Monadic Predicate Logic is Decidable

Boolos et al, *Computability and Logic* (textbook, 4th Ed.)
Nota>on
These slides use
A instead of ∀
E instead of ∃
& instead of ∧
¬ instead of ¬
Equality statements are atomic formulas:
x=y, a=b, x=a, etc
“sentence” = formula with no free variables
Monadic First-Order Predicate Logic (FOPL)

• The fragment of Predicate logic that uses no predicates with more than 1 argument

• In: \[ \text{Ex } F(x) \land \text{Ey } -F(y) \]
  \[ \text{AxEy } -(x=y) \] (equality statements permitted!)
  \[ G(a) \lor G(b) , \text{ etc.} \]

• Out: \[ \text{AxEy } (R(x,y)) \]
  \[ R(a,b,b) \lor \text{Ex } F(x) \]
  (because they use a dyadic/triadic/etc predicates)
Some reasoning tasks

• For given sentences $\phi$ and $\psi$, does $\psi$ follow from $\phi$? ("Does $\phi$ have $\psi$ as a logical consequence?")
  – More precisely: Is it true that for all models $M$,
    if $M \models \phi$ then $M \models \psi$?

• For given a sentence $\phi$, is $\phi$ satisfiable? We mean:
  – Is there a model that $M$ such that $M \models \phi$?
  – E.g., $\exists x F(x) \land \exists x \neg F(x)$ is satisfiable

• Analogous for formulas with free variables
Monadic FOPL satisfiability is decidable
Key theorem
(Lőwenheim-Skolem 1915)

• If $S$ is a monadic sentence that has a model, then $S$ is true in some model whose domain consist of at most $2^k.r$ members, where $k$ is the number of predicate letters in $S$ and $r$ the number of variables in $S$.

• Part 1, proof: The Key Theorem

• Part 2, proof: It follows that monadic FOPL is decidable
Proof of Part 1 (= Key Theorem)

- Let $S$ be a sentence of monadic FOPL. Its predicates are $P_1, \ldots, P_k$
- Let $M \models S$, and let $D$ be the domain of $M$
  - $D$ may be infinite
- Let the signature of $d$ in $D$ (henceforth $\text{sig}(d)$) be the sequence $<j_1, \ldots, j_k>$ where $j_i = 1$ if $M$ specifies that $P_i$ is true of $d$ and $j_i = 0$ otherwise
  - $\text{sig}(d)$ tell us which predicates in $S$ are true of $d$
  - given $S$, there are exactly $2^k$ different possible signatures
• We call d and d’ similar if \( \text{sig}(d) = \text{sig}(d’) \)
  
  – This means that d and d’ happen to share all their properties \( P_1, \ldots, P_k \).

• “similar” is an equivalence relation, so each d in D belongs to an equivalence class of similar domain elements
  
  – Each equivalence class is a subset of D
  – there are at most \( 2^k \) equivalence classes
Towards a **smaller model** $M'$

- Construct a subset $E \subseteq D$ as follows:
- Choose $r$ elements from each equivalence class
  - If a class has fewer than $r$ elements then choose them all
- $E$ cannot have more than $2^k \cdot r$ elements ($r$ for each equivalence class)
- Define: $M'$ is the restriction of $M$ to $E$
  - just like $M$, but defined for elements of $E$ only
- To be proven: $|M'| = S$
A useful concept: match

• Informally: Two sequences of elements of $E$ that are of the same length match if their elements are similar and differences within each sequence are “respected” in the other:

• Formally: $c_1,..,c_n$ matches $d_1,..,d_n$ iff
  1. $c_i$ is similar to $d_i$ (for every $1 <= i <= n$)
  2. $c_i = c_j$ iff $d_i = d_j$ (for every $1 <= i,j <= n$)
Example

These 2 sequences of domain objects do not match:

\[ c_1, \ldots, c_n = a, b, a \]
\[ d_1, \ldots, d_n = a, b, c \]

If a and c are similar then clause 1 is fulfilled, but clause 2 is not (because \( c_1 = c_3 \) but \( d_1 \neq d_3 \))

Reason behind clause 2: equality statements in FOPL (e.g. in the sentence \( A x E y - (x=y) \))
Another useful concept

• A formula $\phi$ containing at most the free variables $x_1,..,x_n$ is **satisfied by elements $d_1,..,d_n$ in a model $M$ iff**

$$M \models \phi (x_1:=d_1,..,x_n:=d_n)$$

(A simple extension of the idea of satisfiability)
Lemma

Let

• $G(x_1,..,x_n)$ be any subformula of $S$, containing at most the free variables $x_1,..,x_n$
• $d_1,..,d_n$ a sequence of elements of $D$ (the original domain)
• $e_1,..,e_n$ a sequence of elements of $E$ (dom constructed above)
• $d_1,..,d_n$ matches sequence $e_1,..,e_n$

Then

$G(x_1,..,x_n)$ is satisfied by $d_1,..,d_n$ in $M$ iff $G(x_1,..,x_n)$ is satisfied by $e_1,..,e_n$ in $M'$
Why does this lemma hold? (informally)

• As far as the predicates $P_1, \ldots, P_k$ occurring in $S$ are concerned, each element $d_i$ is just like $e_i$
  – Clause 1 of “match”

• The only other thing that can matter (because of equality statements!) is whether two elements in a given sequence are identical
  – Clause 2 of “match”
Sketch of a formal proof (by formula induction)

• **Base Cases:** G is atomic. G is of the form $P_i(t)$ or of the form $t_1 = t_2$ (t, $t_1$, and $t_2$ are variables or constants)

1. Let $G = P_i(t)$. We need to prove:
   
   $P_i(t)$ is satisfied by $d_1$ in $M$ iff
   $P_i(t)$ is satisfied by $e_1$ in $M'$

   But $d_1$ and $e_1$ are similar, hence the same predicates hold true of $d_1$ and $e_1$ (including the predicate $P_i$). This proves the first Base Case.
Sketch of proof by formula induction

- **Base Cases**: G is atomic. G is of the form $P_i(t)$ or of the form $t_1=t_2$.

2. Let $G = t_1=t_2$. We need to prove

   $t_1=t_2$ is satisfied by $d_1, d_2$ in $M$ iff
   
   $t_1=t_2$ is satisfied by $e_1, e_2$ in $M'$

   But the sequences $d_1 d_2$ and $e_1 e_2$ match, hence $d_1 = d_2$ iff $e_1 = e_2$. This proves the second Base Case.
Sketch of proof by formula induction

**Inductives Cases:** [Proofs omitted, but see Questions for the Practical] It suffices to address -, v, A.

1. Assume the Lemma holds for $\phi$. Prove that it holds for $\neg \phi$.
2. Assume the Lemma holds for $\phi$ and $\psi$. Prove that it holds for $\phi \lor \psi$.
3. Assume the Lemma holds for $\phi$. Prove that it holds for $\forall x \phi$. 
• S is itself a subformula of S, hence it follows directly (with n=0) from the Lemma that

\[
S \text{ is true in } M \text{ iff } S \text{ is true in } M'
\]

*Recall:* M may be infinite, but M’ is finite, with at most \(2^k \cdot r\) elements
Proof of Part 2

• Let S be a FOPL sentence
• Associate with S a quantifier-free formula S’ such that $\text{S’ is satisfiable iff S is.}$ (Next page)

If we manage to do this then we deduce:

• The satisfiability of S can be decided using truth tables (since these suffice for deciding the satisfiability of S’)
• Hence the satisfiability of S can be decided
Proof of Part 2

Making use of Part 1, associate with S a quantifier-free formula $S'$ which is satisfiable iff S is. As follows:

Inductively associate a quantifier-free $H'$ with each subformula $H$ of $S$, as follows:

- If $H$ is atomic: $H' = H$ (no change!)
- If $H$ is a truth functional compound: $H' = H$
- If $H = \text{Ex}F$: $H' = F(a_1)v..vF(a_m)$ $m = 2^k.r$
- If $H = \text{Ax}F$: $H' = F(a_1)&..&F(a_m)$ $m = 2^k.r$

$S$ itself is a subformula of $S$, so this constructs a quantifier-free $S'$ as well. The construction guarantees:

$S'$ is satisfiable iff $S$ is satisfiable
Example  (using an arbitrary $S$)

Consider $S =$

$$(((\text{Ex}F(x) \& \text{Ex}G(x)) \& -\text{(Ex}(F(x)\&G(x))))$$

Here $k=2$, $r=3$, so $2^k.r=12$

The following formula is constructed:

$$F(a_1) \lor .. \lor F(a_{12}) \& G(a_1) \lor .. \lor G(a_{12}) \&$$

$$- ((F(a_1)\&G(a_1)) \lor .. \lor (F(a_{12})\&G(a_{12}))))$$
Example

\[
F(a_1) \lor \ldots \lor F(a_{12}) \land G(a_1) \lor \ldots \lor G(a_{12}) \land
- ((F(a_1) \land G(a_1)) \lor \ldots \lor (F(a_{12}) \land G(a_{12})))
\]

Propositional formula with 24 atoms
Each can be True or False => truth table has \(2^{24}\) rows. Try to find a row that is True. Example:

\[
\begin{array}{cccccccc}
F(a_1), F(a_2), \ldots F(a_{12}), G(a_1), G(a_2), G(a_3), \ldots G(a_{12}) \\
T & F & F & F & F & T & F & F
\end{array}
\]
Example

F(a_1), F(a_2), \ldots F(a_{12}), G(a_1), G(a_2), G(a_3), \ldots G(a_{12})

\begin{array}{cccccccc}
T & F & F & F & F & T & F & F \end{array}

We can read off from this a model with 12 elements that satisfies the formula. The same model must satisfy the original (quantified) formula S too.
Concluding

• The proof suggests an algorithm for deciding whether a formula is satisfiable
  – Not satisfiable $\Rightarrow$ no row of the truth table is True
  – Also applicable to logical consequence
  – Implementations exist
• $2^k$ implies Exponential in complexity (though faster methods exist)
• Decidability proofs often tell us something about the **worst-case runtime** of a program
Other FOPL fragments

• For every $n$, it is decidable whether a given formula of FOPL has a model of size $m \leq n$
  – Not proven here
• However, dyadic FOPL is undecidable
  – If time permits, we will prove this later
  – For now, just one observation
Observe:

- Key Theorem does not hold for dyadic FOPL
- Example: the following FOPL sentence does not have a finite model

    $\forall x \exists y (x = x) \& \exists x \forall y (x < y) \& \forall x \forall y \forall z ((x < y \& y < z) \rightarrow x < z) \& \forall x \neg (x < x)$

Why not?