

Decidability of WS1S and S1S (An Exposition)

William Gasarch-U of MD

Buchi proved that WS1S was decidable.
I don't know off hand who proved S1S decidable.

PART I OF THIS TALK:
WE DEFINE WS1S AND PROVE ITS DECIDABLE

(This is informal since we did not specify the language.)

1. A *Formula* allows variables to not be quantified over. A Formula is neither true or false. Example: $(\exists x)[x + y = 7]$.
2. A *Sentence* has all variables quantified over. Example: $(\forall y)(\exists x)[x + y = 7]$. So a Sentence is either true or false.

(This is informal since we did not specify the language.)

1. A *Formula* allows variables to not be quantified over. A Formula is neither true or false. Example: $(\exists x)[x + y = 7]$.
2. A *Sentence* has all variables quantified over. Example: $(\forall y)(\exists x)[x + y = 7]$. So a Sentence is either true or false.
WRONG- need to also know the domain.
 $(\forall y)(\exists x)[x + y = 7]$ — TRUE if domain is Z , the integers.
 $(\forall y)(\exists x)[x + y = 7]$ — FALSE if domain is N , the naturals.

Variables and Symbols

In our lang

1. The logical symbols \wedge , \neg , (\exists) .
2. Variables x, y, z, \dots that range over N .
3. Variables A, B, C, \dots that range over finite subsets of N .
4. Symbols: $<$, \in (usual meaning), S (meaning $S(x) = x + 1$).
5. Constants: $0, 1, 2, 3, \dots$
6. Convention: We write $x + c$ instead of $S(S(\dots S(x))\dots)$.
NOTE: $+$ is NOT in our lang.

Called WS1S: Weak Second order Theory of One Successor. Weak Second order means quantify over finite sets.

What Does One Successor Mean?

OUR basic objects are NUMBERS. View as UNARY strings, elements of 1^* . SUCC is APPEND 1.

So could view $7 = ((5 \text{ CONCAT } 1) \text{ CONCAT } 1)$.

WHAT IF our basic objects were STRINGS in $\{0, 1\}^*$? Would have TWO SUCC's: APPEND0, APPEND1.

WS1S= Weak Second Order with ONE Successor- just one way to add to a string. Basic objects are strings of 1's.

WS2S= Weak Second order with TWO Successors- two ways to add to a string. Basic objects are strings of 0's and 1's.

WS2S is also decidable but we will not prove this.

An *Atomic Formula* is:

1. For any $c \in \mathbb{N}$, $x = y + c$ is an Atomic Formula.
2. For any $c \in \mathbb{N}$, $x < y + c$ is an Atomic Formula.
3. For any $c, d \in \mathbb{N}$, $x \equiv y + c \pmod{d}$ is an Atomic Formula.
4. For any $c \in \mathbb{N}$, $x + c \in A$ is an Atomic Formula.
5. For any $c \in \mathbb{N}$, $A = B + c$ is an Atomic Formula.

A *WS1S Formula* is:

1. Any atomic formula is a WS1S formula.
2. If ϕ_1, ϕ_2 are WS1S formulas then so are
 - 2.1 $\phi_1 \wedge \phi_2$,
 - 2.2 $\phi_1 \vee \phi_2$
 - 2.3 $\neg\phi_1$
3. If $\phi(x_1, \dots, x_n, A_1, \dots, A_m)$ is a WS1S-Formula then so are
 - 3.1 $(\exists x_i)[\phi(x_1, \dots, x_n, A_1, \dots, A_m)]$
 - 3.2 $(\exists A_i)[\phi(x_1, \dots, x_n, A_1, \dots, A_m)]$

PRENEX NORMAL FORM

A formula is in **Prenex Normal Form** if it is of the form

$$(Q_1 v_1)(Q_2 v_2) \cdots (Q_n v_n)[\phi(v_1, \dots, v_n)]$$

where the v_i 's are either number of finite-set variables, and ϕ has no quantifiers.

Every formula can be put into this form using the following rules

1. $(\exists x)[\phi_1(x)] \vee (\exists y)[\phi_2(y)]$ is equiv to $(\exists x)[\phi_1(x) \vee \phi_2(x)]$.
2. $(\forall x)[\phi_1(x)] \wedge (\forall y)[\phi_2(y)]$ is equiv to $(\forall x)[\phi_1(x) \wedge \phi_2(x)]$.
3. $\phi(x)$ is equivalent to $(\forall y)[\phi(x)]$ and $(\exists y)[\phi(x)]$.

Definition: If $\phi(x_1, \dots, x_n, A_1, \dots, A_m)$ is a WS1S-Formula then $TRUE_\phi$ is the set

$$\{(x_1, \dots, x_n, A_1, \dots, A_m) \mid \phi(x_1, \dots, x_n, A_1, \dots, A_m) = T\}$$

This is the set of $(x_1, \dots, x_n, A_1, \dots, A_m)$ that make ϕ TRUE.

REPRESENTATION

We want to say that *TRUE* is regular. Need to represent $(x_1, \dots, x_n, A_1, \dots, A_m)$.

We just look at (x, y, A) . Use the alphabet $\{0, 1\}^3$.

Below: Top line and the x, y, A are not there- Visual Aid.

The triple $(3, 4, \{0, 1, 2, 4, 7\})$ is represented by

	0	1	2	3	4	5	6	7
x	0	0	0	1	0	*	*	*
y	0	0	0	0	1	*	*	*
A	1	1	1	0	1	0	0	1

Note: After we see 0001 for x we DO NOT CARE what happens next. The *'s can be filled in with 0's or 1's and the string from $\{0, 1\}^3$ above would still represent $(3, 4, \{0, 1, 2, 4, 7\})$.

STUPID STRINGS

What does

	0	1	2	3	4	5	6	7
x	0	0	0	0	0	0	0	0
y	0	0	0	0	1	1	0	1
A	1	1	1	0	1	0	0	1

represent?

This string is **STUPID!** There is no value for x This string does not represent anything!

Our DFA's will have THREE kinds of states: ACCEPT, REJECT, and STUPID. STUPID means that the string did not represent anything because it has a number-variable be all 0's. (It is fine for a set var to be all 0's- that would be the empty set.)

KEY THEOREM

Theorem: For all WS1S formulas ϕ the set $TRUE_\phi$ is regular.

We prove this by induction on the formation of a formula. If you prefer- induction on the LENGTH of a formula.

THEOREM FOR ATOMIC FORMULAS

Lemma: For all WS1S ATOMIC formulas ϕ the set $TRUE_\phi$ is regular.

We prove in class, but not hard.

THEOREM FOR FORMULAS (I)

Assume true for ϕ_1, ϕ_2 — so $TRUE_{\phi_1}$ and $TRUE_{\phi_2}$ are REG.

1. $TRUE_{\phi_1 \wedge \phi_2} = TRUE_{\phi_1} \cap TRUE_{\phi_2}$.
2. $TRUE_{\phi_1 \vee \phi_2} = TRUE_{\phi_1} \cup TRUE_{\phi_2}$.
3. $TRUE_{\neg \phi_1} = \Sigma^* - TRUE_{\phi_1}$.

Good News!: All of the above can be shown using the Closure properties of Regular Langs.

Not Bad News But a Caveat: Must be do carefully because of the stupid states. (Stupid is as stupid does. Name that movie reference!)

Next slides for what to do about quantifiers.

THEOREM FOR FORMULAS (II)

$TRUE_{\phi(x_1, \dots, x_n, A_1, \dots, A_m)}$ is regular.

We want $TRUE_{(\exists x_1)[\phi(x_1, \dots, x_n, A_1, \dots, A_m)]}$ is regular.

Ideas?

THEOREM FOR FORMULAS (II)

$TRUE_{\phi(x_1, \dots, x_n, A_1, \dots, A_m)}$ is regular.

We want $TRUE_{(\exists x_1)[\phi(x_1, \dots, x_n, A_1, \dots, A_m)]}$ is regular.

Ideas?

Use NONDETERMINISM.

Will show you in class.

DFA DECIDABILITY THEOREM

We need the following easy theorem:

Theorem: The following problem is decidable: given a DFA determine if it accepts ANY strings.

DFA DECIDABILITY THEOREM PROOF

Theorem: The following problem is decidable: given a DFA determine if it accepts ANY strings.

Proof: Given $M = (Q, \Sigma, \delta, s, F)$ view as directed graph. Let $n = |Q|$.

$$A_0 = \{s\}$$

For $i = 1$ to n

$$A_{i+1} = A_i \cup \{p \mid (\exists \sigma \in \Sigma)(\exists q \in A_i)[\delta(q, \sigma) = p]\}$$

$L(M) \neq \emptyset$ iff $A_n \cap F \neq \emptyset$.

End of Proof

Theorem: WS1S is Decidable.

Proof:

1. Given a SENTENCE in WS1S put it into the form

$$(Q_1 A_1) \cdots (Q_n A_n) (Q_{n+1} x_1) \cdots (Q_{n+m} x_m) [\phi(x_1, \dots, x_m, A_1, \dots, A_n)]$$

2. Assume $Q_1 = \exists$. (If not then negate and negate answer.)
3. View as $(\exists A)[\phi(A)]$, a FORMULA with ONE free var.
4. Construct DFA M for $\{A \mid \phi(A) \text{ is true}\}$.
5. Test if $L(M) = \emptyset$.
6. If $L(M) \neq \emptyset$ then $(\exists A)[\phi(A)]$ is TRUE.
If $L(M) = \emptyset$ then $(\exists A)[\phi(A)]$ is FALSE.

An Example

We will do the following TOGETHER

$$(\exists A)(\exists x)(\forall y)[x \in A \wedge x \geq 2 \wedge (y \leq x \rightarrow y \in A)].$$

FIRST STEP: rewrite getting rid of $(\forall y)$ and the \rightarrow .

$$(\exists A)(\exists x)\neg(\exists y)\neg[x \in A \wedge x \geq 2 \wedge (y \leq x \rightarrow y \in A)].$$

$$(\exists A)(\exists x)\neg(\exists y)\neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)].$$

(RECALL: $P \rightarrow Q$ is equivalent to $\neg P \vee Q$.)

Atomic Formulas we Need

We need DFA's for the following:

1. $\{(x, y, A) \mid x \in A\}$
2. $\{(x, y, A) \mid x \geq 2\}$
3. $\{(x, y, A) \mid y > x\}$
4. $\{(x, y, A) \mid y \notin A\}$

Atomic Formulas we Need

We need DFA's for the following:

1. $\{(x, y, A) \mid x \in A \wedge x \geq 2\}$
2. $\{(x, y, A) \mid y > x \vee y \notin A\}$
3. $\{(x, y, A) \mid x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)\}$
4. $\{(x, y, A) \mid \neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)]\}$

NOTE- we don't use de Morgans Law- we just complement the DFA.

Atomic Formulas we Need

We need DFA's for

$$\{(x, y, A) \mid \neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)]\}$$

We need DFA's for

1. $\{(x, A) \mid (\exists y)\neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)]\}$
2. $\{(x, A) \mid \neg(\exists y)\neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)]\}$
3. $\{A \mid (\exists x)\neg(\exists y)\neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)]\}$

The Finale!

Take the DFA for

$$\{A \mid (\exists x)\neg(\exists y)\neg[x \in A \wedge x \geq 2 \wedge (y > x \vee y \notin A)]\}.$$

TEST it- does it accept ANYTHING?

If YES then the original sentence is TRUE.

If NO then the original sentence is FALSE.

COMPLEXITY OF THE DECISION PROCEDURE

Given a sentence

$$(Q_1 A_1) \cdots (Q_n A_n) (Q_{n+1} x_1) \cdots (Q_{n+m} x_m) [\phi(x_1, \dots, x_m, A_1, \dots, A_n)]$$

How long will the procedure above take in the worst case?:

COMPLEXITY OF THE DECISION PROCEDURE

Given a sentence

$$(Q_1 A_1) \cdots (Q_n A_n) (Q_{n+1} x_1) \cdots (Q_{n+m} x_m) [\phi(x_1, \dots, x_m, A_1, \dots, A_n)]$$

How long will the procedure above take in the worst case?:

$2^{2^{\cdots n}}$ steps since we do n nondet to det transformations.

VOTE:

1. Much better algorithms are known (e.g., $2^{2^{n^3 \log n}}$.)
2. $2^{2^{\cdots n}}$ is provably the best you can do (roughly).
3. Complexity of dec of WS1S is unknown to science!
4. Stewart/Colbert in 2016!

COMPLEXITY OF THE DECISION PROCEDURE

Given a sentence

$$(Q_1 A_1) \cdots (Q_n A_n) (Q_{n+1} x_1) \cdots (Q_{n+m} x_m) [\phi(x_1, \dots, x_m, A_1, \dots, A_n)]$$

How long will the procedure above take in the worst case?:

$2^{2^{\dots n}}$ steps since we do n nondet to det transformations.

VOTE:

1. Much better algorithms are known (e.g., $2^{2^{n^3 \log n}}$.)
2. $2^{2^{\dots n}}$ is provably the best you can do (roughly).
3. Complexity of dec of WS1S is unknown to science!
4. Stewart/Colbert in 2016!

And the answer is:

$2^{2^{\dots n}}$ **is provably the best you can do (roughly).**

CAN ANYTHING INTERESTING BE STATED IN WS1S?

Is there interesting problems that can be STATED in WS1S?

VOTE:

1. YES
2. NO
3. Stewart/Colbert in 2016!

CAN ANYTHING INTERESTING BE STATED IN WS1S?

Is there interesting problems that can be STATED in WS1S?

VOTE:

1. YES
2. NO
3. Stewart/Colbert in 2016!

Depends what you find interesting.

YES: Extensions of WS1S are used in low-level verification of code fragments. The MONA group has coded this up and used it, though their code uses MANY tricks to speed up the program in MOST cases.

NO: There are no interesting MATH problems that can be expressed in WS1S.

PRESBURGER ARITHMETIC

In our lang

1. The logical symbols $\wedge, \vee, \neg, (\exists), (\forall)$.
2. Variables x, y, z, \dots that range over N .
3. Symbols: $<, +$. Constants: $0, 1, 2, 3, \dots$

Terms and Formulas:

1. Any variable or constant is a term.
2. t_1, t_2 terms then $t_1 + t_2$ is term.
3. t_1, t_2 terms then $t_1 = t_2, t_1 < t_2$ are atomic formulas.
4. Other formulas in usual way: $\wedge, \vee, \neg, (\exists), (\forall)$.

Presb Arith is decidable by TRANSFORMING Pres Arith Sentences into WS1S sentences.

Presb Arithmetic has been used in Code Optimization (using a better dec procedure than reducing to WS1S).

PART II OF THIS TALK:
WE DEFINE S1S AND PROVE ITS DECIDABLE

What is S1S?

Whats The Same: We use the same symbols and define formulas and sentences the same way

Whats Different: We interpret the set variables as ranging over ANY set of naturals, including infinite ones.

Question: Can we still use finite automata?

Essence of WS1S proof:

1. Reg langs closed: UNION, INTER, COMP, PROJ.
2. Emptiness problem for DFA's is decidable.

KEY: We never actually RAN a DFA on any string.

Definition: A B -N DFA as an N DFA. If $x \in \Sigma^\omega$ then x is accepted by B -N DFA M if there is a path such that $M(x)$ hits a final state inf often.

Good News: (PROVE IN GROUPS)

1. B -reg closed: UNION, INTER, PROJ
2. emptiness problem for B -N DFA's is decidable.

NEED B -reg closed under complementation.

GOOD NEWS EVERYONE!

- GOOD NEWS: B -reg IS closed under Complementation.
- GOOD NEWS: That is ALL we need to get S1S decidable.
- GOOD NEWS: It's the only hard step!
- GOOD NEWS: CMSC 452: We are DONE!
- GOOD NEWS: CMSC 858/Math 608 you'll see proof!
- GOOD NEWS: CMSC 858/Math 608 proof uses

GOOD NEWS EVERYONE!

GOOD NEWS: B -reg IS closed under Complementation.

GOOD NEWS: That is ALL we need to get S1S decidable.

GOOD NEWS: It's the only hard step!

GOOD NEWS: CMSC 452: We are DONE!

GOOD NEWS: CMSC 858/Math 608 you'll see proof!

GOOD NEWS: CMSC 858/Math 608 proof uses **Ramsey Theory!**

Definition: A MU -aut M is a $(Q, \Sigma, \delta, s, \mathcal{F})$ where Q, Σ, δ, s are as usual but $\mathcal{F} \subseteq 2^Q$. That is \mathcal{F} is a set of sets of states. M accepts $x \in \Sigma^\omega$ if when you run $M(x)$ the *set of states visited inf often* is in \mathcal{F} .

Easy (IN GROUPS): MU -reg Closed: UNION, INTER, COMP.

RECAP and PLAN:

- ▶ B -reg easily closed: UNION, INTER, PROJ. But COMP seems hard.
- ▶ Mu -reg easily closed: UNION, INTER, COMP. But PROJ seems hard.
- ▶ Our plan if we were to do the entire proof: Show B -reg = Mu -reg.

Theorem: S1S is Decidable.

Proof:

1. Given a SENTENCE in S1S put it into the form

$$(Q_1 A_1) \cdots (Q_n A_n) (Q_{n+1} x_1) \cdots (Q_{n+m} x_m) [\phi(x_1, \dots, x_m, A_1, \dots, A_n)]$$

2. Assume $Q_1 = \exists$. (If not then negate and negate answer.)
3. View as $(\exists A)[\phi(A)]$, a FORMULA with ONE free var.
4. Construct B-NDFA M for $\{A \mid \phi(A) \text{ is true}\}$.
5. Test if $L(M) = \emptyset$.
6. If $L(M) \neq \emptyset$ then $(\exists A)[\phi(A)]$ is TRUE.
If $L(M) = \emptyset$ then $(\exists A)[\phi(A)]$ is FALSE.

COMPLEXITY OF THE DECISION PROCEDURE

Given a sentence

$$(Q_1 A_1) \cdots (Q_n A_n) (Q_{n+1} x_1) \cdots (Q_{n+m} x_m) [\phi(x_1, \dots, x_m, A_1, \dots, A_n)]$$

How long will the procedure above take in the worst case?
 $2^{2^{\cdots n}}$ steps since we do n nondet to det transformations. (This is not quite right- there are some log factors as well.)

CAN ANYTHING INTERESTING BE STATED IN S1S?

Is there interesting problems that can be STATED in S1S?

YES: Verification of programs that are supposed to run forever like Operating systems. Verification of Security protocols.

NO: There are no interesting MATH problems that can be expressed in S1S.

EXTENSIONS

WS1S and S1S are about strings of the form 0^*1 and sets of such strings.

WS2S and S2S are about strings of the form $\{0, 1\}^*$ and sets of such strings.

CAN ANYTHING INTERESTING BE STATED IN WS2S or S2S:

WS2S: YES for verification, no for mathematics.

S2S: YES for Mathematics (finally!). Verification- probably.

I do not think S2S has ever been coded up.