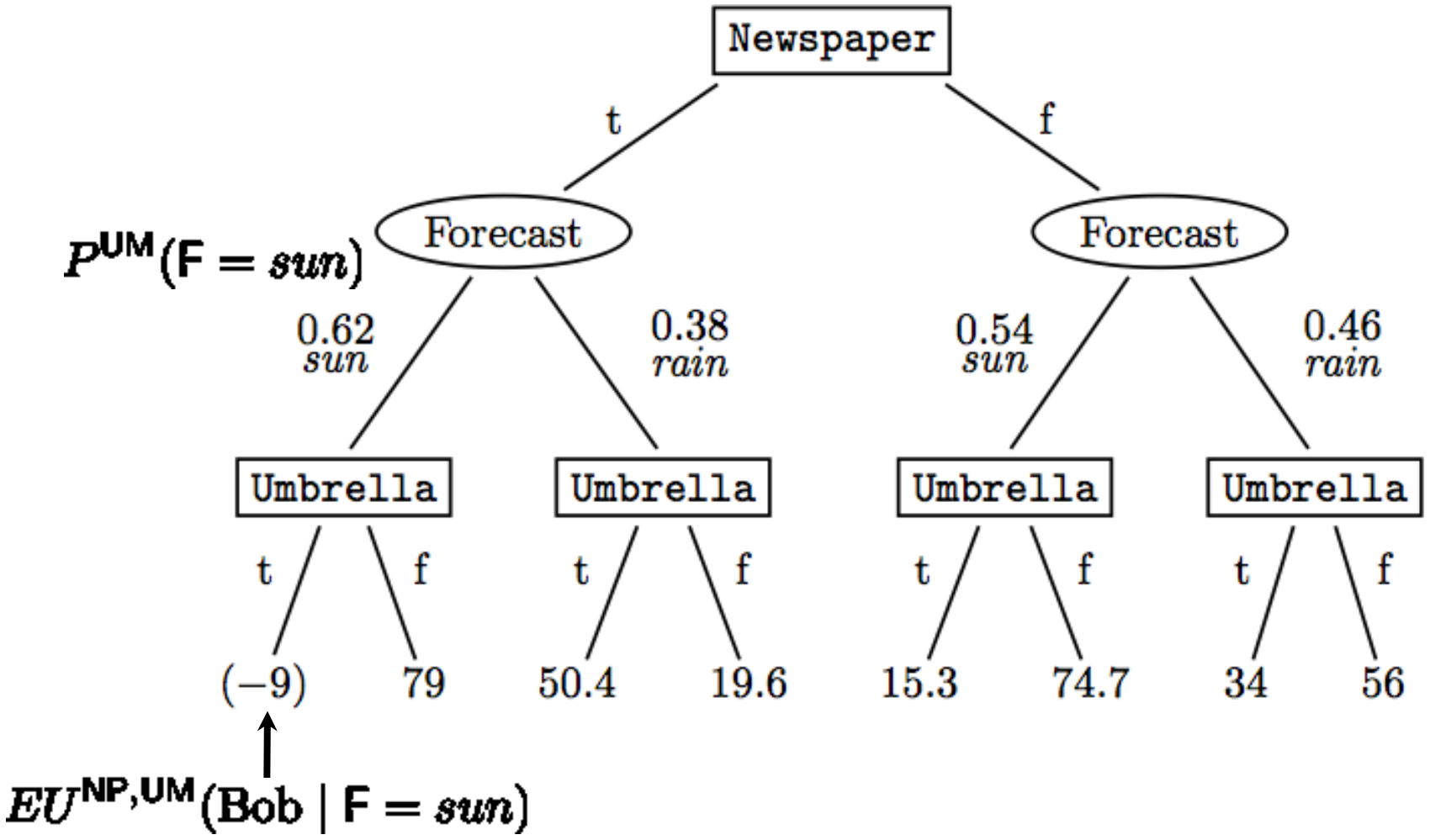
Bayes Rule

$$\begin{aligned} P^{\mathsf{NP}}(\mathsf{W} = sun \mid \mathsf{F} = sun) =& \frac{P^{\mathsf{NP}}(\mathsf{F} = sun \mid \mathsf{W} = sun) \cdot P(\mathsf{W} = sun)}{P^{\mathsf{NP}}(\mathsf{F} = sun)} \\ =& \frac{0.8 \cdot 0.7}{0.62} = 0.90 \end{aligned}$$

Expected utility

$$\begin{aligned} EU^{\mathsf{NP},\mathsf{UM}}(\mathrm{Bob} \mid \mathsf{F} = sun) =& P^{\mathsf{NP}}(\mathsf{W} = sun \mid \mathsf{F} = sun) \cdot U^{\mathsf{UM}}(\mathrm{Bob} \mid \mathsf{W} = sun) + \\ & P^{\mathsf{NP}}(\mathsf{W} = rain \mid \mathsf{F} = sun) \cdot U^{\mathsf{UM}}(\mathrm{Bob} \mid \mathsf{W} = rain) \\ =& 0.90 \cdot (-20) + 0.10 \cdot 90 = (-9) \end{aligned}$$

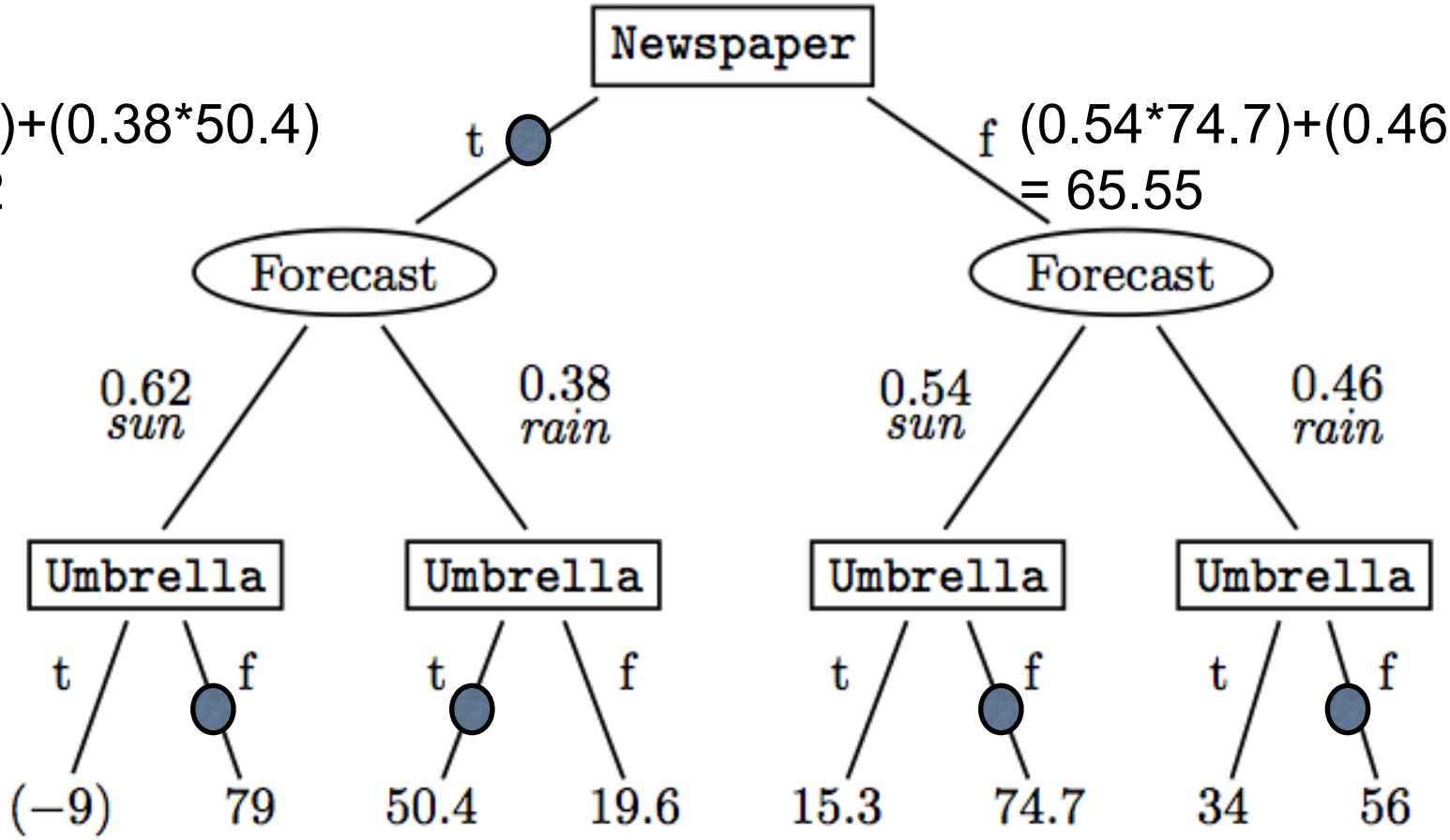
Decision Trees



Solving Decision Trees

(0.62*79)+(0.38*50.4)
= 68.132

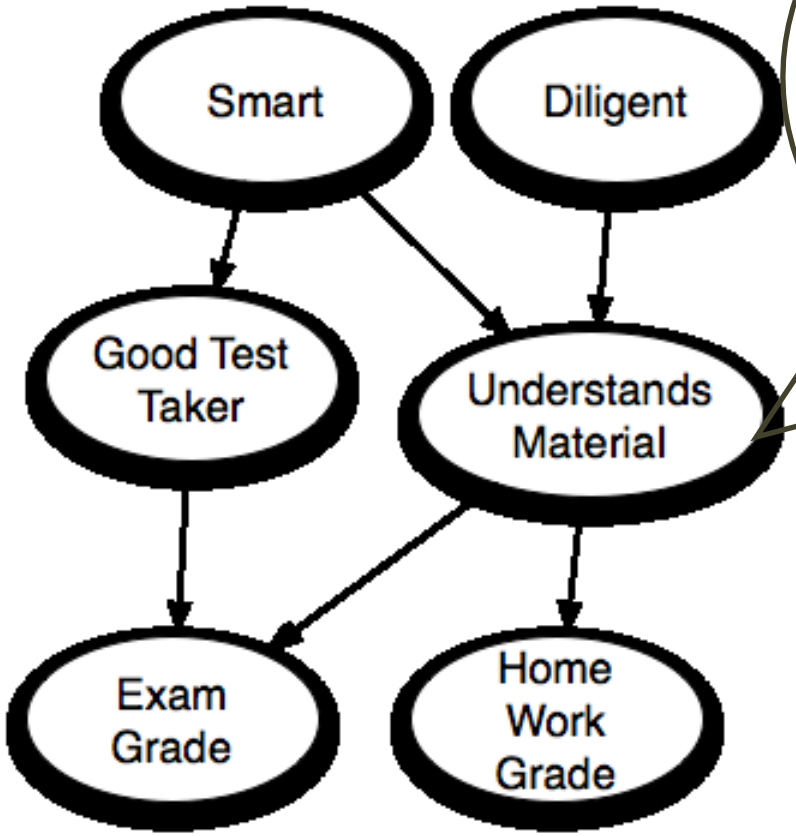
(0.54*74.7)+(0.46*56)
= 65.55



More General Probabilistic Models: An Example

- Domain Variables:
  - Smart (S)
  - Diligent (D)
  - Good test taker (G)
  - Understands Material (U)
  - Home work grade (H)
  - Exam grade (E)
- Each variable has a discrete domain.
- For example, Dom(S)={true, false}

Bayesian Networks



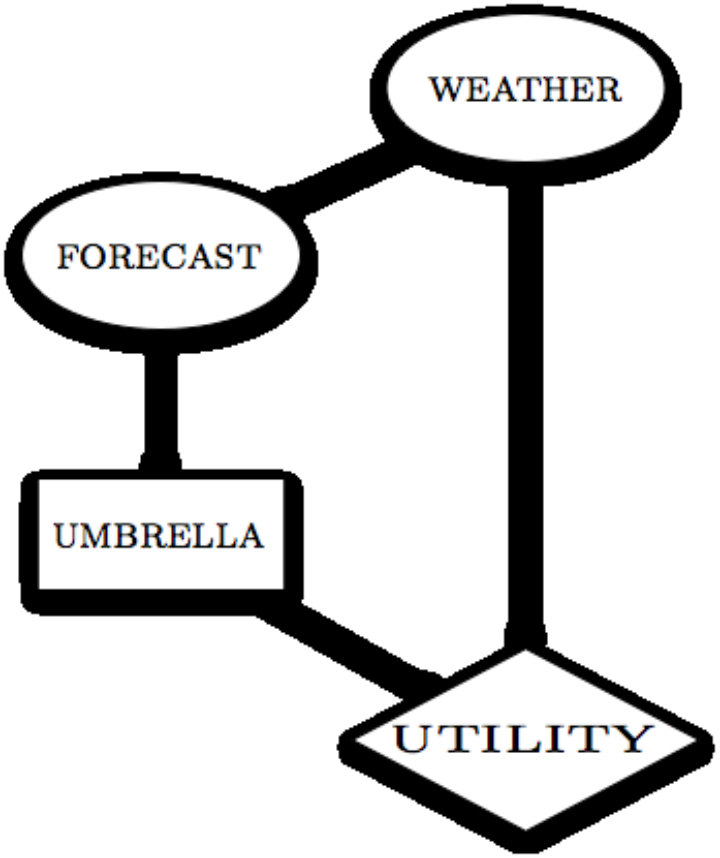
S	D	P(U | S, D)	
s	d	0.9	0.1
s	$\overline{d}$	0.3	0.7
$\overline{s}$	d	0.6	0.4
$\overline{s}$	$\overline{d}$	0.1	0.9

Conditional Probability Table

nodes = domain variables
edges = direct causal influence

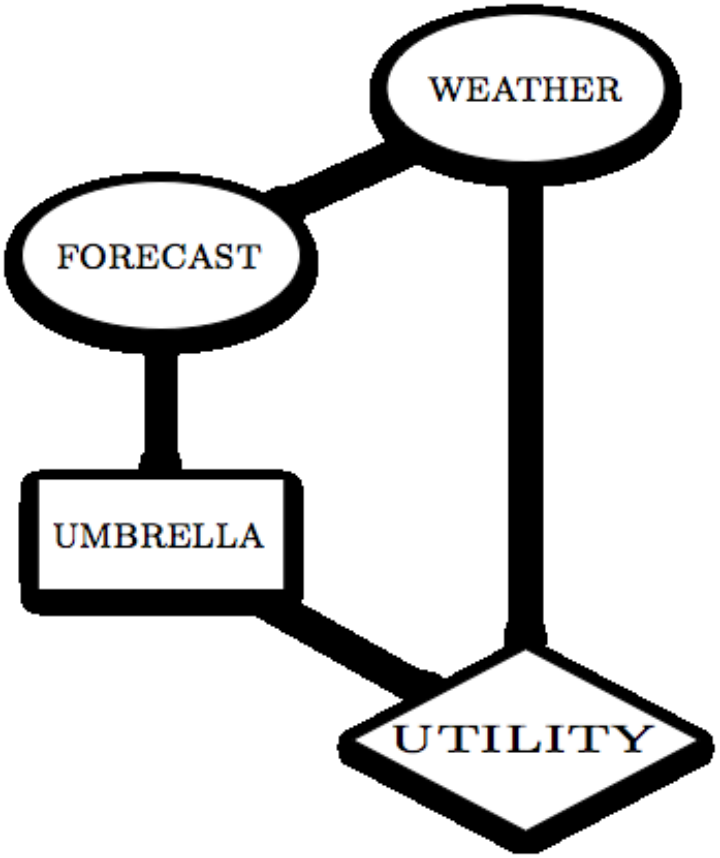
Influence Diagrams [Howard & Matheson '84]

- Influence Diagrams extend BNs for decision making.
- *Rectangles* are decisions; *ovals* are chance variables; *diamonds* are utility functions.
- Graph topology describes decision problem.
- Each node specifies a probability distribution (CPD) given each value of parents.

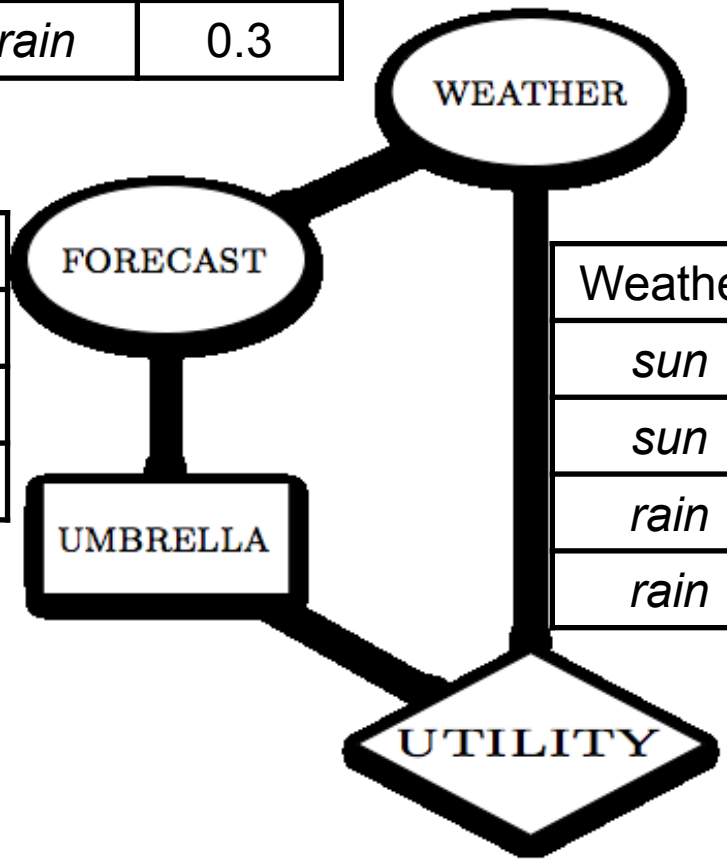


Influence Diagrams (ID)

- Parents of decisions (informational parents) represent observations.
- Parents of chance nodes represent probabilistic dependence.
- Parents of utility nodes represent the parameters of the utility functions.
- A strategy for a decision is a function from its informational parents to a choice for the decision. For each observation, a *pure* strategy prescribes a single choice of action for an agent.



Influence Diagram for Umbrella Scenario



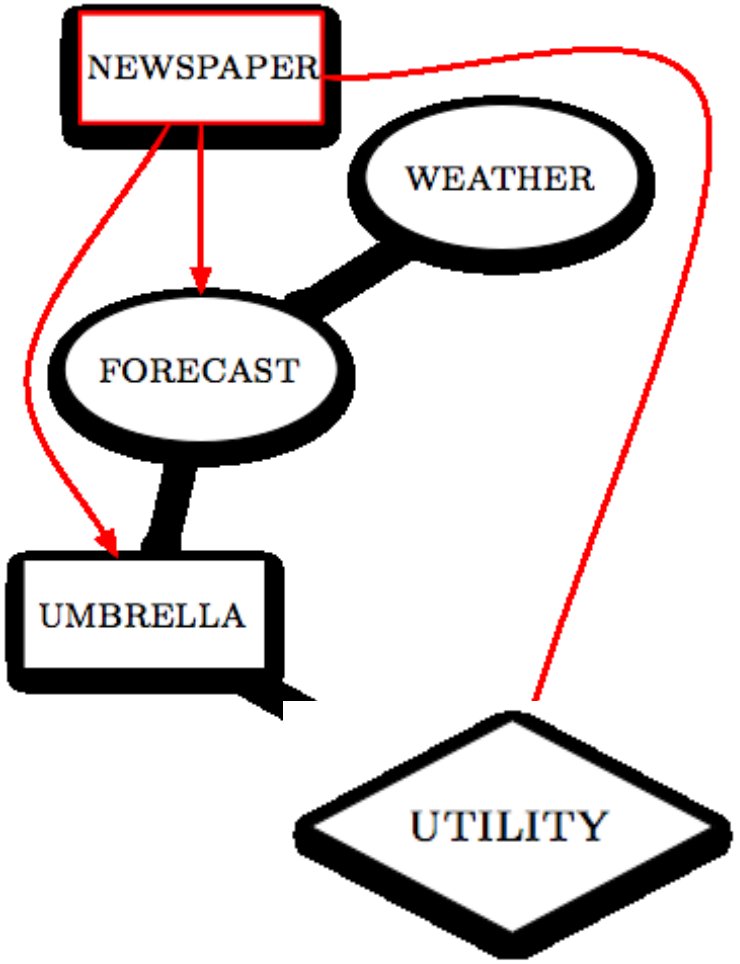
	Weather
sun	0.7
rain	0.3

	Forecast	
Weather	*sun*	*rain*
sun	0.6	0.4
rain	0.4	0.6

Weather	Umbrella	Utility
sun	*TRUE*	-10
sun	*FALSE*	100
rain	*TRUE*	100
rain	*FALSE*	-10

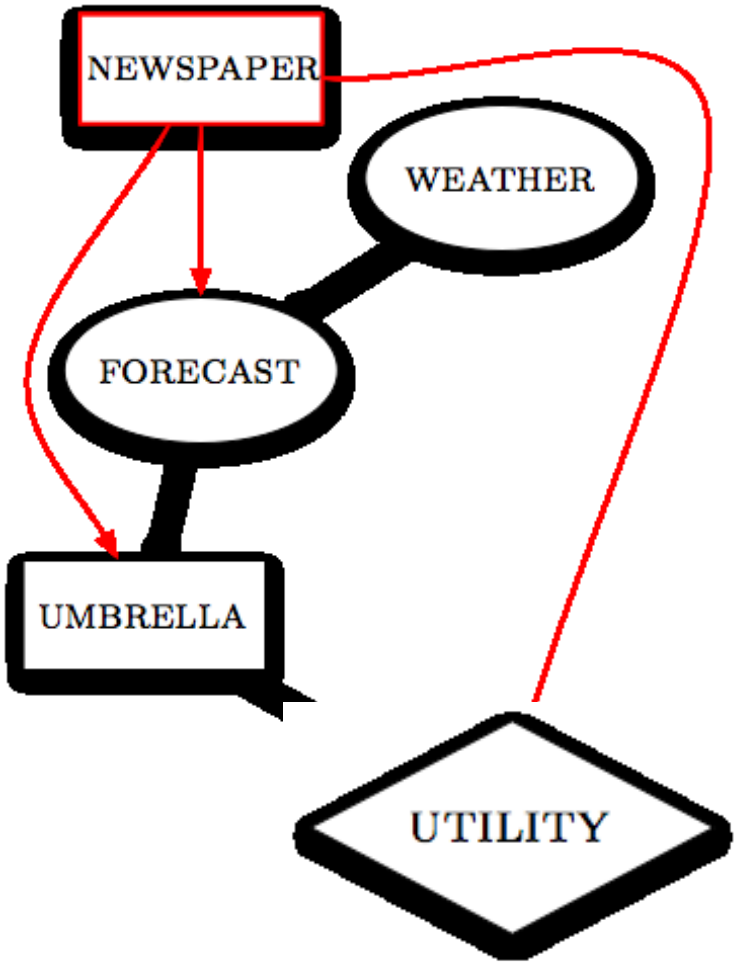
ID for Extended Umbrella Example

New decision node *Newspaper* added, that affects the forecast and Bob's annoyance level.



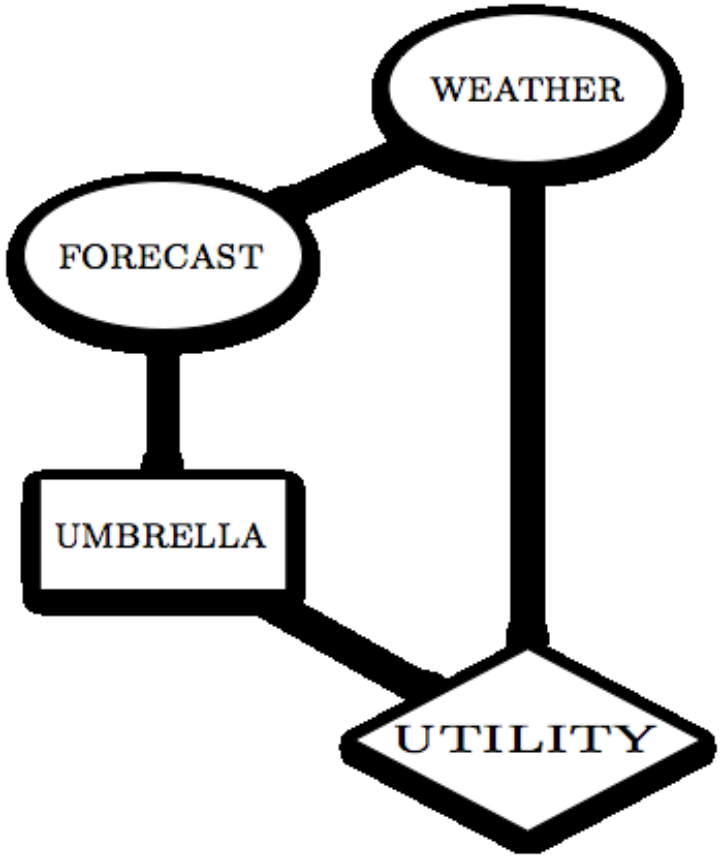
ID for Extended Umbrella Example

- "No forgetting" edges added from *Newspaper* to *Umbrella*.
- Agents remember their past decisions when they make future decisions.
- Information available to past decisions is also available to future decisions.

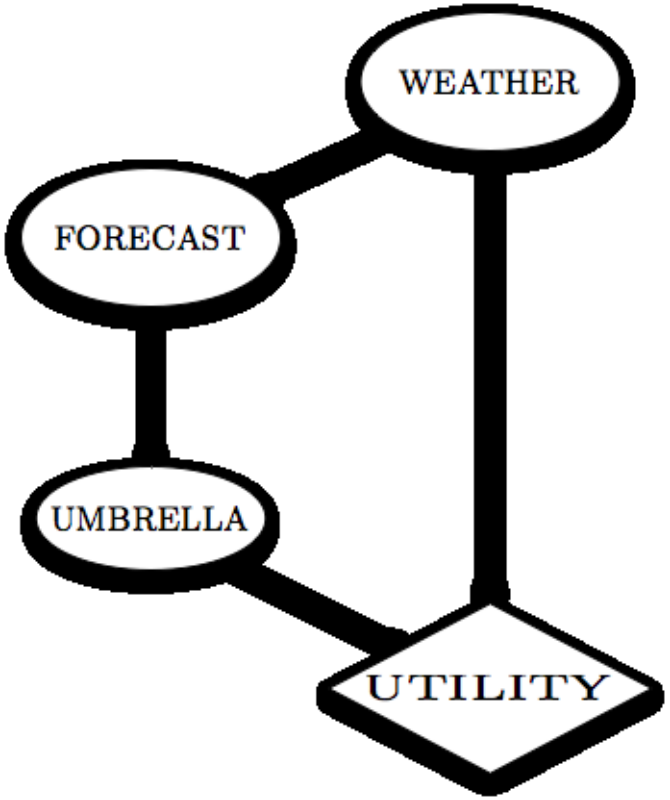


IDs and BNs

- A new chance node *implements* a strategy *s* for decision *D* if it has the same informational parents as *D* and chooses the same action as does *s* for each instantiation of the parents of D.
- A chance node implements a utility node *V* if it assigns probability 1 to the value associated with the utility node for each instantiation of the parents.



IDs and BNs



	UM	
F	*true*	*false*
sun	0	1
rain	1	0

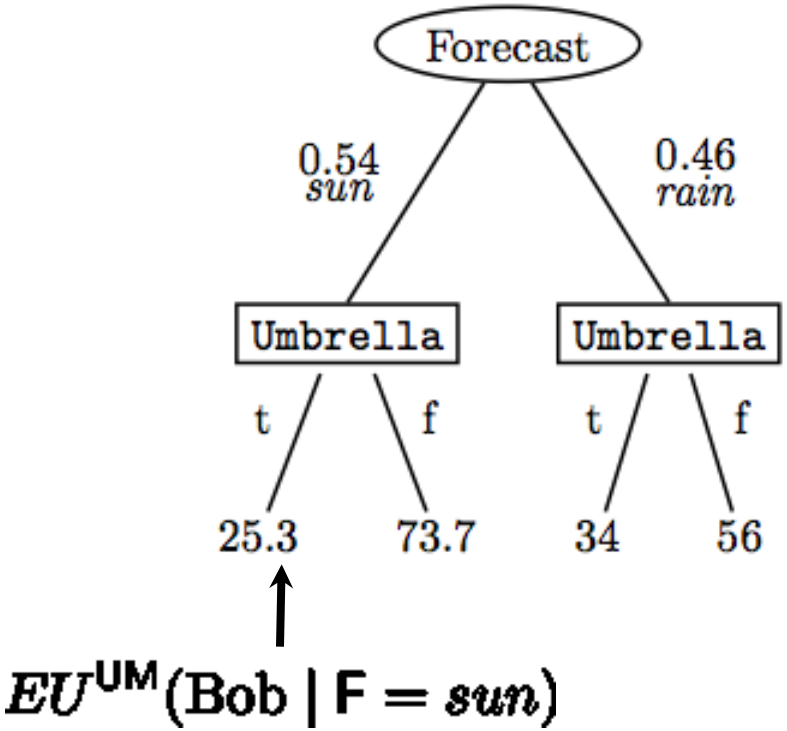
A BN implements an ID given strategies for all decisions if it implements all decisions and chance nodes.

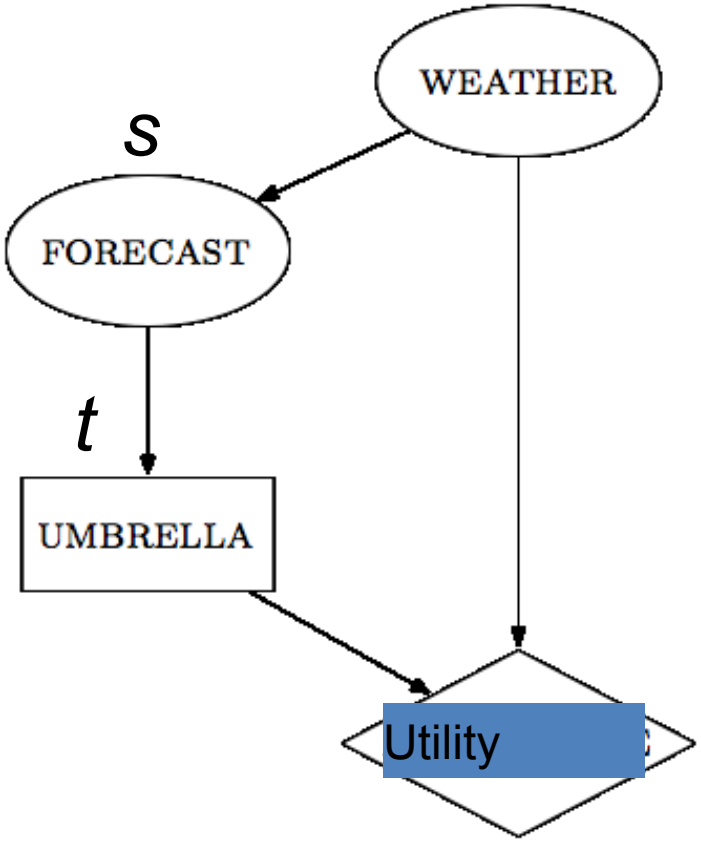
Converting IDs to Decision-trees

1. Traverse ID top-down.
2. If node is a decision or an informational parent
   - create vertex in decision tree.
   - create edges and label with node values.
3. Compute probability of each value of informational parents and annotate edge.
4. Label leaves with expected utility for agent given a path instantiating values for all decisions and informational parents.

Steps 3 and 4 require to query Bayesian network

Converting ID to Decision Tree: Umbrella Example





Disadvantage : Lose the graph structure

Solving IDs

- Use backward induction.
- For each decision D, from bottom up:
- Assume that decisions later than D implement their optimal strategy.
- For each value **v** of the decision node parents of D, and each value **w** of the chance node parents of D:
  - For each action d of D
    - Implement the decision node parents of D with a deterministic strategy that always plays **v**
    - Implement D with a deterministic strategy that always plays d
    - Compute expected utility given **w**
      - This is a BN query
  - Given **v**, **w**, choose the action that maximizes this expected utility
- Implement the optimal strategy for D.

Acknowledgements

The decision theory material is drawn from a tutorial by Gal and Pfeffer at AAAI ‘08.