Combining Logic, Rules, and Ontologies, for Medical Decision Support

Terrance Swift
Medical Decision Logix, Inc

- Review proplositional logic, predicate logic, and Prolog
- Discuss differences between predicate logic and Prolog
- Discuss how these differences are overcome via Constraint Logic Programming
- Discuss how description logics bridge gap between ontologies and predicate logic
- Discuss how FLORA bridges gap between ontologies and rules
- Discuss how FLORA + constraints combines logic, rules and ontologies

Review: Propositional Logic

Atomic propositions and connectives $(\wedge, \neg)$

"if a patient has breast cancer, Doxorubicin and Tamoxifen are indicated"

might be translated as

$$
\begin{array}{l}
breast\_cancer \Rightarrow \\
\qquad\qquad (\\
\qquad\qquad\qquad doxorubicin\_indicated \\
\qquad\qquad \wedge \\
\qquad\qquad\qquad tamoxifen\_indicated \\
\qquad\qquad)
\end{array}
$$

Review: Predicate Logic

- Propositional connectives + quantifiers + n-ary relations + functions
- Variables range over elements of a universe, rather than over propositions
- Maps well to relational databases

"if a patient has breast cancer, Doxorubicin and Tamoxifen are indicated"

$$
\begin{aligned}
&\forall P.breast\_cancer(P) \Rightarrow \\
&\qquad (\\
&\qquad\qquad indicated(doxorubicin, P) \\
&\qquad \wedge \\
&\qquad\qquad indicated(tamoxifen, P) \\
&\qquad)
\end{aligned}
$$

Review: Prolog

- Logical rules with a proof theory based on unification and resolution
- Default negation
- Aggregation

"if a patient has breast cancer, Doxorubicin and Tamoxifen are indicated"

```
indicated(doxorubicin,P):-
           breast_cancer(P).

indicated(tamoxifen,P):-
           breast_cancer(P).
```

or

```
indicated_doxorubicin:- breast_cancer.

indicated_tamoxifen:- breast_cancer.
```

A little theory...

- We just saw three formalisms: Propositional Logic, Predicate Logic, Rules
  - A set of sentences in a logic is called a *theory*
  - A set of rules in Prolog is called a *program*
  - The meaning of either is based on a *structure*, a set of relations and functions over a set of elements.
  - If all sentences are true in the structure it is a *model* of the theory/program

A little theory...

- Predicate Logic: Godel's Incompleteness Theorem
  - For certain sentences T in a basic theory of arithmatic over the integers.
  - T is true in the standard model of the integers.
  - No proof theory of predicate logic can indicate whether T is true
  - No computer can answer whether T is true in our knowledge base.
  - So predicate logic may not always be a good basis for knowledge representation.
  - Predicate logic over the integers is an *undecidable theory*
  - Other theories in predicate logic may or may not be decidable.

A little theory...

What about rules?

- Church's Thesis: Any effective computational method is equivalent to a Turing Machine [1]
- Prolog is Turning-Complete, even without negation, aggregation, etc.
- Thus, anything you can "effectively" compute or effectively prove you can compute in Prolog! (... or Java ... or Lisp...)

This is great, but...

- Turing machines exhibit the halting problem on undecidable theories – they may or may not come up with an answer.

[1] Actually this is a restatement: Church said that any effectively computable set can be computed via a formalism for recursive functions – and Turning machines can compute all recursive functions.

A little theory...

But wait, there's more...

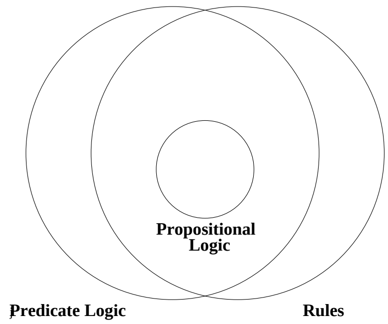
- There are also finite Prolog programs that cannot be finitely axiomatized in predicate logic

```
a(X,Y):- a(X,Z),a(Z,Y).
```

- This means there is no finite theory of predicate logic that

We have a mismatch

- Predicate logic has undecidable theories
  - No effective computation method for these theories
- Finite programs cannot be finitely axiomitized in predicate logic
  - Some programs have a meaning that is not effectively expressible first-order logic.



Propositional logic

- Decidable subset of predicate logic
- Terminating subset of Prolog rules (if you use XSB :-)
- Propositional satisfiability checkers
  - Solve problems from sudoku to scheduling to clustering in graphs [3]
  - In principle can solve convergence problems for belief networks (might not be pretty).

Default and Classical Negation

So far we haven't considered negation (or aggregation)

- Default negation can be seen as an assumption

Infer that a patient has XY chromosomes if he is a normal-looking mail. Of course he could have XYY chromosomes.

```
patient_has_XY :- not patient_has_XYY.
```

- Default negation is implemented as a failure to prove.
- Only minimal "models" are taken – models in which `patient_has_XY` and `patient_has_XYY` are not considered.

Default and Classical Negation

An incorrect logical formulation

$$patient\_has\_XY \Leftarrow \neg patient\_has\_XYY$$

allows a model in which $patient\_has\_XY$ and $patient\_has\_XYY$ are both true.

- Default negation may arise when an action needs to be taken without having all facts (e.g. medical workflow)
- Considerr a diagnostic criterion for Autistic Disorder from DSM-IV

The disturbance is not better accounted for by Rett's Disorder or Childhood Disintegrative Disorder.

This negation has a default character as well [4]

A Transition Slide

- Predicate Logic and Rules differ based on computability results
- Predicate Logic differs from Prolog (and other AI lanugages) based on negation.
- Differences in negation are not as fundamental — so are they easier to solve?

Constraint Logic Programming

Example:

"If a patient has breast cancer and is pregnant
then Doxyrubicin is contra-indicated"

How do you represent this in Prolog?

```
not indicated(doxyrubicin,P):-/* incorrect! */
           breast_cancer(P),pregnant(P).
```

Add a logical theory to your rules

```
indicated_with_constraints(Drug,Patient):-
     indicated(Drug,Patient),
     { ¬ (indicated(doxyrubicin,P) ∧
          breast_cancer(P) ∧ pregnant(P)) },
     { ¬ (indicated(doxyrubicin,P) ∧
          prev_history_of_doxyrubicin(P)) }.
```

Constraint Logic Programming

- Logical rules find partial solutions that may be fed to constraint solvers to check or refine these solutions.
- Above example assumes a propositional logic solver
- Many other solvers integrated with rules through CLP (cf. [2])
  - Linear in-equations over the reals (e.g. integer programming)
  - Finite domain constraints for combinatorial problems (like propositional solvers)
  - Belief Network Solvers [1]

And ... description logic solvers.

Description Logic

- Propositional logic may be difficult to formulate for non-trivial problems
- Propositional logic does not include relations
- People like to think with classes, relations etc.
  - witness Protege and numerous other ontology editors
  - Ontologies can be communicated between different groups via the Ontology Web Layer (OWL)
  - Medical ontologies include NCI's ontology [5]. (Today, however, most big medical ontologies simply represent volcabilaries.)
  - OWL-based ontologies have a semantics based on description logic

Depending on your point of view description logics are a logic for ontologies *or* ontologies are a user interface for description logics.

Description Logic

"If a patient has breast cancer and is pregnant then Doxyrubicin is contra-indicated"

might be represented as

$$
\begin{aligned}
&not(\\
&\quad patient \\
&\quad \wedge\exists(hasDisease\ breast\_cancer) \\
&\quad \wedge\exists(hasCondition\ pregnant) \\
&\quad \wedge\exists(indicatedTreatment\ doxyrubicin))
\end{aligned}
$$

Think of it as a database query

Compute a set S as follows
take the set of things that are patients ($patent$)
intersect them ($\wedge$) with
the set of things that have a $hasDisease$ relation to
a $breastcancer$.
intersect that with
the set of things that have an $indicatedTreatement$
of $doxyrubicin$
The set S must be empty.

Description Logic

- Formally, description logic (ALC) is a restriction of predicate logic, and propositional logic is a restriction of description logic
- Description logic is decidable although certain problems of description logic have a high computational complexity
- Different variants of description logic allow different quantification, specifications of relations, etc.
- Some approaches use full first-order logic for knowledge-representation (e.g. KIF)

Description Logic as an Object Logic

"If a patient has breast cancer and is pregnant
then Doxyrubicin is contra-indicated"

How would an object-oriented designer view the statament?

- The class of "patient" has a disease attribute whose value is breast cancer.
- The domain of breast cancer is disease.
- The class patient may also has a subclass of female_patient who have a boolean attribute of "pregnant"
- Objects of type patient may also have a relation to objects of type treatment, with subclasses drug-treatment, pharmacological treatment, etc.

FLORA

- Logical rules whose literals are based on
  - objects, classes, isa relations
  - attributes and (mostly) binary relations
- Compiles into XSB Prolog [6]:
  - Combines with vanilla Prolog
  - Negation is XSB's Well-founded Negation
  - Constraints may be used in FLORA

"If a patient has breast cancer and is pregnant
then Doxyrubicin is contra-indicated"

```
Patient[contraIndicated -> doxyrubicin] :-
       Patient:patient,
       Patient[condition -> pregnant],
       Patient[disease -> breast_cancer].
```

FLORA

FLORA with DL-style constraints

- Object-oriented rules allow logic + object orientation
- DL-style constraints give semantics of a decidable subset of predicte logic

Other formalisms not yet as mature, such as Coherent Description Framework, used by MD Logix

Conclusions

- Predicate logic and logic programming are close, but not identical
  - Can be combined through constraint logic programs
- Ontological knowledge can be combined through object logics and constaints
- Mechanisms are not seamless
- Knowledge acquisition remains a problem
  - Natural language systems may produce description logics
  - Ontologies may be mapped to description logics
  - OWL and other standards allows these description logics to be shared
  - RuleML allows Prolog, FLORA and other rules to be intercommunicated

References

[1] V. Santos Costa, D. Page, M. Qazi, and Cussens J. Clp(*bn*): Constraint logic programming for probabilistic knowledge. In *Uncertainty in AI*, 2003.

[2] Thom Fruhwirth. Constraint handling rules. *Journal of Logic Programming*, 37(1-3), 1997.

[3] M. Garey and D. Johnson. *Computers and Intractibility: A Guide to the Theory of NP-Completness*. W.H. Freeman and Company, 1979.

[4] J. Gartner, T. Swift, A. Tien, L. M. Pereira, and C. Damásio. Psychiatric diagnosis from the viewpoint of computational logic. In *8th International Workshop on Non-Monotonic Reasoning*, 2000.

[5] F. Hartel, S. de Coronado, R. Dionne, G. Fragoso, and J. Golbeck. Modeling a description logic volcabulary for cancer research. *Journal of Biomedical Informatics*, 38:114–129, 2005.

[6] XSB. The XSB Logic Programming System version 2.7.1, 2005. `http://xsb.sourceforge.net`.