

Checking Graph Model Satisfiability

ϕ -GRAPHS: Given a model G for ϕ (not necessarily a sentence) does $G \models \phi$?

Theorem 1. *For any expression ϕ over Σ_G , the problem ϕ -graphs is in P .*

Proof: By induction on the structure of ϕ .

True when ϕ is an atomic expression of the form $G(x, y)$ or $G(x, x)$.

By induction it is true when $\phi = \neg\psi$, $\psi_1 \vee \psi_2$, $\psi \wedge \psi_2$.

Suppose $\phi = \forall x\psi$.

By induction there is a polynomial algorithm for ψ -GRAPHS.

Repeat the following for each node $v \in G$:

1. Given the model Γ for ϕ , attach the value $x = v$ to get a model for ψ .

Test whether the resulting model satisfies ψ . Answer "yes" if the test satisfies for all $v \in G$, else "no".

Overall time is polynomial in the size of the graph.

Validity and Satisfiability

A first-order expression is **satisfiable** if there exist a model (appropriate to its vocabulary) that satisfies the expression.

A first-order expression ϕ is **valid** if it is satisfied by any model (as long as it is appropriate). Denoted as $\models \phi$.

Example: $\exists y \forall x (x = y + 1) \Rightarrow \forall w \forall z (w = z)$

An expression can be valid due to three reasons: Boolean validity, equality property, quantifier property.

Proposition 1. *An expression is unsatisfiable iff its negation is valid.*

Boolean Validity

Definition 1. [Principal Expressions] *Let ϕ be an expression. Then the principal subexpressions of ϕ consists of:*

1. ϕ is atomic: ϕ
2. ϕ is of the form $\forall x\psi$: ϕ
3. ϕ is of the form $\neg\psi$: same as those of ψ .
4. ϕ is of the form $\psi_1 \vee \psi_2$ or $\psi_1 \wedge \psi_2$: union of the principal subexpressions of ψ_1 and ψ_2 .

Example: $\forall xG(x, y) \wedge \exists xG(x, y) \wedge (G(z, x) \vee \forall xG(x, y))$.

Principal subexpressions: $\forall xG(x, y)$, $\exists xG(x, y)$, $G(z, x)$, $G(x, y)$.

Boolean Validity

Any first order expression ϕ can be considered to be a Boolean expression on its principal subexpressions:
Boolean form of ϕ .

Proposition 2. *If the Boolean form of the expression ϕ is a tautology then ϕ is valid.*

Modus Ponens

Gives a basis for a proof system.

Proposition 3. *If ψ and $\psi \Rightarrow \phi$ are valid then ϕ is valid.*

Equality

Proposition 4. *Suppose that $t_1, \dots, t_k, t'_1, \dots, t'_k$ are terms. Any expression of the following forms is valid:*

1. $t_1 = t_1$

2. $(t_1 = t'_1 \wedge \dots \wedge t_k = t'_k) \Rightarrow f(t_1, \dots, t_k) = f(t'_1, \dots, t'_k)$

3. $(t_1 = t'_1 \wedge \dots \wedge t_k = t'_k) \Rightarrow (R(t_1, \dots, t_k) \Rightarrow R(t'_1, \dots, t'_k))$

Examples:

$$x + 1 = x + 1$$

$$x = 1 \Rightarrow 1 + 1 = x + 1$$

$$x = y \Rightarrow (G(x, x) \Rightarrow G(y, x))$$

Quantifiers

Definition 2. [Substitution] *Let ϕ be an expression, x be a variable, and t be a term. We define the **substitution of t for x in ϕ** , denoted $\phi[x \leftarrow t]$, to be the expression obtained by replacing each free occurrence of variable x by the term t . We substitute t for x only when there is no variable y in t such that some part of ϕ of the form $\forall y\psi$ contains a free occurrence of x .*

Examples:

1. Let $\phi = (x = 1) \Rightarrow \exists x(x = y)$.

$\phi[x \leftarrow y + 1] = (y + 1 = 1) \Rightarrow \exists x(x = y)$.

2. Let $\phi = (x = 1) \Rightarrow \exists y(x = y)$

x is not substitutable by $y + 1$.

Quantifiers

Proposition 5. *Any expression of the form $\forall x\phi \Rightarrow \phi[x \leftarrow t]$ is valid.*

Any expression of the form $\phi[x \leftarrow t] \Rightarrow \exists x\phi$ is valid.

Proposition 6. *If ϕ is valid, then so is $\forall x\phi$.*

If x does not appear free in ϕ , then $\phi \Rightarrow \forall x\phi$ is valid.

Proposition 7. *For all expressions ϕ and ψ , $(\forall x(\phi \Rightarrow \psi)) \Rightarrow (\forall x\phi) \Rightarrow (\forall x\psi)$ is valid.*

Proof: The only way for a model to fail to satisfy this expression is for the following 3 things to happen:

$$M \models (\forall x(\phi \Rightarrow \psi)), M \models \forall x\phi, M \not\models \forall x\psi.$$

That is, there is a u for which $M_{x=u} \not\models \psi$.

But $M_{x=u} \models \phi$ and $M_{x=u} \models \phi \Rightarrow \psi$.
Contradiction.

Prenex Normal Form

Definition 3. [Prenex Normal Form] *An expression is said to be in **Prenex Normal Form** if it consists of a sequence of quantifiers followed by an expression that is free of quantifiers (a Boolean combination of atomic expressions).*

Theorem 2. *Any first-order expression can be transformed to an equivalent one in Prenex normal form.*

Proposition 8. *Let ϕ and ψ be arbitrary first-order expressions. Then*

$$(1) \forall x(\phi \wedge \psi) \equiv (\forall x\phi \wedge \forall\psi).$$

$$(2) \text{ If } x \text{ does not appear free in } \psi, \forall x(\phi \wedge \psi) \equiv (\forall x\phi \wedge \psi).$$

$$(3) \text{ If } x \text{ does not appear free in } \psi, \forall x(\phi \vee \psi) \equiv (\forall x\phi \vee \psi).$$

$$(4) \text{ If } y \text{ does not appear in } \phi, \forall x\phi \equiv \forall y\phi[x \leftarrow y].$$

Example

$$(\forall x(G(x, x) \wedge (\forall yG(x, y) \vee \exists y\neg G(y, y))) \wedge G(x, 0))$$

$$(\forall x(G(x, x) \wedge (\forall yG(x, y) \vee \exists z\neg G(z, z))) \wedge G(w, 0))$$

$$\forall x((G(x, x) \wedge (\forall yG(x, y) \vee \exists z\neg G(z, z))) \wedge G(w, 0))$$

$$\forall x\forall y((G(x, x) \wedge (G(x, y) \vee \exists z\neg G(z, z))) \wedge G(w, 0))$$

$$\forall x\forall y((G(x, x) \wedge \neg(\neg G(x, y) \wedge \forall zG(z, z))) \wedge G(w, 0))$$

$$\forall x\forall y((G(x, x) \wedge \neg(\neg G(x, y) \wedge \forall zG(z, z))) \wedge G(w, 0))$$

$$\forall x\forall y((G(x, x) \wedge \neg\forall z(\neg G(x, y) \wedge G(z, z))) \wedge G(w, 0))$$

$$\forall x\forall y(\neg\forall z(\neg G(x, x) \vee (\neg G(x, y) \wedge G(z, z))) \wedge G(w, 0))$$

$$\forall x\forall y\neg\forall z((\neg G(x, x) \vee (\neg G(x, y) \wedge G(z, z))) \vee \neg G(w, 0))$$

$$\forall x\forall y\exists z\neg((\neg G(x, x) \vee (\neg G(x, y) \wedge G(z, z))) \vee \neg G(w, 0))$$