Graphical Models for Decision Making

Lesson 2, week 12, class 24

Juan A Fdez del Pozo (CIG-UPM)

December 2, 2015

Contens

1 Decision Tables and Decision Trees

Outline of Contens

1 Decision Tables and Decision Trees

Outline of Contens

1. Decision Tables and Decision Trees
   - Decision Tables
   - Decision Trees

Graphical Models for Decision Making

under uncertainty

B. Russell
"What men really want is not knowledge but certainty"

Decision Tables: definition

- Most elementary way of representation
- Illustrate simply the basic concepts
- Elements: $decision(a) + state(\theta) \rightarrow consequence(c(a, \theta) \in C)$

Decision Tables: **Example**: software development

	0.5 θ_1 =price up	0.3 θ_1 =price stay	0.2 θ_1 =price down
a_1= new staff	$u(a_1, \theta_1)$ =90	$u(a_1, \theta_2)$ =30	$u(a_1, \theta_3)$ =-50
a_2= try current	$u(a_2, \theta_1)$ =50	$u(a_2, \theta_2)$ =10	$u(a_2, \theta_3)$ =-20
a_3= not to expand	$u(a_3, \theta_1)$ =0	$u(a_3, \theta_2)$ =0	$u(a_3, \theta_3)$ =0

Decision Tables: evaluation

Discrete case (A and Θ discrete)

- Assign beliefs: $p(\theta_j) = p_j, (p_j \geq 0, \sum p_j = 1)$
- Assign preferences: $u(a_i, \theta_j)$
- Compute expeected utiltiy of each alternative a_i

$$\sum p_j * u(a_i, \theta_j) = Eu(a_i)$$

- Decision of maximun expected utitlity a^*

$$maxEu(a_i = Eu(a^*))$$

Continuos case:

$$Eu(a) = \int u(a, \theta) dP(\theta)$$

, a non linear programming problem

Decision Tables: evaluation

Example: software development

	0.5 θ_1 =price up	0.3 θ_1 =price stay	0.2 θ_1 =price down	$Eu(a_i)$
a_1= new staff	$u(a_1, \theta_1)$ =90	$u(a_1, \theta_2)$ =30	$u(a_1, \theta_3)$ =-50	**44**
a_2= try current	$u(a_2, \theta_1)$ =50	$u(a_2, \theta_2)$ =10	$u(a_2, \theta_3)$ =-20	24
a_3= not to expand	$u(a_3, \theta_1)$ =0	$u(a_3, \theta_2)$ =0	$u(a_3, \theta_3)$ =0	0

Outline of Contens

1. Decision Tables and Decision Trees
   - Decision Tables
   - Decision Trees

Decision Trees: definition

- In practice tables very limited: only 1 moment to make decisions
- Typically, real problems are dynamic, with chained decisions and alternating different outcomes
- Elements :
  - Decision (square), alternatives
  - Chance (circle), states
  - Terminal (end path), Utility of outcome of the scenario

Decision Trees: construction

- Skeleton (structure): from left to right (time passes), from root node to terminal nodes –Qualitative info–
- Relationships between elements and DM judgments via probabilities and utilities –Quantitative info–

Decision Trees: construction

Conditional probabilities given that events to the left have already happened and previous decisions have been made

			up	0.5	90
	a_1	P	–	0.3	30
	(44)		down	0.2	-50
			up	0.5	50
D	a_2	P	–	0.3	10
	(24)		down	0.2	-20
			up	0.5	0
	a_3	P	–	0.3	0
	(0)		down	0.2	0

Software problem, max(u).

Decision Trees: evaluation

- Tree= set of strategies or policies indicating action plans on what to do at each decision point reached with that plan
- Decision tree is evaluated to find an optimal strategy
- Backward dynamic programming method (optimality principle, Bellman & Dreyfus'62): assume we've made some decisions and nature has taken some outcomes, we'll have arrive at a node that it's:
  - terminal $\rightarrow$ and we assign it the utility of the consequence;
  - chance node $\rightarrow$ the max EU, from that node;
  - decision node $\rightarrow$ the EU of the decision with max EU, from that node

Example: Helicopter (structure, quantitative info, evaluation).

Decision Trees: properties

- Expressive, intuitive: complete paths
- Doesn't separate quantitative and qualitative info
- Exponential - growth $\rightarrow$ computational requirements too (since algorithm enumerates when passing through all the paths)
- Schematic trees (generalization)
- Symmetric: all scenarios with same variables and in the same sequence
- Can handle asymmetric problems
- Helicopter: 2 asymmetries: test outcome only if the test is conducted; helicopter state only if it's bought
- Inadequate representation of probabilistic$\lhd$ relationships (ad hoc)
- Preprocess of probabilities for representing the tree (2 trees problem)
- Non local Computations (joint distribution), $P(A, B, C) = P(A)P(B|A)P(C|A, B)$
- Coalescence (subtree replica) detected ad hoc
- Explicit information constraints, but requires total ordering (not always specified)

¿Remarks and Questions?

IDA 2015

12/01/2015, CIG