Implementing Uncertainty in a Logic Programming Framework

Trevor Martin,

AI Group, University of Bristol, UK
(Senior Research Fellow, BT Intelligent Systems Lab)

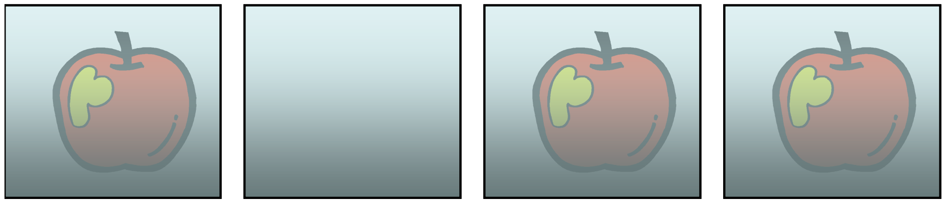
Trevor.Martin@bristol.ac.uk

{ University of BRISTOL / 0.2 , BT / 0.8 }

How should we handle uncertainty?

- mathematician : probability
  - model (subjective) uncertainty by gambling
  - anyone who does not follow the laws of probability is guaranteed to make a loss (Dutch book)

3 out of 4 boxes contain an apple



Option 1

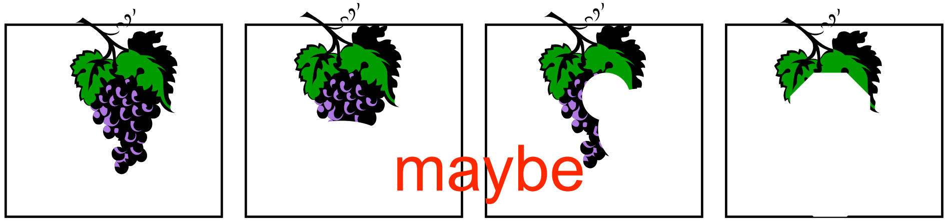
Play as many times as you like.
The (opaque) boxes are shuffled.
You pay £1 and choose a box.
If the box contains an apple, you win £1-50

Option 2

Same but if the box contains an apple, you win £1-05

Problems with Propositions

how many of these boxes contain a bunch of grapes ?



- sorites paradox : take one grain from a heap of sand and you still have a heap of sand. One grain is not a heap; adding another grain is still not a heap
- similarly - what is meant by
  - checkout time is 11:00
  - speed limit is 30mph

In between the black and white cases, there are shades of grey

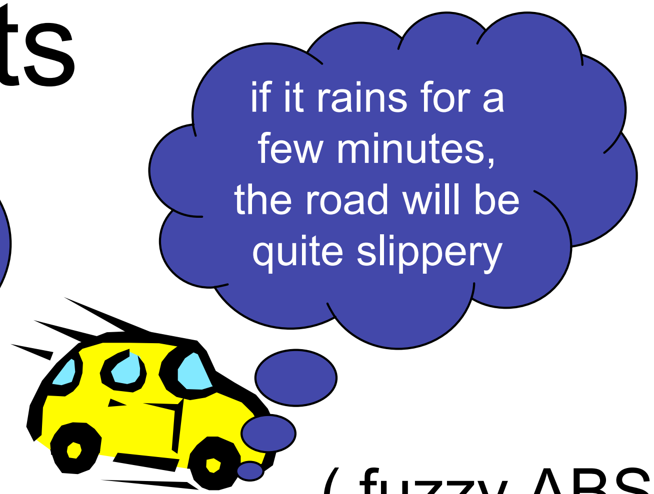
Natural Language

- words mean what we agree that they mean
  - *wicked, cool, bling, chav, rubbish*
  - light snowfall, bright colour, rock music
  - what makes a White Christmas? "*That one flake of snow will fall on Met office monitoring stations over the 24 hr period of the 25th of December*"

    www.mybetting.co.uk/white-christmas-betting.htm

- communication is made more efficient by use of loose definitions
- over-precision is not user-friendly
  - should we adapt our thinking to the computer *or*
  - adapt the computer to us

Fuzzy Sets



(fuzzy ABS)

drizzle = *"uniform precipitation composed exclusively of fine drops with diameters of less than 0.02 inch (0.5 mm) very close together"*

a few minutes = *2-10 minutes*
(www.daml.org)

- in mathematics we have precisely defined terms
  - a set has a characteristic function $\chi : U \rightarrow \{0, 1\}$
- in human language, most terms are defined by use
  - a fuzzy set has a characteristic function $\chi : U \rightarrow [0, 1]$
  - it indicates the degree to which an object has some property
  - more generally $\chi : U \rightarrow L$ where L is a lattice
  - *X* is fuzzy if an object can be *very X, slightly X, etc*
  - fuzzy can be related to probability via random sets / mass assignment

$U=\{1, 2, 3, 4, 5, 6\}$

x	$\chi_{even}(x)$	$\chi_{small}(x)$
1	0	1
2	1	0.7
3	0	0.3
4	1	0
5	0	0
6	1	0

Mass Assignments

mass assignment

distribution over power set

fuzzy set

(family of) distributions

(prior)

least prejudiced distribution

enables deduction, defuzzification,
partial matching (semantic unification)
e.g. Pr(x is *small* | x is *medium)*

(cf random sets, basic probability assignment)

- voting model : interpret fuzzy set membership as *the proportion of voters (possible worlds) who agree that a value satisfies a fuzzy label*
  - *small = {1/1, 2/0.7 3/0.3}*
  - 30% accept {1, 2, 3} as *small*, 40% accept {1, 2}, 30% accept {1}
  - least prejudiced distribution 1 : 0.6, 2 : 0.3, 3 : 0.1

- MA references : Google ← baldwin AND “mass assignment”

Uncertain Logic Programming

- Many approaches - typically using *numerical* uncertainty
  - implemented systems e.g. Fril, fuzzy Prolog, Fprolog, f-Prolog, Prolog-ELF, …
  - theoretical studies e.g. van Emden, Kifer & Subrahmanian, Vojtas, Lukasiewicz, Damasio & Pereira

- Support logic uses numerical uncertainty - voting model

luxuryCar(rolls-royce)	everyone agrees
luxuryCar(jaguar) : 0.9	9 out of 10 agree
likes(John, Jill) : (0.7 0.8)	7/10 yes, 1/10 abstain, 2/10 no

highMaintenanceCost(X) :- luxuryCar(X), oldCar(X) : 0.9

highMaintenanceCost(X) :- exRentalCar(X), highMileage(X) : (0.8 1)

- what about deduction ?

Support Logic rules

- Probabilistically quantified rules

A if B and C : S1 S2 ← interval containing pr(conclusion | ¬conditions)

conclusion conditions interval containing pr(conclusion | conditions)

((performance of company X is good in YEAR)
(turnover of company X in YEAR is *HighTurnover)*
(profit of company X in YEAR is *TenToTwentyPC*)) : ((0.8 1) (0 1))

A if B and C : S1
A if B and ¬C : S2 **General (extended) rule**
A if ¬B and C : S3
A if ¬B and ¬C : S4

evidential logic uses weighted combination of features

consider basic rule (top)
default conjunction is (interval) product
(**calculation of supports is more complex for general rule)**

Basic Fril rule - examples

((blond X) (swede X)) : (0.9 0.9)
((fairSkinned X) (blond X)) : (0.8 0.95)
((swede Bjorn))

((fairSkinned Bjorn)) : (0.72, 0.955)

There are between 80% and 95% of blond people who are fair

Pr(blond Bjorn) = 0.9

Pr(fairSkinned Bjorn) = 0.9x where $x \in [0.8, 0.95]$

((blond X) (swede X)) : (0.9 0.9)
((fairSkinned X) (blond X)) : ((0.8 0.95) (0 0.3))
((swede Bjorn)) : (0.9 1)

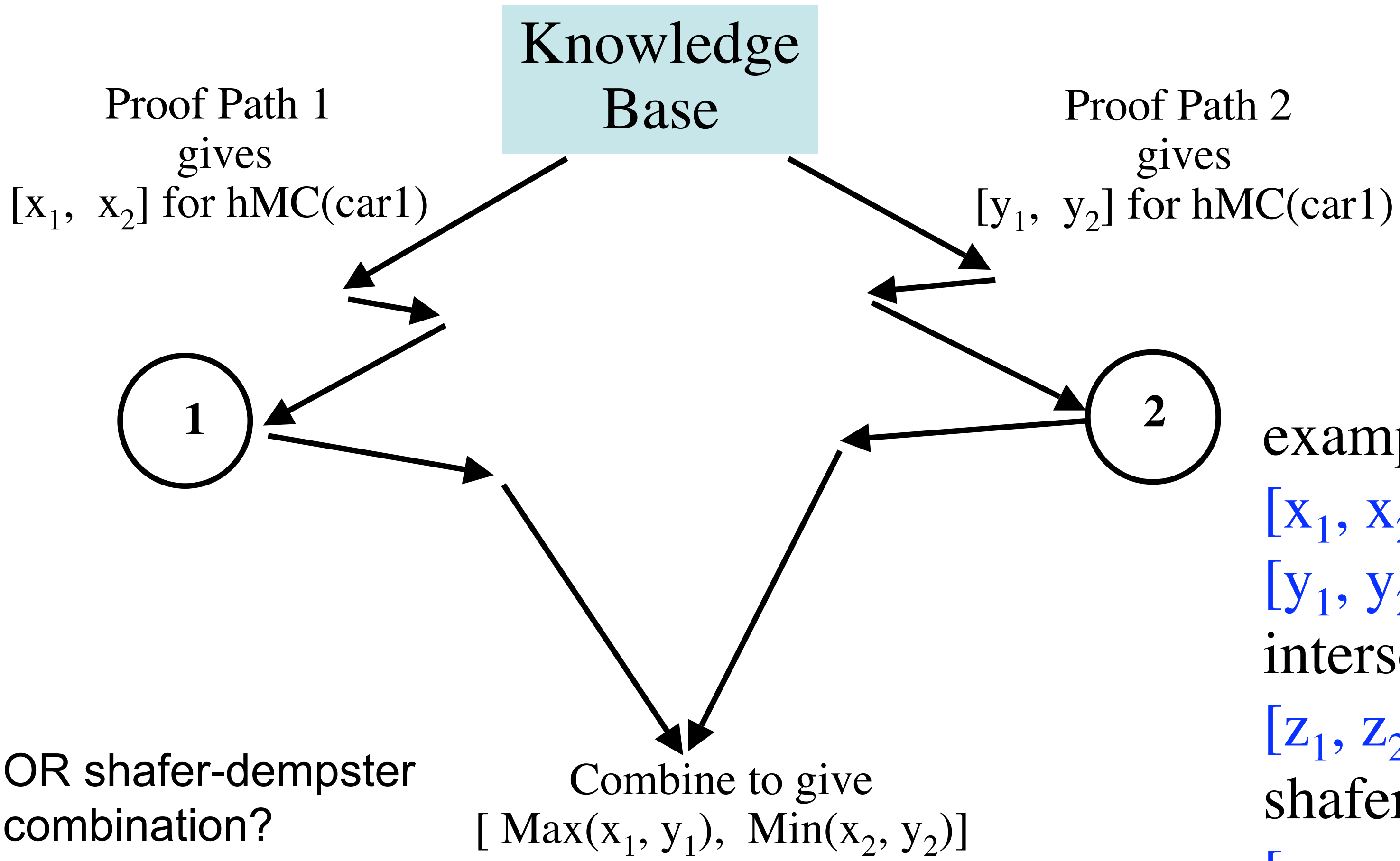
((fairSkinned Bjorn)) : (0.648 0.8915)

our belief that Bjorn is a Swede

Standard logic program with support calculated on proof path

Multiple proof paths

highMaintenanceCost(X) :- luxuryCar(X), oldCar(X) : 0.9
highMaintenanceCost(X) :- exRentalCar(X), highMileage(X) : (0.8 1)



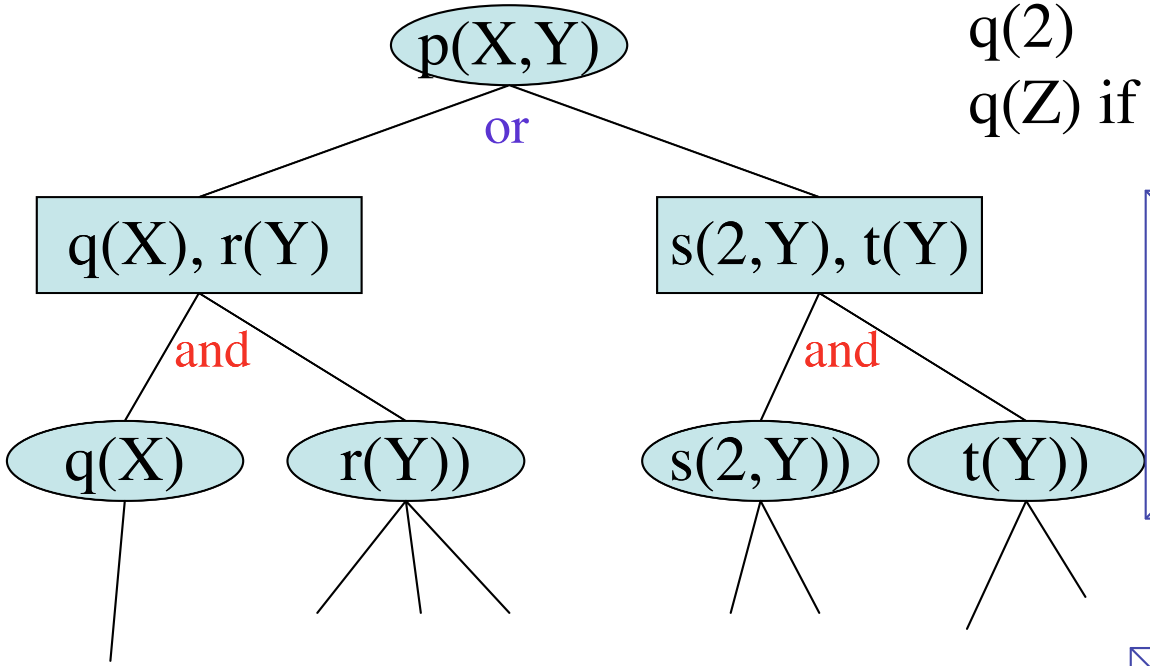
OR shafer-dempster combination?

example
$[x_1, x_2] = [0.3\ 0.9]$
$[y_1, y_2] = [0\ \ 0.5]$
intersection
$[z_1, z_2] = [0.3\ 0.5]$
shafer-dempster
$[z_1, z_2] = [0.18\ 0.53]]$

Execution of logic programs

p(*A*,*B*) if q(*A*), r(*B*)
p(*1*, *C*) if s(2, *C*), t(*C*)
q(1)
q(2)
q(Z) if t(Z) …

logic program = and/or tree



etc

depth search

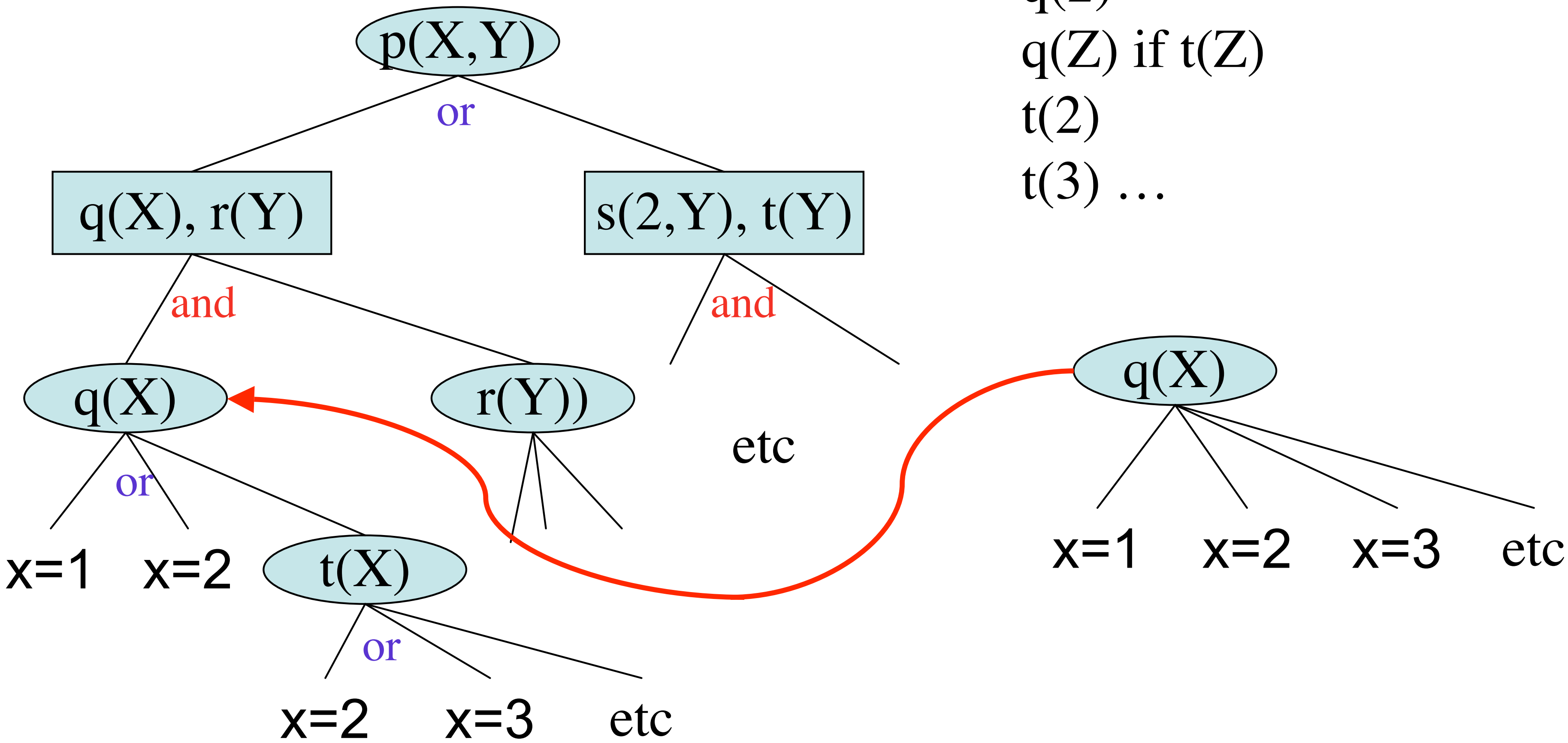
support logic program
= logic program
+ calculation of support

breadth search
- all solutions at *or* nodes

Support : replace each *or*-node

p(*A*,*B*) if q(*A*), r(*B*)
p(*1*, *C*) if s(*2*, *C*), t(*C*)
q(1)
q(2)
q(Z) if t(Z)
t(2)
t(3) …

logic program = and/or tree



Fril Abstract Machine Code

p1:	
set_mode inter	((p A B)(q A)(r B)) : ((.9 1)(0 .1))
try_me_else fail	creates choicepoint, support frame
allocate 2	saves environment, continuation
push_support ((.9 1)(0 .1)) <cond>	fills support frame
call q, 1, 2	
put_var A1, Y2	puts variable 2 (B) in reg 1
deallocate	reset continuation, discard env
execute r, 1	
q1:	
set_mode inter	((q a)) : (.85 1)
try_me_else q2	create choicepoint, support frame
get_const A1, a	unify a with argument 1
push_support (.85 1) <conj>	fills support frame
proceed	evaluate support stack
q2 :	((q Z)(s Z)) : ((.6 .9)(.3 .5))
trust_me_else fail	
push_support ((.6 .9)(.3 .5)) <cond>	
execute s	

warren machine (Prolog) extended by support ops

Support operations

- *conj* - overall support for rule body (conjunction) from individual goals (product)

  supp(q(A), r(B)) = *conj*(supp(q(A)), supp(r(B))

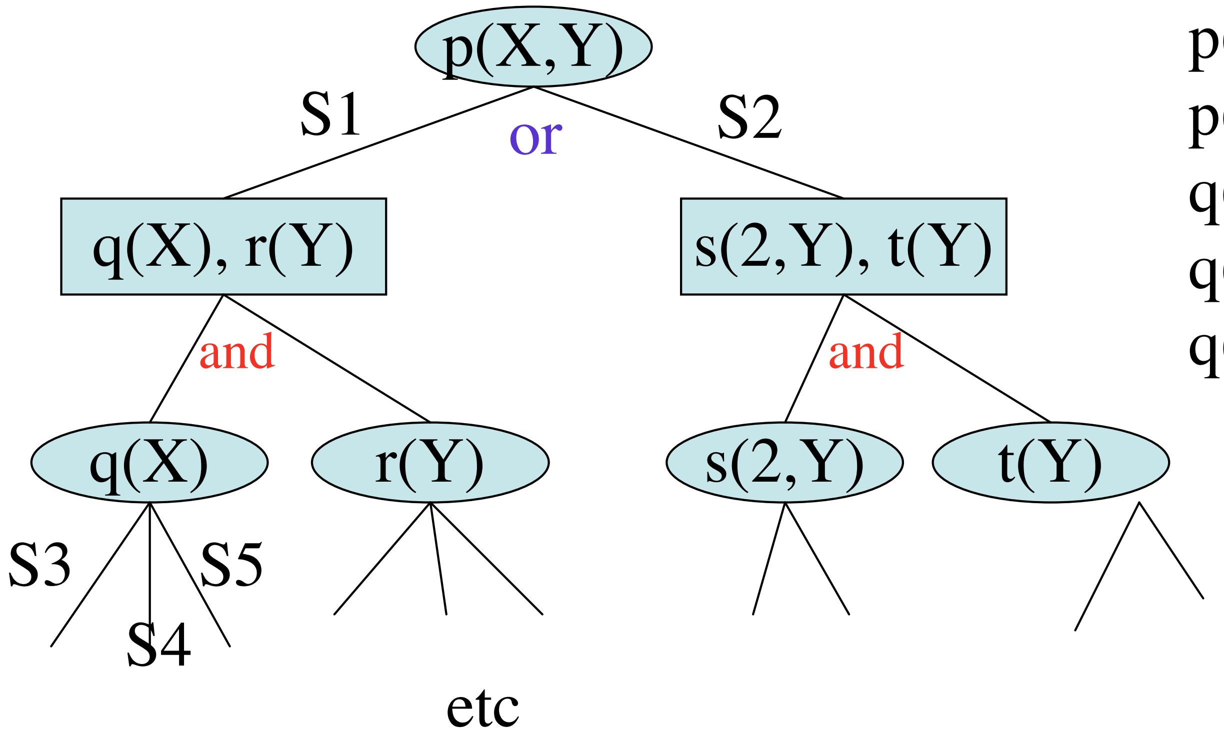
- *cond* - support for rule head from rule body and rule (conditional) support

  supp(head) = *cond*(supp(rule)), supp(body))

- *comb* - support for conclusion from multiple paths (intersection)

  supp(conc) = *comb*(supp(path1)), supp(path2))

Calculation of support



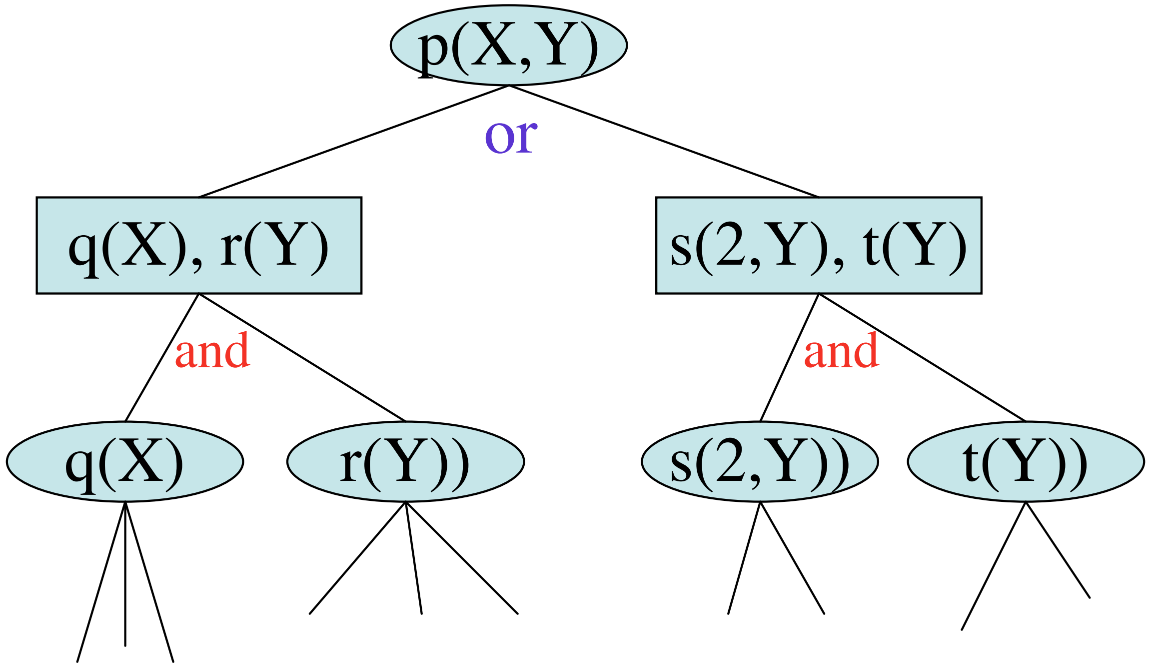
p(A,B) if q(A), r(B) : S1
p(1, C) if s(2, C), t(C) : S2
q(1) : S3
q(2) : S4
q(Z) if t(Z) : S5 …

supp(p(1, 2)) = *comb* (*cond*(S1, *conj* (supp(q(1)), supp(r(2))),
cond(S2, *conj* (supp(s(2, 2)), supp(t(2))))

supp(q(1)) = *comb* (*S3*,
cond(S5, *conj* (supp(t(1))))

Execution of support logic programs

logic program = and/or tree



etc

depth search

p(A,B) if q(A), r(B) : S1
p(1, C) if s(2, C), t(C) : S2
q(1) : S3
q(2) : S4
q(Z) if t(Z) : S5 …

support logic program
= logic program
+ calculation of support

exhaustive depth search
(+ support calculation)

Transformation of SLP $\Rightarrow$ LP

p(A,B) if q(A), r(B) : S1
p(1, C) if s(2, C), t(C) : S2
q(1) : S3
q(2) : S4
q(Z) if t(Z) : S5 …

p(*cond*(S1, *conj*(S_q,S_r)), A, B) if q(S_q,A), r(S_r,B)
p(*cond*(…), 1, C) if s(*Ss*, 2, C), t(*St*, C)
q(*S3*, 1)
q(*S4*, 2)
q(*cond*(…), Z) if t(Z)
…

replace

$$Supp(head) = cond\left(S_r, \underset{i=1\ldots n}{conj}\left(\underset{j=1\ldots ki}{comb}\left(S_i^j\right)\right)\right)$$

with

$$Supp(head) = comb\left(cond\left(S_r, \underset{\substack{i=1\ldots n\\ j=1\ldots ki}}{conj}\left(S_i^j\right)\right)\right)$$

because (e.g)

$$conj\left(comb\left(S_1,S_2\right),S_3\right)$$

$$= comb\left(conj\left(S_1,S_3\right),conj\left(S_2,S_3\right)\right)$$

Is it a real problem?

two examples

Cooking without butter - dairy-free spread

Bakery
Beers, Wines, Spirits
Beverages, Hot Drinks
Breakfast Cereals
Clothing
Confectionery, Biscuits, Cakes
Cooking/Baking Ingredients
Crisps, Nuts, Snacks
Dairy
Delicatessen
Easter Confectionery
…
Pickles, Preserves, Oils, Spreads → No luck …
…

Animal and Vegetable Fats → Cadbury's Mini Eggs Nest, 190
Lard 250g
Artificial Sweeteners
Colouring and Decoration
Custard and Cornflour
Dried Fruit
Flour - Other
…

Cheese - American
Cheese - Canadian
…
Cheese - Snacking
…
Spreads (Butter, Margarine, etc.) → Dairy Free Spread 500g
…
Yogurt - Twin Pots

dairy-free classified as *dairy*. Not logic! But it works.

www.sainsburystoyou.com

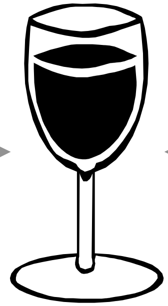
Wine example - conflicting definitions

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource =
            "&food;PotableLiquid"/>
  <rdfs:label xml:lang="en">wine</rdfs:label>
  <rdfs:label xml:lang="fr">vin</rdfs:label>
        … … …
</owl:Class>
```

wine

- *wine* is a subclass of *potable liquid*
- *wine* -> (madeFromGrape) -> *wineGrape* (at least one)
- *vintage wine* is made from *wineGrapes* harvested in a *single year*

French supplier using French ontology



classed as vintage wine in UK?

US supplier using US ontology

- *US regulations : a vintage wine* is *wine* made from *wineGrapes* at least 95 % of which were harvested in a *single year*

Single ontologies *may* be crisp.
Combined (multiple) ontologies are very unlikely to be crisp

Modelling with support logic

- first case can be expressed *within* existing frameworks

  combinedOnt:vintageWine(X) ← Fr: vintageWine(X)

  – a straightforward logic programming rule

- what about

  Pr(*combinedOnt* : *vintageWine* | *US*: *vintageWine*) ∈ [0.9, 1]

  combinedOnt : vintageWine(X) ← US: vintageWine(X) : (0.9 1)

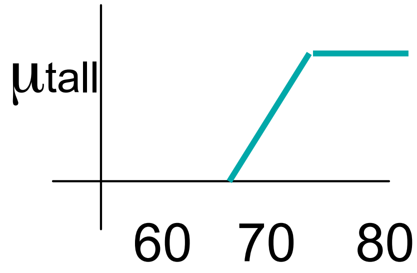
  (support logic program)

  – not expressible in RuleML / SWRL / logic program

Fuzzy attribute values

- Support logic also allows uncertainty in attribute values

  height(John, *tall*)
  height(Bill, 72)
  maintenanceCost(X, *high*) :- luxuryCar(X), age(X, *old*)



- interpretation - single value, but not known precisely
  - e.g. contrast
    - safe-speed on an open highway is *about-80mph*
    - current-speed of car-1 is *about-80mph*

  safe-speed(open-highway, 60) : 0.1
  safe-speed(open-highway, 80) : 1
  …
  safe-speed(open-highway, 90) : 0.2

  current-speed(car-1, {60/0.1 … 80/1 … 90/0.2}

- how do we unify a query with a clause head — height(X, *medium)* / height(John, *tall*)

height(X, *medium) IF* height(X, *tall*) : Pr(*medium* | *tall*)

Semantic Unification

e.g. Dice values

- *small* = {1/1, 2/0.7 3/0.3}
- mass assignment is {1, 2, 3} : 0.3, {1, 2} : 0.4 {1} : 0.3
- *about2* = {1/0.4, 2/1 3/0.4} , MA = {1} : 0.6, {1, 2, 3} : 0.4
- Pr (*about2* | *small*) $\in$ [0.4, 0.82]
- point value can be calculated if a prior is known

Same method for crisp value given fuzzy set and vice-versa

about2 | small	{2} : 0.6	{1, 2, 3} : 0.4
{1} : 0.3	F 0.3×0.6	T 0.3×0.4
{1, 2} : 0.4	U 0.4×0.6	T 0.4×0.4
{1, 2, 3} : 0.3	U 0.3×0.6	T 0.3×0.4

This requires modifications to the logic programming unification method, and so is not easily embedded in a conventional framework

Summary - transformed support logic

- restricted form of rules and operators
  - represent *attribute uncertainty* via *predicates*
- embed support as *argument* to predicates

- standard execution model (efficiency)
  - evaluate support when query is completed

- no conflict with “crisp” standards
- no conflict with logic program semantics
  - but rule/fact uncertainty is now implicit, not explicit
  - object level vs meta-level

Summary - the need for fuzziness

- Machine-based solutions must be understandable
  - humans use natural language - machines cannot understand NL but can process it
  - engineering approach - be consistent with (fuzzy) humans
  - soft methods are vital because most relations / categories / attributes / … are not defined by unbreakable rules, data can be missing, inconsistent, unreliable, …
- a useful semantic web needs to be fuzzy
  - meta-data comes from
    - humans (subjective) or
    - machines (cannot guarantee correctness)
- we expect multiple sources to be (slightly) inconsistent
  - logic says anything follows from inconsistency
  - most rules are not true all the time
  - humans manage, so should machines

Thank you for your attention

Questions …

Comments …

acknowledgments : slides 9 and 10 adapted from originals by jim baldwin
picture of Nicholas Cage (the Weatherman) taken from www.imdb.com/
The organisation of www.sainsburystoyou.com has changed recently but still includes non-dairy spread in a "dairy" category