

# Logic Programming for Healthcare Informatics: an Informal Survey

Terrance Swift

Medical Decision Logic, Inc

CENTRIA, Universidade Nova de Lisboa

# Disclosures

- Terrance Swift has financial interest in mdlogix, Inc which is mentioned in this talk
- Terrance Swift has financial interest in XSB, Inc which is not mentioned in this talk
  - The programming system XSB Prolog is open-source and freely available. It is not controlled in any way by XSB, Inc, although they have generously contributed to its development

# Learning Objectives

- to survey how formal logic and logic programming have been used for knowledge representation and reasoning for a variety of medical applications
- to sketch the current state of logic programming systems, focusing on open-source systems
- to indicate how some current research directions in logic programming may be relevant to areas such as adaptive workflow management for health-care or for dynamic decision support.

... a little learning is a dangerous thing, and so is writing your learning objectives before your talk :-)

# Motivation

- Computer Science has been highly successful in some areas, such as medical imaging or genetic sequence analysis.
- It has been less successful in other areas such as clinical decision support, or adaptive workflow management.
  - For these areas background knowledge is complex, multi-paradigm, and diffuse
- Can computational logic help? What advantages does it offer?

# Overview

- Some concepts in formal logic
  - Propositional logic; predicate logic; derivations
  - As little as possible and as informal as possible
- Uses of different brands of logic programming
  - Rule based systems with constraints
  - Answer Set Programming
  - Formal ontologies and their reasoners
- Uncertainty: where we are now
  - Different kinds of uncertainty
  - Combining probabilities with rules
  - Combining probabilities with Answer Set Programs
- Putting it mostly together
  - Open source systems: XSB, YAP, and others

# Background: Propositional Logic

Boolean statements about atomic propositions

*"if a patient has breast cancer, Doxorubicin and Tamoxifen are indicated"*<sup>1</sup>

might be translated as

*breast\_cancer* implies

(  
    *doxorubicin\_indicated*  
    and  
    *tamoxifen\_indicated*  
)

---

<sup>1</sup>All medical examples should be taken only to illustrate features of logics or computations.

## Background: Predicate Logic

- also represents individuals, functions, relations, and the existence or universality of statements
  - e.g. *affects(Drug, TumorType)* can be used to make lots of similar statements about different drugs and tumors
  - The logic can make inferences about the individuals (or functions) within the relation
- Maps well to relational databases

## Background: Predicate Logic

A universal statement: *"if any patient has breast cancer, Doxorubicin and Tamoxifen are indicated"*

---

$\forall P.breast\_cancer(P)$  implies  
(  
    *indicated(doxorubicin, P)*  
    and  
    *indicated(tamoxifen, P)*  
)

---

An existential statement: *"if fever persists with elevated white blood cell count, there is an underlying infection"*

## Background: Logic Programs

*”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.
```

# Background: Logic

- In Logic
  - Knowledge is encoded in *axioms*
  - A set of axioms is a *theory*
- In a logic program
  - Knowledge is encoded in rules
  - A program is a set of rules
- So what is the difference?

# Background: Derivations

- Given a logic and a theory, a *derivation procedure* indicates sets of statements entailed by the theory
  - Such statements range from the mundane *This Sudoku has no solution with 9 in the top left box ...*
  - To the profound *an abelian group has a composition series iff it is finite...*
  - To the interesting *When a patient enters the protocol, she will always leave the protocol if at any time her white blood count is critical for 5 successive days*
- Running a logic program is the same as performing a derivation procedure on the program
- Different types of derivation differ most critically in their *abstract complexity* (cf. [35]) – in how big a problem you can reliably solve
- Less complex derivation methods lean toward the programming side of logic programming
- More complex derivation methods lean toward the logic side

# Background: Derivations

- *Linear resolution* scales well, e.g. Prolog (SLD [54]) or tabling (SLG [12, 11])<sup>2</sup>.
  - If your program or data grows 2x your derivation will take at most 2x longer – not 4x or 16x
  - Here *linear resolution = rule application*
  - Rule application can perform general programming (like Java, C# or other languages)
  - Rule application may be augmented with special-purpose constraint solvers
- *Satisfiability-checking* for truth assignments of propositional programs can be more powerful (for such programs) but more expensive (NP-complete)
- Full reasoning with ontologies is more powerful still and also more expensive (PSPACE-complete or worse)
- No complete derivation procedure exists for all theories in predicate logic (Godel's Incompleteness Theorem)

---

<sup>2</sup>Technically SLG is not linear in the size of a program when computing the well-founded semantics, but it scales linearly in practice for non-pathological programs.

# Rule Application for Medical Workflows

- Broadly, medical workflows are systems that take action depending on the state of a patient and her environment
- May be used to generate alerts, pre-schedule resources, indicate relevant information
- Basis for workflow may be research protocol or clinical guideline

# Rule Application for Medical Workflows

(Adaptive) Medical Workflows [50, 21, 7, 27, 33]

- A given patient may be treated with one first protocol or according to one guideline, and then another
- Sample rule:  
*ETOPOSID must be dropped when a patient has had a critical blood status for the last five days, and ETOPOSID can only be given again once the blood status becomes normal again (leukocyte count > 1000)*
- A single research protocol might contain a hundred or more such rules.
- Event then it would not (fully) model
  - background knowledge (e.g. chemical structure of Etoposid)
  - factors outside the protocol or guideline (e.g. co-morbidity)
  - uncertainty
  - resource constraints for treatment
- There could be lots of such guidelines, all linked together.

# Temporal Constraints in Medical Workflows

- As the above example illustrates, you don't get too far with rules for medicine before time elements become very complex.
- A Phase II study example:  
*Patients will start adjuvant TMZ 28 days (+/- 10 days) post the completion of Radiation Therapy and concomitant TMZ. Temozolomide will be administered orally once a day for 5 consecutive days every 28 days [18]*
- There are many different types of temporal reasoning: temporal constraints (e.g. [29, 45]) are relatively easy to understand and have low complexity (compared to other types of temporal reasoning).

# Temporal Constraints in Medical Workflows

- Temporal constraints have been studied in clinical guidelines [4, 5] and in research protocols at mdlogix
- In both contexts, temporal constraints are require
  - qualitative intervals [2] *concomitant TMZ occurs **during** radiation therapy*
    - \* before, after, during, contains, starts, started\_by, finishes, finished\_by, overlaps, overlapped\_by, meets
  - quantitative intervals: *28 days (+/- 10 days) post*
  - repetitions and periodicity *once a day for 5 consecutive days every 28 days*

# Temporal Constraints in Medical Workflows

Input to temporal constraint solver may be

- a study table (e.g. in mdlogix's Clinical Research Management System, CRMS)
- a workflow rule like

```
WHEN    critical-blood-status(P) VALID-TIME [now - (5,day),now] AND
        in-further-workflow(Drug-Administration[drug = Etoposid],P)
THEN    drop(Drug-Administration[drug = Etoposid],P)
        Unless normal-blood-status
        VALID-TIME now
```

- this is an extension of the syntax of F-logic rules [25, 58]
- Temporal constraints may be *incrementally* added by application of a particular rule
  - The length of time for the treatment of Etoposid could be limited by several independent factors (events). Each event would constrain the length of Etoposid treatment
- Many other solvers integrated with rules through special and general solvers (cf. [6, 19]).
  - Spatial constraints, integer programming, Belief nets

# Temporal Constraints in Medical Workflows

How constraints might look to a temporal constraint solver:

```
arc(adjuvant_tmz,adjuvant_tmz_1,[before([28,28]))],
arc(adjuvant_tmz_1, cycles_1_3_5_1, [contains, started_by]),
arc(adjuvant_tmz_1, cycles_2_4_6_1, [contains]),
arc(within_10_days_cycles_1_3_5_1, cycles_1_3_5_1, [before([1,10]))],
arc(within_3_days_cycles_1_3_5_1, cycles_1_3_5_1, [before([1, 3]))],
arc(within_10_days_cycles_2_4_6_1, cycles_2_4_6_1, [before([1,10]))],
arc(within_10_days_cycles_2_4_6_1, cycles_2_4_6_1, [before([1, 3]))],
arc(within_3_days_cycles_1_3_5_1,
    within_3_days_cycles_1_3_5_delay_cycle_1, [before([0,7]))],
arc(within_3_days_cycles_2_4_6_1,
    within_3_days_cycles_2_4_6_delay_cycle_1, [before([0,7]))],
```

# Medical Workflows

- Guidelines or protocol are encoded in logic rules
- These rules are declarative to a programmer – perhaps to a domain expert
- The rules are extended with temporal constraints
  - Solving temporal constraints is more complex than performing the rule application, but the temporal constraint part can be “factored out” and does not effect the rule application
  - Constraint evaluation periodically interrupts rule application
- The rules may be written in an object logic, which uses object-orientation to help organize the rules and background knowledge

# Verification of Medical Workflows

- The workflows, guidelines, etc. written with rules can be formally verified [40, 17, 28]) to answer questions like
  - Under a given protocol, would a patient  $P$  ever receive a particular treatment? Under what conditions?
  - Under a given protocol, can we ensure  $P$  would never be given a contra-indicated drug
- They can also easily simulate scenarios
  - Given a patient in a particular state, illustrate a sequence of actions and events that would take him to another state
- Verification and scenario simulation of a workflow are both more complex than running the workflow.

# Verification of Medical Workflows

- A potential disadvantage is that the rule-based workflows do not always make explicit the issues of concurrency that arise when modeling multiple
  - agents – patients, physicians, nurses
  - resources – beds on a particular ward, slots for surgery
- For multi-agent workflows, the rules should be combined with a process-logic, such as CCS [31],  $\pi$ -calculus [32] or Colored Petri Nets [49].

# Medical Workflows: Changes and Policy

- There are also semantics for how a rule-based workflow evolves for changing knowledge [1]
  - How would a particular scenario (set of actions and events) change if I change a rule?
  - The update logic can *introspect* the workflow logic
- Rule-based systems can be extended to allow preferences [16] in case more than one rule applies in a given situation
  - Preferences can be seen as “meta”-rules that can fine tune for policies of a ward or hospital

# Satisfiability of Propositional Programs

The propositional formula

$(A \text{ or } B \text{ or } \neg C)$  and  $(\neg A \text{ or } \neg B)$  and  $(\neg A \text{ or } B)$

is satisfied by the truth assignment

$$\{\neg A, B, C\} \quad \{\neg A, B, \neg C\} \quad \{\neg A, \neg B, \neg C\}$$

Propositional satisfiability is the canonical combinatorial problem — a problem where a search for a solution may need to consider a lot of different combinations

# Satisfiability of Propositional Programs

E.g. Sudoku

1,1	1,2						1,9
2,1	2,2						
		3,3					
9,1							9,9

$(A_{1,1} = 1 \text{ or } A_{1,1} = 2 \text{ or } \dots \text{ or } A_{1,1} = 9)$

and  $\neg A_{1,1} = A_{2,1}$  and  $\neg A_{1,1} = A_{3,1} \dots$  and  $\neg A_{8,1} = A_{9,1}$

and  $\neg A_{1,1} = A_{1,2}$  and  $\neg A_{1,1} = A_{1,3} \dots$  and  $\neg A_{1,8} = A_{1,9}$

and  $(A_{1,1} = 1 \text{ or } A_{1,2} = 1 \text{ or } A_{1,3} = 1 \text{ or } \dots \text{ or } A_{3,3} = 1)$

etc.

The best techniques can handle  $36 \times 36$  Sudokus

# Satisfiability of Propositional Programs

*Answer Set Programming* (ASP) uses rules to encode combinatorial problems in a more readable way.

---

## *Domain rules*

```
row{1..9}. col{1..9}. val{1..9}.  
cell(Row,Col,Val):- row(Row),col(Col),val(Val).
```

## *Row constraints*

```
:- cell(Row,Col1,Val),cell(Row,Col2,Val),not(Col1 = Col2).
```

## *Column constraints*

```
:- cell(Row1,Col,Val),cell(Row2,Col,Val),not(Col1 = Col2).
```

## *Region constraints (first region)*

```
:- cell(1,1,Val),cell(2,2,Val).      :- cell(1,1,Val),cell(3,3,Val).  
:- cell(1,1,Val),cell(2,3,Val).      :- cell(1,1,Val),cell(3,2,Val).  
:- cell(1,2,Val),cell(2,1,Val).      :- cell(1,2,Val),cell(3,1,Val).  
:- cell(1,2,Val),cell(2,3,Val).      :- cell(1,2,Val),cell(3,3,Val).  
:- cell(1,3,Val),cell(2,1,Val).      :- cell(1,3,Val),cell(3,1,Val).  
:- cell(1,3,Val),cell(2,2,Val).      :- cell(1,3,Val),cell(3,2,Val).  
:- cell(2,1,Val),cell(2,3,Val).      :- cell(2,1,Val),cell(3,3,Val).  
:- cell(2,2,Val),cell(3,1,Val).      :- cell(2,2,Val),cell(3,3,Val).  
:- cell(2,3,Val),cell(3,1,Val).      :- cell(2,3,Val),cell(3,2,Val).
```

# Uses of ASP

... just so you don't think all computer scientists live in an ivory tower... epidemiology of tapeworm infestations [9]

- Determine phenotypic (or genotypic) characteristics of tapeworms taken in different regions
- Construct a minimal cladistic tree
- Relies on a “preference” of minimal cladistic trees over other trees

# Uses of ASP

- At MDL, we have started to look into ASP to solve problems in epidemiology using social network analysis [56, 47]
- Rule-based determination of coherent subgroups
  - Define core properties of a group: minimum and maximum number of entities, connectedness, diameter, etc.
  - Define preferential properties: gender balance of entities, each subgroup should contain a weak link to another subgroup, etc.
- Convergence problems in attributed belief or influence networks (also studied in [20])
  - Is there a ranking that *satisfies* the known arcs of influence?
  - Properties of nodes can be taken into account to distinguish the quality of different solutions, and reduce noise
- While still preliminary, these methods generalize those of traditional Social Network Analysis

## Uses of ASP

- Assessing risk for Assisted Living Patients [30] (field prototype)
- Information about Patients from wireless sensor networks, test results, patient profile
- Rules relate information to risk factors, encode common-sense reasoning
- Rule based approach to choosing among scenarios – does not make use of probability or other uncertainty measures.

## Satisfiability of Propositional Programs

- It is easier to check whether a solution to Sudoku is correct than to derive a solution to Sudoku
- Rule application can be used to check a solution or to *program an algorithm* to find a solution
- ASP can be used to derive a solution directly. Programming occurs in setting up the problem and in stating heuristics.
- Sudoku can also be programmed using constraints
- ASP may be faster because constraints are incremental, and SAT is not.
  - A tremendous amount of effort has been put into (non-incremental) SAT-solvers (cf. [24]), because they are used to check properties of computer chips
  - ASP can exploit this work
- However, because constraints are incremental, they combine better with rules (e.g. the Etoposid rule)

# Uses of Predicate Logic

Predicate logic (or a decidable fragment) is good for representing knowledge such as

- background medical or policy knowledge [41, 51]
  - E.g. Contra-indicated drugs, procedures, etc.
- To organize knowledge arising from different sources [52, 23]
  - e.g. sharing knowledge about immune tolerance for organ transplants [48]
- To help understand the state of a patient based on physician discourse with a patient [13]
- The most popular approach is formal ontologies (a.k.a. *description logics*)

# Ontologies

Ontologies are another way to represent knowledge

- Much of ontologies is easy to grasp
- Example from NCI Thesaurus (apologies for not using screen shots of Protege [39])

<b>Class</b>	'Acetyl Coenzyme A
<b>isa</b>	'Coenzyme A'
<b>isa</b>	'Biologically Active Substance'
<b>isa</b>	'Organic Chemical'
<b>hasSynonym</b>	'Acetyl-CoA'
<b>hasDefinition</b>	'The condensation product of coenzyme A and acetic acid which participates in the biosynthesis of fatty acids and sterols, in the oxidation of fatty acids and in the metabolism of many amino acids. In addition, Acetyl Coenzyme A acts as a biological acetylating agent.'

# Advantages of Ontologies

- Many parts of an ontology are easy to understand
  - One class is a subclass of another
  - Several subclasses can be declared to be disjoint.
  - Objects are individuals in subclasses
- Binary relations are allowed and inherited by subclasses
  - If the class SSRI affects a metabolic pathway, and Fluoxetine hydrochloride is a SSRI, then Fluoxetine hydrochloride affects a metabolic pathway
- Ontologies have formal basis with a simple set-theoretic semantics
- Most information in ontologies are easily visualizable (sorry about not showing Protege)
- There are standards such as OWL for communicating ontologies
- Ontologies can be combined, since reasoners can check consistency
- Queries can be made not only about what is in an ontology, but what an ontology entails

# Limitations of Ontologies

- Some knowledge bases call themselves ontologies but are not actually based on logic
- Other ontologies are little more than hierarchies
- Certain aspects of ontologies are difficult to visualize (Ontology constraints, axioms)
- Full reasoning in ontologies is *very* complex if constraints and axioms are used.
  - This means that in a large complex ontology, even a simple query might not be answered in a day, an hour, or a week
- The restriction to binary relations (which makes ontologies decidable) also makes it difficult to model information about processes or time. It can be done (e.g. [22]) but it ain't pretty
- Ditto for aggregates (min, max, average...)

## Using Ontologies

- Everything has its trade-offs, and some of the limitations are the price off success
- Still, you have to be very careful about using reasoners within ontologies due to their complexity
- Ontologies make a *great* combination with rules. The rules can rely on the ontology to maintain background knowledge about pharmaceuticals, procedures, metabolic pathways, etc.
- It can be helpful if the rules use an object logic which can make explicit use of ontology style information (e.g. F-logic (shown above) or CDF [53])
- Full integration of rules and ontologies is an active research area

# Uncertainty in Medical Knowledge

1. Lexical: How certain am I that I am using the right term?
2. Observational: How certain am I that I saw what I thought I saw? ( $P(\textit{observation})$ )
3. Occurrence: How likely is it that the event happened? ( $P(\textit{event})$ )
4. Measurement: How likely is the event really to have happened even if I think it did? ( $P(\textit{observation}|\textit{event})$ )
5. Causal: How influential are different preconditions in making the event happen? ( $P(\textit{event}|\textit{cond}_1)$  vs  $P(\textit{event}|\textit{cond}_2)$ )
6. Reference Class: How do the previous uncertainties change if I change my assumptions about what the context is? ( $P(\textit{event}|\textit{context}_1)$  vs  $P(\textit{event}|\textit{context}_2)$ )
7. Temporal: How does the chance of true occurrence change over time?
8. Methodological: How does the strategy I used to collect data affect the likelihood of uncertainties 2-4?
9. Model: Have I modeled everything properly?
10. Meta: uncertainty about these other uncertainties
11. How certain am I that I made the right decision?  
(Based on a discussion with Harold Lehmann)

# Probability and Logic

- Depending on how much you know, you may want to use techniques such as fuzzy logic, rough sets, or most importantly probability.
- Probabilistic reasoning is most often done through Bayesian Nets, which use conditional independence to reduce the effort of modeling large joint distributions
- Some medical diagnosis/assessment systems without probabilities are actually used (e.g. [42]), but probabilities greatly improve diagnostic capability (e.g. [46])

# Google-based Medicine

Consider some discrete variables:

<i>Mini-Stroke</i>	<i>Meningitis</i>	<i>Optic Nerve Hypoplasia</i>
<i>Nystagmus</i>		<i>Optic Nerve Color</i>
<i>Patient Age</i>	<i>Smoker</i>	<i>Patient Gender</i>
<i>Visual Field Disturbance</i>		

- There may be 1000+ combinations of variable values, depending on how many values each variable takes
- If we want to treat these as random variables, do we have to create a joint pdf with 1000+ values?
- Bayesian Nets (cf. [36]): depending on what we know, some variables may be *conditionally independent* of others,
  - Given that *Optic Nerve Color = normal* *Nystagmus* may be independent of *Optic Nerve Hypoplasia*

# Probability and Logic

Bayesian Nets are popular because

- They can answer queries:
  - What is the probability of *Mini-Stroke* given *Nystagmus*, *Patient Age* and *Smoker*
  - What is the probability of *Optic Nerve Color = Pale* given *Meningitis* vs. given *Optic Nerve Hypoplasia*
- There are numerous algorithms to learn nets from data
- They can be extended to deal with imprecise or heterogenous probabilities [15]

# Probability and Logic

- Belief nets are an important technology that has gotten mature. But you still may need rules to
  - indicate relevant information, pre-schedule resources or take other actions
  - perform general programming tasks (though this could be done in other languages, perhaps not as easily)
- We could just build systems with a probabilistic part and a non-probabilistic part, but would like to combine probability with logic rules in a more meaningful way

# Probability and Logic

- There has been a tremendous amount of work on relating logic programs to probability, both semantically and computationally (e.g. [26, 44, 55] and many others).
- However, there is an issue in linking rules to probability: deriving posterior probabilities with a Bayesian net is as complex as satisfiability checking in propositional logic (both are NP-complete)
- One approach, Plog [8], combines discrete probabilities with ASP (and so does not consider linear rule application)
- Another approach  $CLP(BN)$  builds up a Bayesian Network incrementally, as constraints

## Probability and Logic: Plog

- Plog uses the conventions of ASP (which we used to solve Sudoku)
- Allows given rule to be associated with a discrete probability
- The answer to a query

$$p(event|observation_1, \dots, observation_n)$$

is determined by

- finding *all* satisfiable solutions that include *event* after the observations have been added to the program
- counting the solutions in which *event* is true (e.g. counting the Sudoku solutions in which  $A_{1,1} = 9$ )
- This may not be what we want for most kinds of uncertainties in medical knowledge
  - Probability distributions must be discrete
  - Since ASP-style programs are sensitive to the number of possibilities that they search, distributions must be small (coarse) for the program to run well — e.g. *high, medium, low*
  - Probability is based on the frequency of solutions — which could differ from the posterior probabilities determined by a net

# Probability and Logic: CLP(BN)

CLP(BN) appears to be a more promising approach

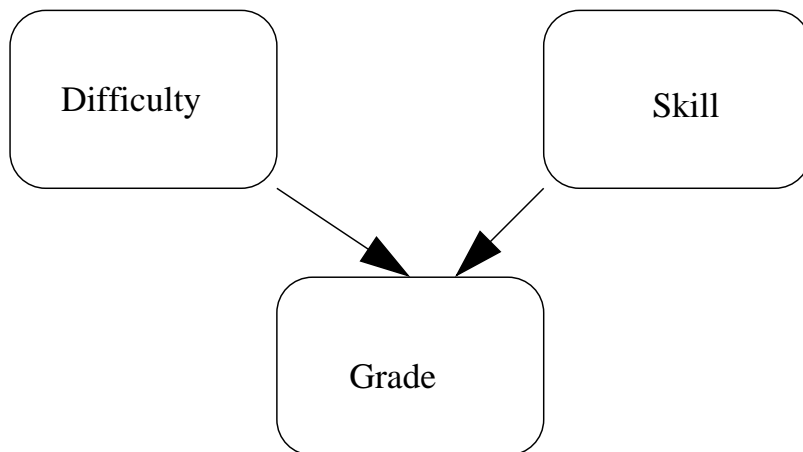
- CLP(BN) [14] adds probabilistic dependencies as constraints between variables – conditional independencies may be inferred from these dependencies
- CLP(BN) is incremental, so rule application can be used to construct a Bayesian net from a given patient (or other) context
- The nets constructed by CLP(BN) are formally related to joint probability distributions (and therefore Bayesian nets)
- Queries to the resulting Bayesian net can be sent to any solver

# Probability and Logic: CLP(BN)

A simple example

```
grade(Student,Grade):-  
  student_in_course(Course),  
  difficulty(Course,Difficulty),  
  skill(Student,Skill),  
  grade_table(Table),  
  {Grade = grade(Difficulty,Skill) with  
    p('A','B','C','D',Table,[Difficulty,Skill])}.
```

Grade is a random variable that is a function of the difficulty of the course, and the skill of the student, i.e.



## Probability and Logic: CLP(BN)

- In the above example, **Skill** and **Difficulty** were both obtained before **Grade** was queried, so that it was constrained to have a probability distribution, but no definite value
- If **Skill** were known, but **Difficulty** were not, then the **Grade** would be constrained to a subnet.
- CLP(BN) fixes two of the problems with Plog:
  - Since you're computing a Bayesian net, and not all satisfiable solutions, you should be able to have larger nets with finer probability distributions
  - The computation of posterior probabilities is exactly what you want
  - However, you don't get continuous probabilities. CLP(BN) should in principle be amenable to continuous probabilities, but the difficulty of doing so is unknown

# Probability and Logic

- Work on combining probabilities and ontologies is still very new
- Some work in systems biology, has begun to explore combining general stochastic information with process logics [37, 38]
  - This general approach addresses temporal aspects of metabolic pathways

# Putting these things (somewhat) together

My shameless self-promotion slide...

- Most open-source Prolog systems combine rules and constraints.
  - CLP(BN) was developed for YAP Prolog [43]
- Some Prologs allow tabling which is important for guaranteeing complexity and for permitting preferences
  - This is *not* table-driven programming
  - XSB [57] is the most sophisticated Prolog for tabling
- There are numerous ASP solvers. Most are stand-alone, so that they are difficult to integrate into a system. Smodels [34] is the only one to have a good programmers interface
  - XSB has a sophisticated interface to Smodels in XASP [10]
  - This has recently been extended to Plog [3]
- As discussed, how to intermix ontologies and rules is a hot topic
  - XSB has CDF to interface to formal ontologies of a restricted form. It is a work in progress, but it has been used commercially for about 5 years.

## Conclusions

- Rules, Constraints, Satisfiability Checking and Ontologies all have their advantages for Knowledge Representation and Reasoning
- Work is being done to integrate probabilities and other forms of uncertainties with each of these mechanisms
- There are many open questions, but...
- The computational logic community has solved or is working on a number of difficult issues that will benefit healthcare informatics
- Logic programming should be better known by the health informatics community
- Continues collaboration will benefit both fields

## References

- [1] J. Alferes, J. Leite, L. M. Pereira, H. Przymusinska, and T. Przymusinski. Dynamic logic programming. In *Procs of the Sixth International Workshop on Knowledge Representation and Reasoning*, pages 98–109, 1998.
- [2] J. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26:832–843, 1983.
- [3] H. Anh, C. Ramli, and C. Damsio. An implementation of extended P-Log using XASP. In *International Conference on Logic Programming*, pages 739–743, 2008.
- [4] L. Anselma. *Representing and reasoning with classes and instances of possibly periodic events Theory, algorithms and applications*. PhD thesis, Università di Torino, 2005.
- [5] L. Anselma, P. Terenziani, S. Montani, and A. Bottrighi. Towards a comprehensive treatment of repetitions, periodicity and temporal constraints in clinical guidelines. *Artificial Intelligence in Medicine*, 38:171–195, 2006.
- [6] K. APT. *Principles of Constraint Programming*. Cambridge University Press, 2003.
- [7] L. Ardissono, A. Di Leva, G. Petrone, M. Segnan, and M. Sonnessa. Adaptive medical workflow management for a context-dependent home healthcare assistance service. *Electronic notes in Theoretical Computer Science*, 2005.
- [8] C. Baral, M. Gelfond, and N. Rushton. Probabilistic reasoning with answer sets. *Theory and Practice of Logic Programming*, 2005.
- [9] D. Brooks, E. Erdem, J. Minett, and D. Ringe. Character-based cladistics and answer-set programming. In *Practical Applications of Declarative Languages*, pages 37–51, 2005.
- [10] L.F. Castro, T. Swift, and D.S. Warren. XASP: Answer Set Programming in XSB. Open-source software available at [xsb.sourceforge.net](http://xsb.sourceforge.net), 2002.
- [11] W. Chen, T. Swift, and D. S. Warren. Efficient top-down computation of queries under the well-founded semantics. *J. Logic Programming*, 24(3):161–199, September 1995.
- [12] W. Chen and D. S. Warren. Tabled Evaluation with Delaying for General Logic Programs. *Journal of the ACM*, 43(1):20–74, January 1996.
- [13] P. Ciccarese, E. Wu, G. Wong, M. Ocana, J. Kinoshita, A. Ruttenburg, and T. Clark. The SWAN biomedical discourse ontology. *Journal of Biomedical Informatics*, 41:738–751, 2008.
- [14] Vitor Santos Costa and James Cussens. Clp(bn): Constraint logic programming for probabilistic knowledge. In *In Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI03)*, pages 517–524. Morgan Kaufmann, 2003.

- [15] F.G. Cozman. Credal networks. *Artificial Intelligence*, 120:199–233, 2000.
- [16] B. Cui and T. Swift. Preference logic grammars: Fixed-point semantics and application to data standardization. *Artificial Intelligence*, 138:117–147, 2002.
- [17] G. Duftschmid, S. Miksch, and W. Gall. Verification of temporal scheduling constraints in clinical practice guidelines. *Artificial Intelligence in Medicine*, 25:93–121, 2002.
- [18] S. Grossman et. al. Phase ii study of temozolomide during and following external beam radiation therapy in patients with newly diagnosed glioblastoma multiforme who have undergone optimal surgical resection and insertion of gliadel wafer. Technical report, Johns Hopkins Brain Tumor Program, 2005.
- [19] Thom Frühwirth. Constraint handling rules. *Journal of Logic Programming*, 37(1-3), 1997.
- [20] M. Gebser, T. Schaub, S. Thiele, B. Usadel, and P. Veber. Detecting inconsistencies in large influence networks with answer set programming. In *International Conference on Logic Programming*, 2008.
- [21] U. Greiner, J. Ramsch, B. Heller, M. Lffler, R. Müller, and E. Rahm. Adaptive guideline-based treatment workflows with Adaptflow. In *Proceedings of the Symposium on Computerized Guidelines and Protocols*, pages 113–117, 2004.
- [22] V. Haarslev, C. Lutz, and R. Møller. A description logic with concrete domains and a role-forming predicate operator. *Journal of Logic and Computation*, 9(3):351–384, 1999.
- [23] F. Hartel, S. de Coronado, R. Dionne, G. Fragoso, and J. Golbeck. Modeling a description logic vocabulary for cancer research. *Journal of Biomedical Informatics*, 38:114–129, 2005.
- [24] H. Kautz and B. Selman. Ten challenges redux: Recent progress in propositional reasoning and search. In *Ninth International Conference on Principles and Practice of Constraint Programming*, 2003.
- [25] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42:741–843, July 1995.
- [26] D. Koller and A. Pfeiffer. Probabilistic frame-based systems. In *Proceedings of the National Conference on Artificial Intelligence*, pages 580–587, 1998.
- [27] G. Leonardi, S. Panzarasa, S. Quaghlini, M. Stefanelli, and W. van der Aalst. Interacting agents through a web-based health serviceflow management system. *Journal of Biomedical Informatics*, 40:486–499, 2007.
- [28] S. Lu, A. Bernstein, and P. Lewis. Automatic workflow verification and generation. *Theoretical Computer Science*, 353:71–92, 2006.
- [29] I. Meiri. Combining qualitative and quantitative constraints in temporal reasoning. *Artificial Intelligence*, 87(1/2):343–385, 1996.

- [30] Alessandra Mileo, Davide Merico, and Roberto Bisiani. A logic programming approach to home monitoring for risk prevention in assisted living. In *International Conference on Logic Programming*, 2008.
- [31] R. Milner. *Communication and Concurrency*. Prentice-Hall, New York, 1989.
- [32] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, I. *Information and Computation*, 100(1):1–40, 1992.
- [33] N. Mulyar, W. van der Aalst, and M. Pelig. A pattern-based analysis of clinical computer-interpretable guideline modeling languages. *Journal of the American Medical Informatics Association*, 14(6):781–788, 2007.
- [34] I. Niemeä and P. Simons. Smodels: An implementation of the stable and well-founded semantics for normal LP. In *International Conference on Logic Programming and Non-Monotonic Reasoning*, volume 1265 of *LNAI*, pages 420–429, 1997.
- [35] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [36] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan-Kaufmann, 1988.
- [37] C. Piazza. Systems biology: Models and logics I. In *International Conference on Logic Programming*, 2008.
- [38] A. Policriti. Systems biology: Models and logics II. In *International Conference on Logic Programming*, 2008.
- [39] *Protege Documentation Set: release 1.7*, 2001. Available via <http://protege.stanford.edu>.
- [40] Y. S. Ramakrishna, C. R. Ramakrishnan, I. V. Ramakrishnan, S. Smolka, T. Swift, and D. S. Warren. Efficient model checking using tabled resolution. In *Proceedings on the Conference on Automated Verification*, pages 143–154, 1997.
- [41] D. Rubin, J. Gennari, and M. Musen. Knowledge representation and tool support for critiquing clinical trial protocols. In *Proceedings of the AMIA Symposium*, pages 724 – 728, 2000.
- [42] R. Rymon. Goal-directed diagnosis diagnostic reasoning in exploratory-corrective domains. In *International Joint Conference on Artificial Intelligence*, 1993.
- [43] V. Santos Costa, L. Damas, R. Reis, and R. Azevedo. *YAP User’s Manual*, 2000. <http://www.ncc.up.pt/~vsc/Yap>.
- [44] T. Sato and Y. Kameya. Parameter learning of logic programs for symbolic statistical modeling. *Journal of Artificial Intelligence Research*, 15:391–454, 2001.
- [45] E. Schwalb and L. Vila. Temporal constraints: A survey. *Constraints: An International Journal*, 3(2/3):129–149, 1998.

- [46] G.C. Scott and R.D. Schacter. Individualizing generic decision models using assessments as evidence. *Journal of Biomedical Informatics*, 38:281–292, 2005.
- [47] J. Scott. *Social Network Analysis: A Handbook*. Sage Publications, Thousand Oaks, Ca, 2000.
- [48] R. Shankar, S. Martins, M. O’Connor, D. Parrish, and A. Das. An ontology-based architecture for integration of clinical trial management applications. In *Proceedings of the AMIA Symposium*, 2007.
- [49] E. Smith. Principles of high-level nets. In *Lectures on Petri Nets I: Basic Models*, pages 174–211. Springer LNCS 1491, 1998.
- [50] C. Spyropoulos. Ai planning and scheduling in the medical hospital environment. *Artificial Intelligence in Medicine*, 20:101–111, 2000.
- [51] R. Steele and J. Fox. Tallis proforma primer. Technical report, Cancer Research UK Advanced Computation Laboratory, 2002.
- [52] R. Stevens, C. Goble, I. Horrocks, and S. Bechhofer. Building a bioinformatics ontology using OIL. *IEEE Transactions on Information Technology in Biomedicine*, 6(2):135–141, 2002.
- [53] T. Swift and D. S. Warren. The meaning of cold dead fish. Available via <http://www.cs.sunysb.edu/~tswift>, 2003.
- [54] M.H. van Emden and R.A. Kowalski. The semantics of predicate logic as a programming language. *JACM*, 23(1):733–742, 1976.
- [55] J. Vennekens, S. Verbaeten, and M. Bruynooghe. Logic programs with annotated disjunctions. In *International Conference on Logic Programming*, 2004.
- [56] S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, 1994.
- [57] *The XSB Programmer’s Manual: version 3.1*, 2007.
- [58] G. Yang, M. Kifer, and C. Zhao. *FLORA-2: User’s Manual Version 0.94*, 2005. Available via [flora.sourceforge.net](http://flora.sourceforge.net).