Limit expressive power?

Defaults, probabilities, *etc.* can all be thought of as extensions to FOL, with obvious applications

Why not strive for the *union* of all such extensions?

a language co-extensive with English?

Problem: automated reasoning

Lesson here:

reasoning procedures required for more expressive languages may not work very well in practice

Tradeoff: expressiveness vs. tractability

Overview:

- a Description Logic example
- limited languages
- the problem with cases
- vivid reasoning as an extreme case
- less vivid reasoning
- hybrid reasoning systems

Simple Description Logic

Consider the language FL defined by:

<concept> ::= atom
| (AND <concept> ... <concept>)
| (ALL <role> <concept>)
| (SOME <role>)

<role> ::= atom
| (RESTR <role> <concept>)

Example:

- (ALL child (AND FEMALE STUDENT))

  an individual whose children are female students

- (ALL (RESTR child FEMALE) STUDENT)

  an individual whose female children are students

  there may or may not be male children and they may or may not be students

Extension functions as before with

$\Phi[(\text{RESTR } r\ c)] =$
$\{ (x,y) \mid (x,y) \in \Phi[r] \text{ and } y \in \Phi[c] \}$

Subsumption defined as usual

Computing subsumption

First for FL$^-$ = FL without the `RESTR` operator

- put the concepts into normalized form

  (`AND` p_1 ... p_k
  (`SOME` r_1) ... (`SOME` r_m)
  (`ALL` s_1 c_1) ... (`ALL` s_n c_n))

- to see if C subsumes D make sure that
  1. for every $p \in C$, $p \in D$
  2. for every (`SOME` r) $\in C$, (`SOME` r) $\in D$
  3. for every (`ALL` s c) $\in C$, find an (`ALL` s d) $\in D$ such that c subsumes d.

Can prove that this method is sound and complete relative to definition based on extension functions

Running time:

- normalization is $O(n^2)$
- structural matching:

  for each part of C, find a part of D:

  again $O(n^2)$

What about all of FL, including `RESTR`?

Subsumption in FL

Not so easy:

- cannot settle for part-by-part matching

  (ALL (RESTR friend (AND MALE DOCTOR))
  (AND TALL RICH))

  subsumes

  (AND (ALL (RESTR friend MALE)
  (AND TALL HAPPY))
  (ALL (RESTR friend DOCTOR)
  (AND RICH SURGEON)))

- complex interactions

  (SOME (RESTR r (AND a b)))

  subsumes

  (AND (SOME (RESTR r (AND c d)))
  (ALL (RESTR r c) (AND a e))
  (ALL (RESTR r (AND d e)) b))

In general: can prove that FL is powerful enough to encode *all* of propositional logic

there is a mapping Ω from CNF wffs to FL where

$\models (\alpha \supset \beta)$ iff $\Omega[\alpha]$ is subsumed by $\Omega[\beta]$

but $\models (\alpha \supset (p \wedge \neg p))$ iff α is unsatisfiable

Conclusion: there is no good algorithm for FL

unless P=NP

Moral

Even small doses of expressive power come at a computational price

Questions:

- what properties of a representation language control its difficulty?
- how far can expressiveness be pushed without losing good algorithms
- when is easy reasoning adequate for KR purposes?

These questions remain unanswered, but some progress has been made

- need for case analyses is a major factor
- tradeoff for DL languages is reasonably well understood
- best addressed (perhaps) by looking at working systems

Approach:

- find reasoning tasks that are tractable
- analyze difficulty in extending them

Limited languages

Some reasoning problems that can be formulated in terms of FOL entailment

$$\text{KB} \overset{?}{\models} \alpha$$

admit very specialized methods because of the restricted form of either KB or α

although problem could be solved using full resolution theorem proving, there is no need

Example 1: Horn clauses

- SLD resolution provides more focussed search
- in propositional case, a linear procedure is available

Example 2: Description logics

- Can do DL subsumption using Resolution

  Introduce predicate symbols for concepts, and "meaning postulates" like

  $$\forall x[P(x) \equiv \forall y(\text{Friend}(x,y) \supset \text{Rich}(y)) \wedge \forall y(\text{Child}(x,y) \supset \forall z(\text{Friend}(y,z) \supset \text{Happy}(z)))]$$

  for (AND (ALL friend RICH) (ALL child (ALL friend HAPPY)))

  Then ask if MP $\models \forall x[P(x) \supset Q(x)]$

Equations

Example 3: linear equations

Let E be the usual axioms for arithmetic

$\forall x \forall y(x+y = y+x), \ \forall x(x+0 = x), ...$ Peano axioms

Then have the following:

$E \models (x+2y=4 \wedge x-y=1) \supset (x=2 \wedge y=1)$

Can “solve” linear equations using Resolution

But there is a much better way:

Gauss-Jordan method with back substitution

- subtract (2) from (1): $3y = 3$
- divide by 3: $y = 1$
- substitute in (1): $x = 2$

In general, a set of linear equations can be solved in $O(n^3)$ operations

This idea obviously generalizes!

always advantageous to use a specialized procedure when it is available, rather than a general method like Resolution

When is reasoning hard?

Suppose that instead of a set of linear equations, we have something like

$(x+2y=4 \vee 3x-y=7) \wedge x-y=1$

Can still show using Resolution: $y > 0$

To use GJ method, we need to split cases:

$x+2y=4 \wedge x-y=1$ ß $y=1$ $\therefore y > 0$

$3x-y=7 \wedge x-y=1$ ß $y=2$

What if 2 disjunctions?

$(eqnA_1 \vee eqnB_1) \wedge (eqnA_2 \vee eqnB_2)$

there are *four* cases to consider with GJ method

What if n binary disjunctions?

$(eqnA_1 \vee eqnB_1) \wedge ... \wedge (eqnA_n \vee eqnB_n)$

there are 2^n cases to consider with GJ method

with $n=30$, would need to solve 10^9 systems of equations!

Conclusion: even assuming a very efficient method, case analysis is still a *big* problem

Question: can we avoid case analyses??

Expressiveness of FOL

Ability to represent incomplete knowledge

- $P(a) \lor P(b)$ — but which?
- $\exists x\, P(x)$ — $P(a) \lor P(b) \lor P(c) \lor ...$
- and even
- $c \neq 3$ — $c=1 \lor c=2 \lor c=4 \lor ...$

Reasoning with facts like these requires somehow "covering" all the implicit cases

- languages that admit efficient reasoning do not allow this type of knowledge to be represented
  - e.g. Horn clauses, description logics, linear equations, ...

One way to ensure tractability:

- somehow restrict contents of KB so that reasoning by cases is not required

But is complete knowledge enough for tractability?

- suppose KB $\models \alpha$ or KB $\models \neg\alpha$, as in the CWA
  - Get: queries reduce to KB $\models \lambda$, literals
  - But: it can still be hard to answer for literals
- example: KB = $\{(p \lor q), (\neg p \lor q), (\neg p \lor \neg q)\}$
  - Have: KB $\models \neg p \land q$ — complete
  - even literals may require case analysis

Vivid knowledge

Note: If KB is complete and consistent, then it is satisfied by a *unique* interpretation I

- Why? define I by $I \models p$ iff KB $\models p$ — ignoring quantifiers for now
- Then for any I^*, if $I^* \models$ KB then I^* agrees with I on all atomic sentences p

Get: KB $\models \alpha$ iff $I \models \alpha$

- entailments of KB are sentences that are true at I
- explains why queries reduce to atomic case
  - $(\alpha \lor \beta)$ is true iff α is true or β is true, *etc.*
- if we have the I, we can easily determine what is or is not entailed

Problem: KB can be complete and consistent, but unique interpretation may be hard to find

- as in the type of example on the previous slide
- want a KB that wears this unique interpretation on its sleeve

Solution: a KB is vivid if it is a complete and consistent set of literals (for some language)

- e.g. KB = $\{\neg p, q\}$ — specifies I directly

To answer queries need only use KB^+, the positive literals in KB, as in the CWA

Quantifiers

As with the CWA, we can generalize the notion of vivid to accommodate queries with quantifiers

A first-order KB is <u>vivid</u> iff for some finite set of positive function-free ground literals KB^+,

$KB = KB^+ \cup Negs \cup Dc \cup Un$

Get a simple recursive algorithm for $KB \models \alpha$:

$KB \models \exists x.\alpha$ iff $KB \models \alpha[x/c]$, for some $c \in KB^+$

$KB \models (\alpha \vee \beta)$ iff $KB \models \alpha$ or $KB \models \beta$

$KB \models \neg\alpha$ iff $KB \not\models \alpha$

$KB \models (c = d)$ iff c and d are the same constant

$KB \models p$ iff $p \in KB^+$

This is just database retrieval

useful to store KB^+ as a collection of relations

Note: only KB^+ is needed to answer queries, but *Negs, Dc,* and *Un* are required to *justify* procedure

KB and KB^+ are *not* logically equivalent

e.g. $KB^+ \models \lambda$ only if λ is positive

So: could generalize definition to have arbitrary sentences that entail *Negs, Dc,* and *Un*

Analogues

Can think of a vivid KB as an analogue of the world it is talking about

there is a 1-1 correspondence between

- objects in the world and constants in the KB^+
- relationships in the world and syntactic relationships in the KB^+

for example, if constants c_1 and c_2 stand for objects in the world o_1 and o_2

there is a relationship R holding between objects o_1 and o_2 in the world

iff

the constants c_1 and c_2 appear together as a tuple in the relation represented by R

Not true in general

for example, if KB = $\{P(a)\}$ then it only uses 1 constant, but could be talking about a world where there are 5 individuals of which 4 satisfy P

Result: certain operations are easy

- how many objects satisfy P (by counting)
- changes to the world (by changes to KB^+)

Beyond vivid

Requirement of vividness is very strict.

Would like to consider weaker alternatives with good reasoning properties

Extension 1

ignoring quantifiers again

Suppose KB is a finite set of literals

- not necessarily a complete set (no CWA)
- assume consistent, else trivial

Cannot reduce KB |= α to literal queries

for example, if KB = $\{p\}$
then KB |= $(p \wedge q \;\vee p \wedge \neg q)$
but KB $|\neq p \wedge q$ and KB $|\neq p \wedge \neg q$

But: assume α is small. Can put into CNF

α ß $(c_1 \wedge \ldots \wedge c_n)$

- KB |= α iff KB |= c_i, for every clause in CNF of α
- KB |= c iff c has complementary literals – tautology
  or KB $\cap$ c is not empty

Why?

Extension 2

Imagine KB vivid as before + new definitions:

$\forall xyz[R(x,y,z) \equiv$ ... wff in vivid language ...]

Example: have vivid KB using predicate ParentOf

add: $\forall xy[\mathrm{MotherOf}(x,y) \equiv \mathrm{ParentOf}(x,y) \wedge \mathrm{Female}(x)]$

To answer query containing $R(t_1,t_2,t_3)$, simply macro expand it with definition and continue

- can handle arbitrary logical operators in definition since they become part of *query*, not KB
- can generalize to handle predicates not only in vivid KB, provided that they bottom out to KB$^+$

$\forall xy[\mathrm{AncestorOf}(x,y) \equiv \mathrm{ParentOf}(x,y) \;\vee$
$\exists z\, \mathrm{ParentOf}(x,z) \wedge \mathrm{AncestorOf}(z,y)]$

- clear relation to Prolog

Others...

Vivification: given non-vivid KB, attempt to make vivid e.g. by eliminating disjunctions *etc.*

e.g. use defaults to choose between disjuncts

Problem: what to do with function symbols, when Herbrand universe is not finite?

partial Herbrand base?

Hybrid reasoning

Want to be able to incorporate into a single system special-purpose efficient reasoners

How can they coexist within a general scheme such as Resolution?

a variety of approaches for hybrid reasoners

Simple form: semantic attachment

- attach procedures to functions and predicates

  e.g. numbers: procedures on plus, LessThan, ...

- ground terms and atomic sentences can be *evaluated* prior to Resolution

  $P(\text{factorial}(4), \text{times}(2,3))$ ß $P(24, 6)$
  $\text{LessThan}(\text{quotient}(36,6), 5) \vee \alpha$ ß α

- much better than reasoning directly with axioms

More complex form: *theory resolution*

- build theory into unification process

  the way paramodulation builds in =

- extended notion of complementary literals

  $\{\alpha, \text{LessThan}(2,x)\}$ and $\{\text{LessThan}(x,1), \beta\}$
  resolve to $\{\alpha,\beta\}$

Using descriptions

Imagine that predicates are defined elsewhere as concepts in a description logic

Married ≡ (AND ...)   Bachelor ≡ (AT-MOST ...)

then want

$\{P(x), \text{Married}(x)\}$ and $\{\text{Bachelor}(\text{john}), Q(y)\}$

to resolve to

$\{P(\text{john}), Q(y)\}$

since the other two literals are contradictory for x=john, *given DL definitions*

Can use description logic procedure to decide if two predicates are complementary

instead of explicit meaning postulates

Residues: for "almost" complementary literals

$\{P(x), \text{Male}(x)\}$ and $\{\neg\text{Bachelor}(\text{john}), Q(y)\}$

resolve to

$\{P(\text{john}), Q(y), \text{Married}(\text{john})\}$

since the two literals are contradictory *unless* John is married

Main issue: completeness of theory resolution

- what resolvents are necessary to get the same conclusions as if meaning postulates were used
- residues are necessary for completeness