Program Reasoning

6. First-order Theories

Kihong Heo

KAIST

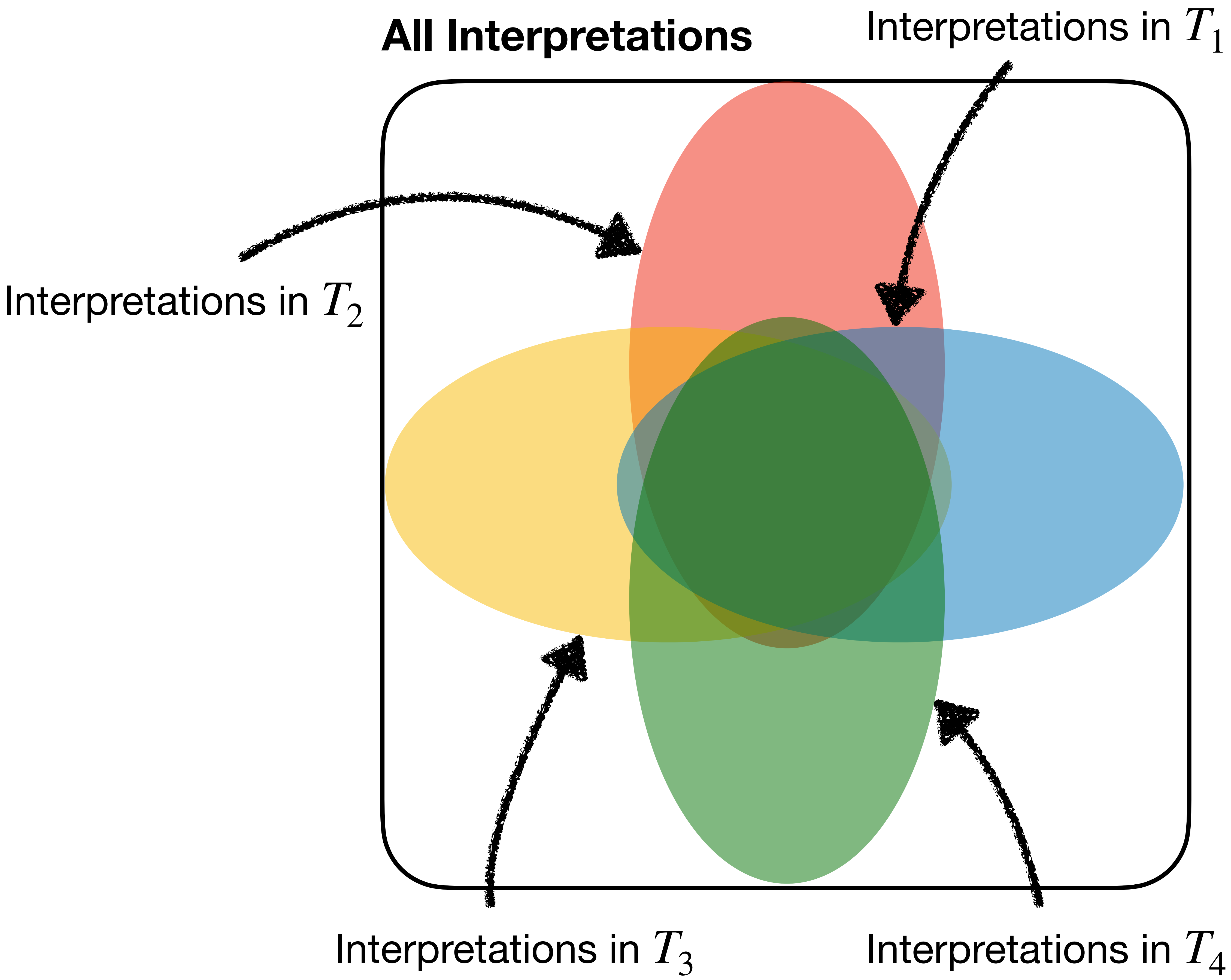
Motivation (1): Interpretation

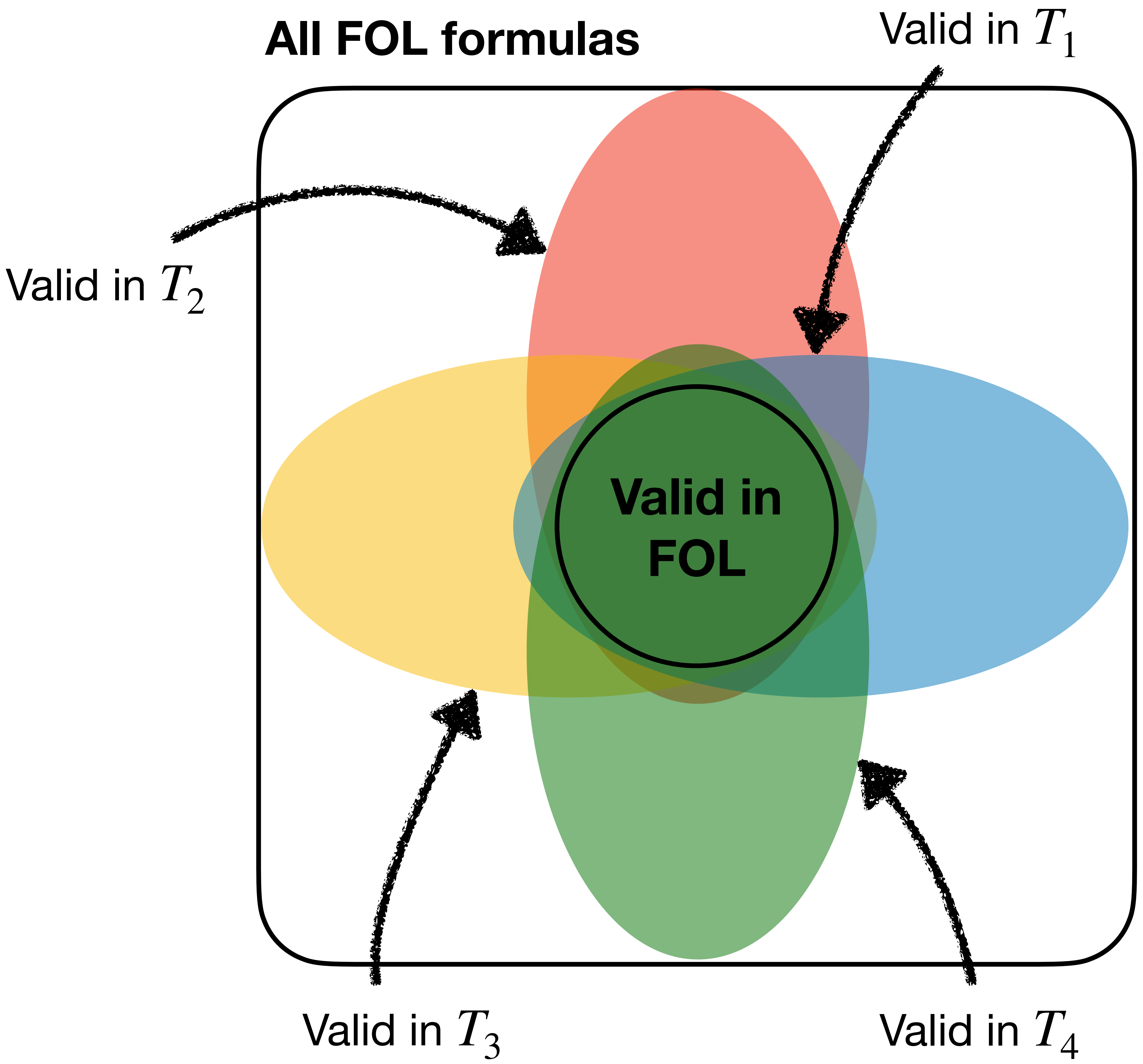
- Full first-order logic: functions and predicates are uninterpreted (i.e., determined by I)
- Validity of full FOL: valid in all interpretations
- Do we really care about all interpretations?
  - For example, $\forall x . x < x + 1$
- NO. Only some specific classes (theory) of interpretations depending on applications
  - Conventional interpretations following axioms
  - E.g., numbers, lists, arrays, strings, etc

Motivation (2): Decidability

- Validity in FOL: undecidable
- Validity in particular theories: sometimes decidable
- Validity in particular fragments of theories: sometimes decidable or efficiently decidable

Validity of Theories





First-order Theory

- Theory T : A restricted class of FOL
  - Signature Σ_T : a set of constants, functions, and predicate symbols
  - Axioms $\mathscr{A}_T$: a set of FOL sentences over Σ_T
- Σ_T -formula: formula constructed from
  - Symbols of Σ_T
  - Variables, logical connectives, and quantifiers
- The symbols of Σ_T does not have prior meaning but the axioms $\mathscr{A}_T$ provide their meaning

Theory of Equality T_E (1)

- $\Sigma_E : \{ = , a, b, c, \ldots, f, g, h, \ldots, p, q, r, \ldots \}$
- Equality “=” is an interpreted predicate symbol
  - The conventional interpretation of “=”
  - The meaning is defined via the axioms
- The other functions, predicates, and constants are uninterpreted
- EUF (Equality with Uninterpreted Functions)

Theory of Equality T_E (2)

- Axioms $\mathscr{A}_E$
  - Reflexivity: $\forall x \,.\, x = x$
  - Symmetry: $\forall x, y \,.\, x = y \rightarrow y = x$
  - Transitivity: $\forall x, y, z \,.\, x = y \land y = z \rightarrow x = z$
  - Function congruence: $\forall \vec{x}, \vec{y} \,.\, (\bigwedge_{i=1}^{n} x_i = y_i) \rightarrow f(\vec{x}) = f(\vec{y})$
  - Predicate congruence: $\forall \vec{x}, \vec{y} \,.\, (\bigwedge_{i=1}^{n} x_i = y_i) \rightarrow (p(\vec{x}) \leftrightarrow p(\vec{y}))$

Example (1)

- $D_I = \{0,1\}$
- Which interpretations of $=$ are allowed in T_E?
  - $\alpha_I(=) = \{\langle 0,1 \rangle, \langle 1,0 \rangle\}$
  - $\alpha_I(=) = \{\langle 0,0 \rangle, \langle 1,1 \rangle\}$
  - $\alpha_I(=) = \{\langle 0,0 \rangle, \langle 0,1 \rangle, \langle 1,0 \rangle, \langle 1,1 \rangle\}$
- Which interpretations of f are allowed in T_E when $\alpha_I(=) = \{\langle 0,0 \rangle, \langle 1,1 \rangle\}$?
  - $\alpha_I(f) = \{0 \mapsto 0,\ 1 \mapsto 1\}$
  - $\alpha_I(f) = \{0 \mapsto 1,\ 1 \mapsto 0\}$

Example (2)

- $D_I = \{0,1,2\}$
- Is the following interpretation of $=$ allowed in T_E?
  - $\alpha_I(=) = \{\langle 0,0\rangle, \langle 0,1\rangle, \langle 1,0\rangle, \langle 1,1\rangle, \langle 2,2\rangle\}$
- Which interpretations of f are allowed in T_E with the above =?
  - $\alpha_I(f) = \{0 \mapsto 0,\ 1 \mapsto 0,\ 2 \mapsto 0\}$
  - $\alpha_I(f) = \{0 \mapsto 1,\ 1 \mapsto 0,\ 2 \mapsto 2\}$
  - $\alpha_I(f) = \{0 \mapsto 1,\ 1 \mapsto 2,\ 2 \mapsto 2\}$

Validity and Satisfiability Modulo Theory

- T-interpretation: an interpretation that satisfies all the axioms of T
  - $I \vDash A$ for every $A \in \mathscr{A}$
- Σ_T -formula F is valid in theory T if all T-interpretations satisfy F
  - F is T-valid or $T \vDash F$
- Σ_T -formula F is satisfiable in theory T if there exists a T-interpretation that satisfies F
  - F is T-satisfiable

Example

- Prove $F : a = b \land b = c \rightarrow g(f(a), b) = g(f(c), a)$ is T_E-valid

First-order Theories for Programs

- Equality
- Integers, rationals, and reals
- Lists
- Arrays
- Pointers
- Bit-vectors
- etc

Theory of Peano Arithmetic (1)

- $\Sigma_{PA} : \{\ 0, 1, + , \cdot , =\ \}$
  - 0 and 1 : constants
  - + (addition) and · (multiplication) are binary functions
  - and = (equality) is a binary predicate

Theory of Peano Arithmetic (2)

- $\mathscr{A}_{PA}$: Axioms of T_{PA}
  - Zero: $\forall x \,.\, \neg(x + 1 = 0)$
  - Successor: $\forall x, y \,.\, x + 1 = y + 1 \rightarrow x = y$
  - Plus zero: $\forall x \,.\, x + 0 = x$
  - Plus successor: $\forall x, y \,.\, x + (y + 1) = (x + y) + 1$
  - Times zero: $\forall x \,.\, x \cdot 0 = 0$
  - Times successor: $\forall x, y, z \,.\, x \cdot (y + 1) = x \cdot y + x$
  - Induction: $F[0] \wedge (\forall x \,.\, F[x] \rightarrow F[x + 1]) \rightarrow \forall x \,.\, F[x]$

An axiom schema for every Σ_{PA}-formula F with one free variable

Theory of Peano Arithmetic (3)

- T_{PA}: a powerful theory for arithmetic over natural numbers
- Natural numbers in T_{PA}
  - $3x + 5 = 2y$ as $(1 + 1 + 1) \cdot x + 1 + 1 + 1 + 1 + 1 = (1 + 1) \cdot y$
- Inequality in T_{PA}
  - $3x + 5 > 2y$ as $\exists z \,.\, z \neq 0 \land 3x + 5 = 2y + z$

Example (1)

- Prove $\exists x, y, z \,.\, x \neq 0 \wedge y \neq 0 \wedge z \neq 0 \wedge x^2 + y^2 = z^2$ is T_{PA}-valid

Example (2)

- Prove $\forall x, y, z \,.\, x \neq 0 \land y \neq 0 \land z \neq 0 \land n > 2 \rightarrow x^n + y^n \neq z^n$ is T_{PA}-valid

Theory of Presburger Arithmetic (1)

- $\Sigma_{\mathbb{N}}$: $\{ 0,1, + , = \}$
  - 0 and 1 : constants
  - + (addition) is a binary function
  - and = (equality) is a binary predicate
- A subset of Σ_{PA} (without multiplication)

Theory of Presburger Arithmetic (2)

- $\mathscr{A}_{\mathbb{N}}$: Axioms of $T_{\mathbb{N}}$
  - Zero: $\forall x\,.\,\neg(x + 1 = 0)$
  - Successor: $\forall x, y, . x + 1 = y + 1 \rightarrow x = y$
  - Plus zero: $\forall x\,.\,x + 0 = x$
  - Plus successor: $\forall x, y\,.\,x + (y + 1) = (x + y) + 1$
  - Induction: $F[0] \wedge (\forall x\,.\,F[x] \rightarrow F[x + 1]) \rightarrow \forall x\,.\,F[x]$
- A subset of $\mathscr{A}_{PA}$

An axiom schema for every Σ_{PA}-formula F with one free variable

Theory of Lists (1)

- Σ_{cons} : $\{\text{cons}, \text{car}, \text{cdr}, \text{atom}, =\}$
  - cons (constructor) is a binary function: “::” in OCaml
  - car (left projector) is a unary function: “List.hd” in OCaml
  - cdr (right projector) is a unary function: “List.tl” in OCaml
  - atom is a unary predicate: $\text{atom}(x)$ is true iff x is a single-element list
  - and = (equality) is a binary predicate

Theory of Lists (2)

- $\mathscr{A}_{cons}$: Axioms of T_{cons}
  - Reflexivity, symmetry, transitivity of T_E
  - Instantiation of the function congruence for cons, car, and cdr
  - Instantiation of the predicate congruence for atom
  - Left projection: $\forall x, y \,.\, \text{car}(\text{cons}(x, y)) = x$
  - Right projection: $\forall x, y \,.\, \text{cdr}(\text{cons}(x, y)) = y$
  - Construction: $\forall x \,.\, \neg\text{atom}(x) \rightarrow \text{cons}(\text{car}(x), \text{cdr}(x)) = x$
  - Atom: $\forall x, y \,.\, \neg\text{atom}(\text{cons}(x, y))$

Example

- Prove $F : \mathsf{car}(a) = \mathsf{car}(b) \land \mathsf{cdr}(a) = \mathsf{cdr}(b) \land \neg\mathsf{atom}(a) \land \neg\mathsf{atom}(b) \rightarrow f(a) = f(b)$ is $T^{=}_{cons}$-valid

1.	I	$\not\models$	F	assumption
2.	I	$\models$	$\mathsf{car}(a) = \mathsf{car}(b)$	$1, \rightarrow, \land$
3.	I	$\models$	$\mathsf{cdr}(a) = \mathsf{cdr}(b)$	$1, \rightarrow, \land$
4.	I	$\models$	$\neg\mathsf{atom}(a)$	$1, \rightarrow, \land$
5.	I	$\models$	$\neg\mathsf{atom}(b)$	$1, \rightarrow, \land$
6.	I	$\not\models$	$f(a) = f(b)$	$1, \rightarrow$
7.	I	$\models$	$\mathsf{cons}(\mathsf{car}(a), \mathsf{cdr}(a)) = \mathsf{cons}(\mathsf{car}(b), \mathsf{cdr}(b))$	2, 3, (function congruence)
8.	I	$\models$	$\mathsf{cons}(\mathsf{car}(a), \mathsf{cdr}(a)) = a$	4, (construction)
9.	I	$\models$	$\mathsf{cons}(\mathsf{car}(b), \mathsf{cdr}(b)) = b$	5, (construction)
10.	I	$\models$	$a = b$	7, 8, 9, (transitivity)
11.	I	$\models$	$f(a) = f(b)$	10, (function congruence)
12.	I	$\models$	$\bot$	6, 11

Theory of Arrays (1)

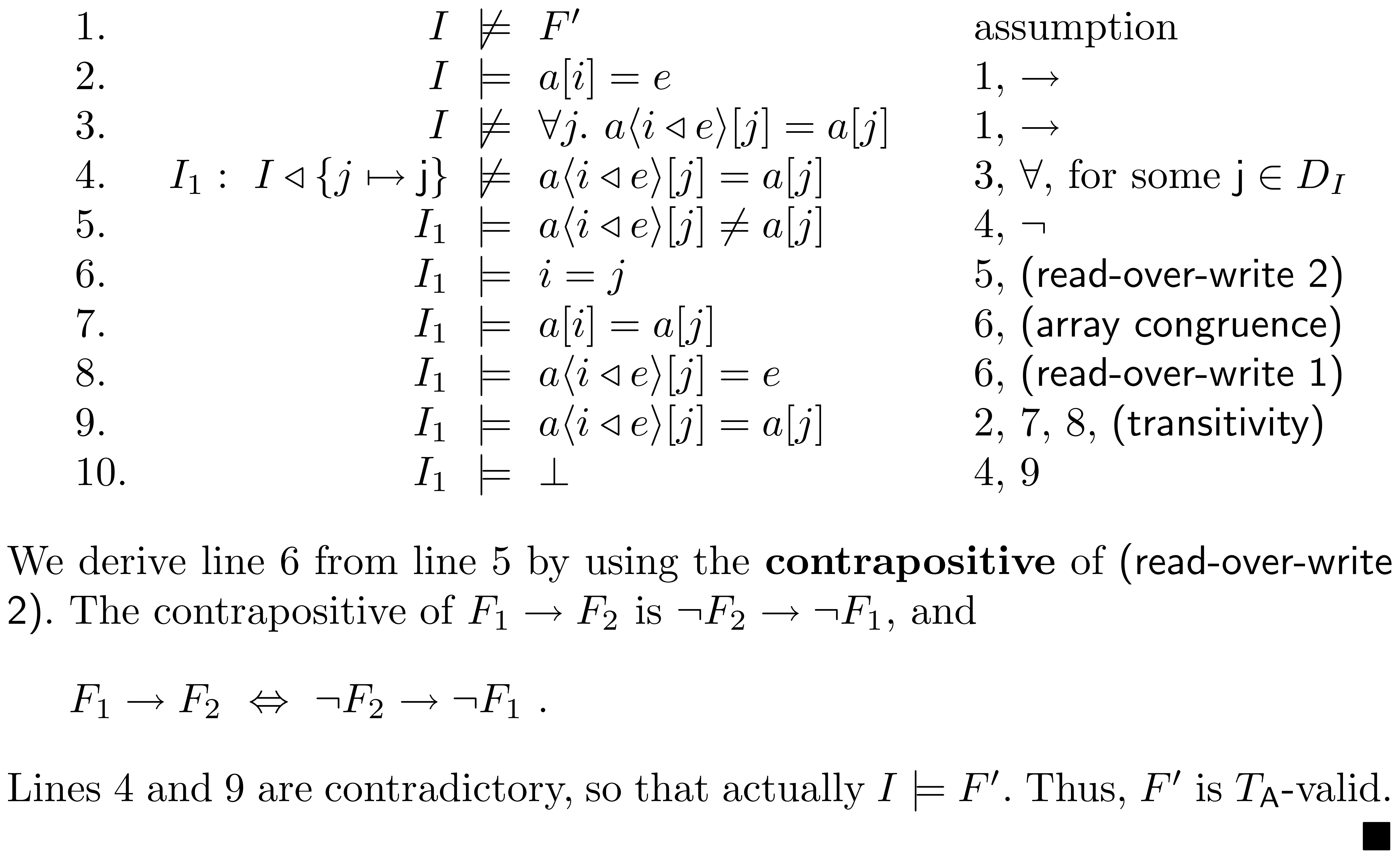
- $\Sigma_A : \{ \cdot[\cdot], \cdot\langle \cdot \triangleleft \cdot \rangle, = \}$
  - $a[i]$ (read) is a binary function: the value of array a at position i)
  - $a\langle i \triangleleft v \rangle$ (write) is a ternary function: the modified array a in which position i has value v
  - and = (equality) is a binary predicate

Theory of Arrays (2)

- Axioms of T_A
  - Reflexivity, symmetry, and transitivity of T_E
  - Array congruence: $\forall a, i, j \,.\, i = j \rightarrow a[i] = a[j]$
  - Read-over-write 1: $\forall a, v, i, j \,.\, i = j \rightarrow a\langle i \triangleleft v \rangle[j] = v$
  - Read-over-write 2: $\forall a, v, i, j \,.\, i \neq j \rightarrow a\langle i \triangleleft v \rangle[j] = a[j]$

Example

- Prove $F : a[i] = e \rightarrow \forall j \, . \, a\langle i \triangleleft e \rangle[j] = a[j]$ is valid



1.	I	$\not\models$	F'	assumption
2.	I	$\models$	$a[i] = e$	1, $\rightarrow$
3.	I	$\not\models$	$\forall j.\ a\langle i \triangleleft e \rangle[j] = a[j]$	1, $\rightarrow$
4.	$I_1 :\ I \triangleleft \{j \mapsto \mathsf{j}\}$	$\not\models$	$a\langle i \triangleleft e \rangle[j] = a[j]$	3, $\forall$, for some $\mathsf{j} \in D_I$
5.	I_1	$\models$	$a\langle i \triangleleft e \rangle[j] \neq a[j]$	4, $\neg$
6.	I_1	$\models$	$i = j$	5, (read-over-write 2)
7.	I_1	$\models$	$a[i] = a[j]$	6, (array congruence)
8.	I_1	$\models$	$a\langle i \triangleleft e \rangle[j] = e$	6, (read-over-write 1)
9.	I_1	$\models$	$a\langle i \triangleleft e \rangle[j] = a[j]$	2, 7, 8, (transitivity)
10.	I_1	$\models$	$\bot$	4, 9

We derive line 6 from line 5 by using the **contrapositive** of (read-over-write 2). The contrapositive of $F_1 \rightarrow F_2$ is $\neg F_2 \rightarrow \neg F_1$, and

$$F_1 \rightarrow F_2 \;\Leftrightarrow\; \neg F_2 \rightarrow \neg F_1 \;.$$

Lines 4 and 9 are contradictory, so that actually $I \models F'$. Thus, F' is T_{A}-valid. ■

Completeness

- A theory T is complete if for every closed Σ_T-formula F, $T \vDash F$ or $T \vDash \neg F$
  - "We must know, we will know" (David Hilbert)
- What happens if a theory is incomplete?
  - "There exists a F such that we don't know either $T \vDash F$ or $T \vDash \neg F$" (Kurt Gödel)
- Gödel's 1st incompleteness theorem: "any theory that includes PA is incomplete"
- Example: T_{PA} is incomplete.

Consistency

- A theory T is consistent if there is at least one T-interpretation
- What happens if a theory is inconsistent?
  - No interpretation satisfies all the axioms of T (there exists a contradiction in the axioms)
  - Both $T \vDash F$ and $T \vDash \neg F$, so $T \vDash \perp$
- Example: $\mathscr{A}_{PA'} = \mathscr{A}_{PA} \cup \{ \forall x \,.\, x + 1 = 0 \}$
  - Both $F : 0 + 1 = 0$ and $\neg F : \neg(0 + 1 = 0)$ are valid
- In a consistent theory T, there does not exist a Σ-formula F s.t. both $T \vDash F$ and $T \vDash \neg F$
- Gödel's 2nd incompleteness theorem:
  "Any theory that includes PA cannot prove its own consistency"

Decidability

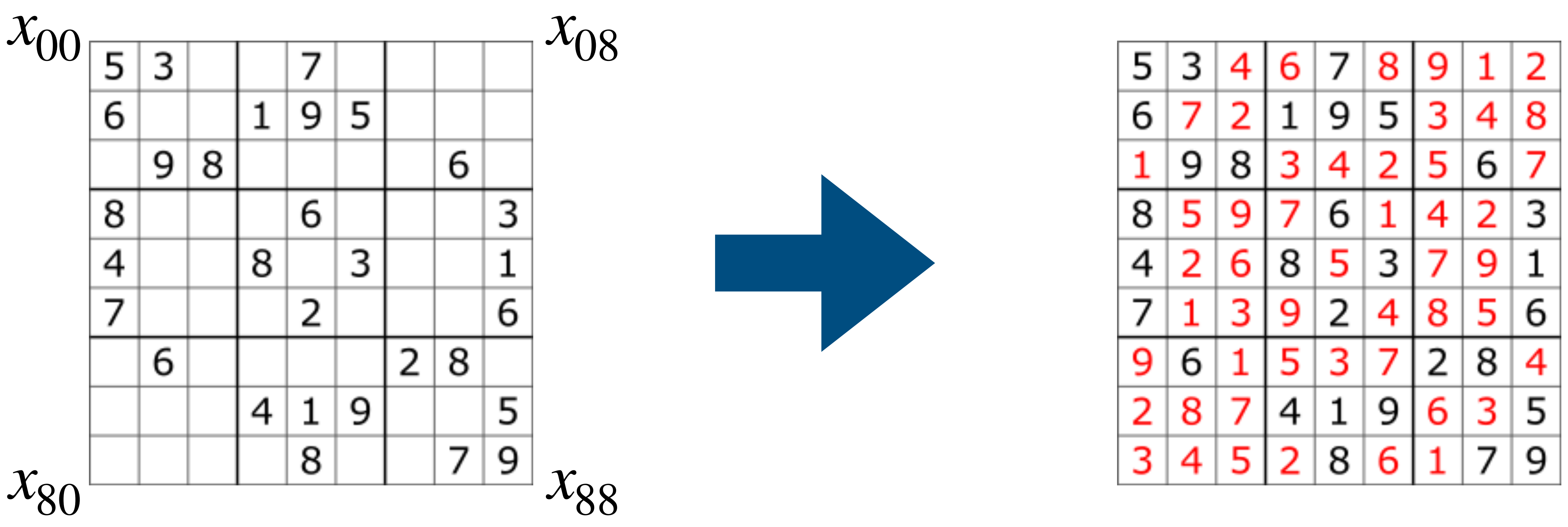
- A theory T is decidable if $T \vDash F$ is decidable for every Σ_T-formula F
  - Always terminating algorithm
  - Says “yes” if F is T-valid, or “no” if F is T-invalid
- Many theories are undecidable
  - E.g., the “empty” theory, theory of equality: undecidable
- Some theories become decidable with further restrictions
  - Quantifier-free fragment: formulae without quantifiers
  - Conjunctive fragment: formulae with only conjunctions

Decidability of Theories

Description	Full	QFF
equality	no	yes
Peano arithmetic	no	no
Presburger arithmetic	yes	yes
linear integers	yes	yes
reals with multiplication	yes	yes
rationals without multplication	yes	yes
recursive data structures	no	yes
acyclic recursive data structures	yes	yes
arrays	no	yes
arrays with extentionality	no	yes

Application: Sudoku

- How to solve Sudoku via SMT?



5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

1. Use numbers 1-9: $\forall 0 \leq i, j \leq 8.\ 1 \leq x_{ij} \leq 9$
2. Don't repeat any numbers in a row: $\forall 0 \leq i \leq 8.\ x_{i0} \neq x_{i1} \neq \cdots \neq x_{i8}$
3. Don't repeat any numbers in a column: $\forall 0 \leq i \leq 8.\ x_{0i} \neq x_{1i} \neq \cdots \neq x_{8i}$
4. Don't repeat any numbers in a square: …

Prove (1 ⋀ 2 ⋀ 3⋀ 4) is satisfiable!

* https://en.wikipedia.org/wiki/Sudoku

Application: Symbolic Execution

- How to find a crashing input via SMT?

```
void f(int x, int y) {
  int z = 2 * x;
  if (y > 0) {
    int w = 2 * y;
    if (w + x == 0)
      crash();
  }
}
```

The program crashes if “crash()” is reachable.
Is this crash possible? What are the values of x and y that cause the crash?

Prove $z = 2 \times x \land y > 0 \land w = 2 \times y \land w + x = 0$ **is satisfiable!**

Application: Translation Validation (1)

- Compiler bugs

```
$ clang -O0 input.c
$ ./a.out
1
$ clang -O1 input.c
$ ./a.out
Aborted (core dumped)
```

```
# without optimization
$ v8 test.js
true
# with optimization
$ v8 test.js
false
```

Application: Translation Validation (2)

- How to check the correctness of a compilation via SMT?

```
# before optimization
let f(x) =
  let y = 1 in
  if x = y then 1
  else x
```

```
# after optimization
let f(x) = x
```

The translation is correct if, for all inputs, the return values of P_1 and P_2 are the same

$\iff$ The translation is incorrect if there exists an input such that the return values of P_1 and P_2 are different

1. $y_{src} = 1 \land r_{src} = (\text{if } x = y_{src} \text{ then } 1 \text{ else } x)$
2. $r_{tgt} = x_{tgt}$
3. $r_{src} \neq r_{tgt}$

Prove (1 /\ 2 /\ 3) is unsatisfiable!

Summary

- First-order theories: instances of FOL
    - Restrict interpretations using axioms
- Many useful theories for program reasoning
    - E.g., equality, integers, arrays, pointers, etc
- Some theories are decidable but some are not
- Many interesting applications
    - E.g., puzzle, bug-finding, verification, etc