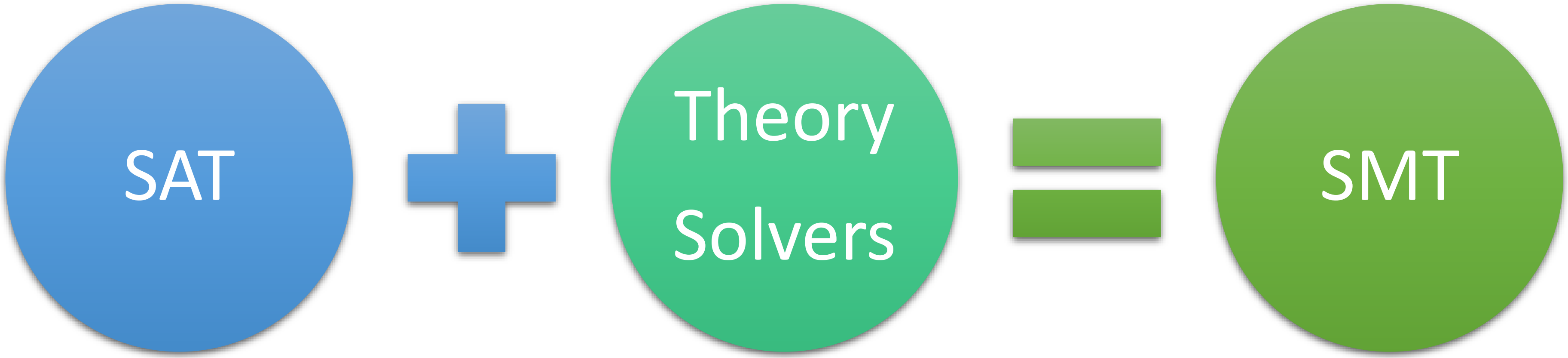
Core Theories

Nikolaj Bjørner, Microsoft Research, RiSE
TU Wien 2025

A Laura Kovacs guest-lecture production

CDCL(T)

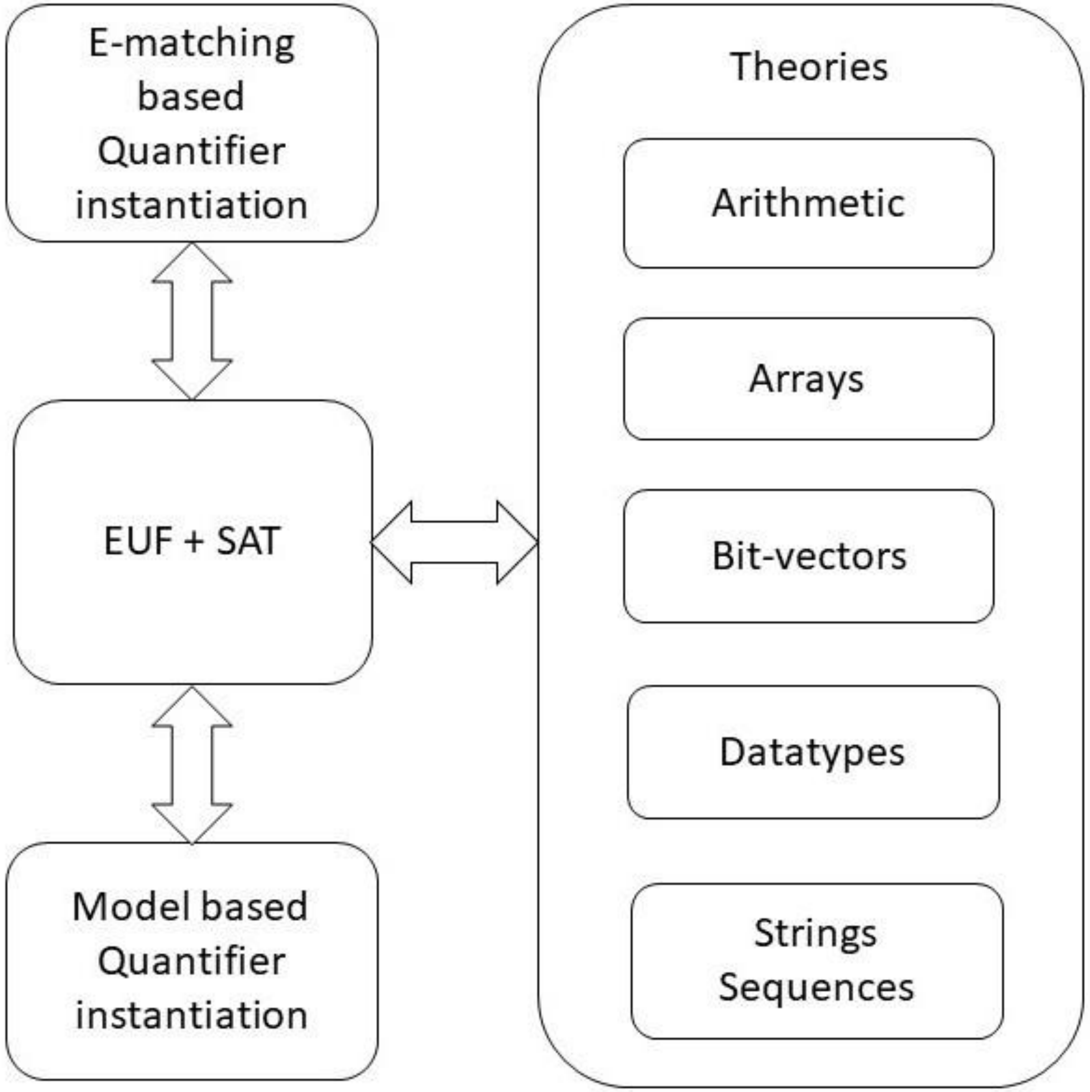


Case Analysis

- Equality + UF
- Arithmetic
- Bit-vectors
- …

Core Decision Procedures

Z3 overview

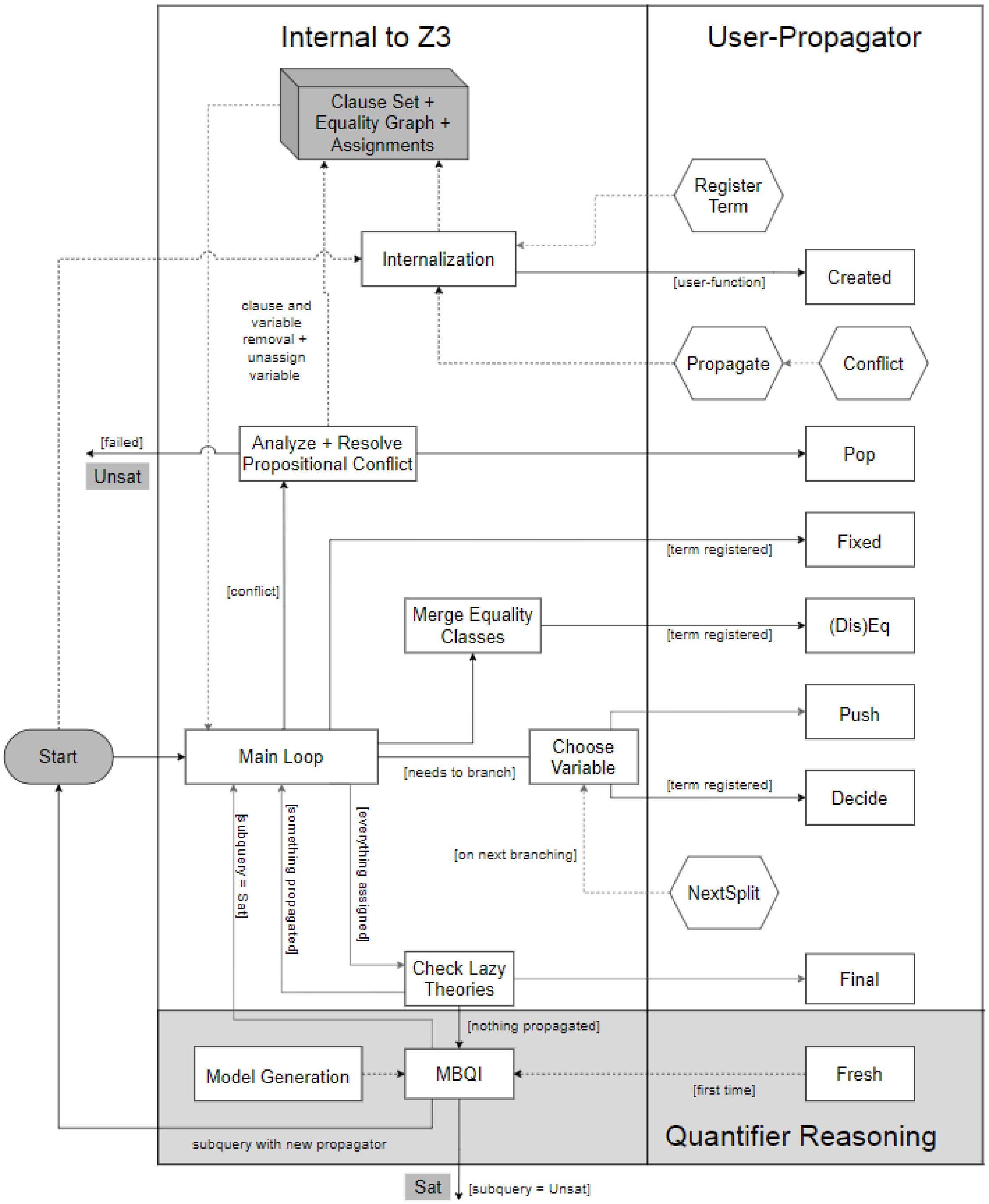

E-matching based Quantifier instantiation
EUF + SAT
Model based Quantifier instantiation
Theories
Arithmetic
Arrays
Bit-vectors
Datatypes
Strings Sequences

CDCL(T)

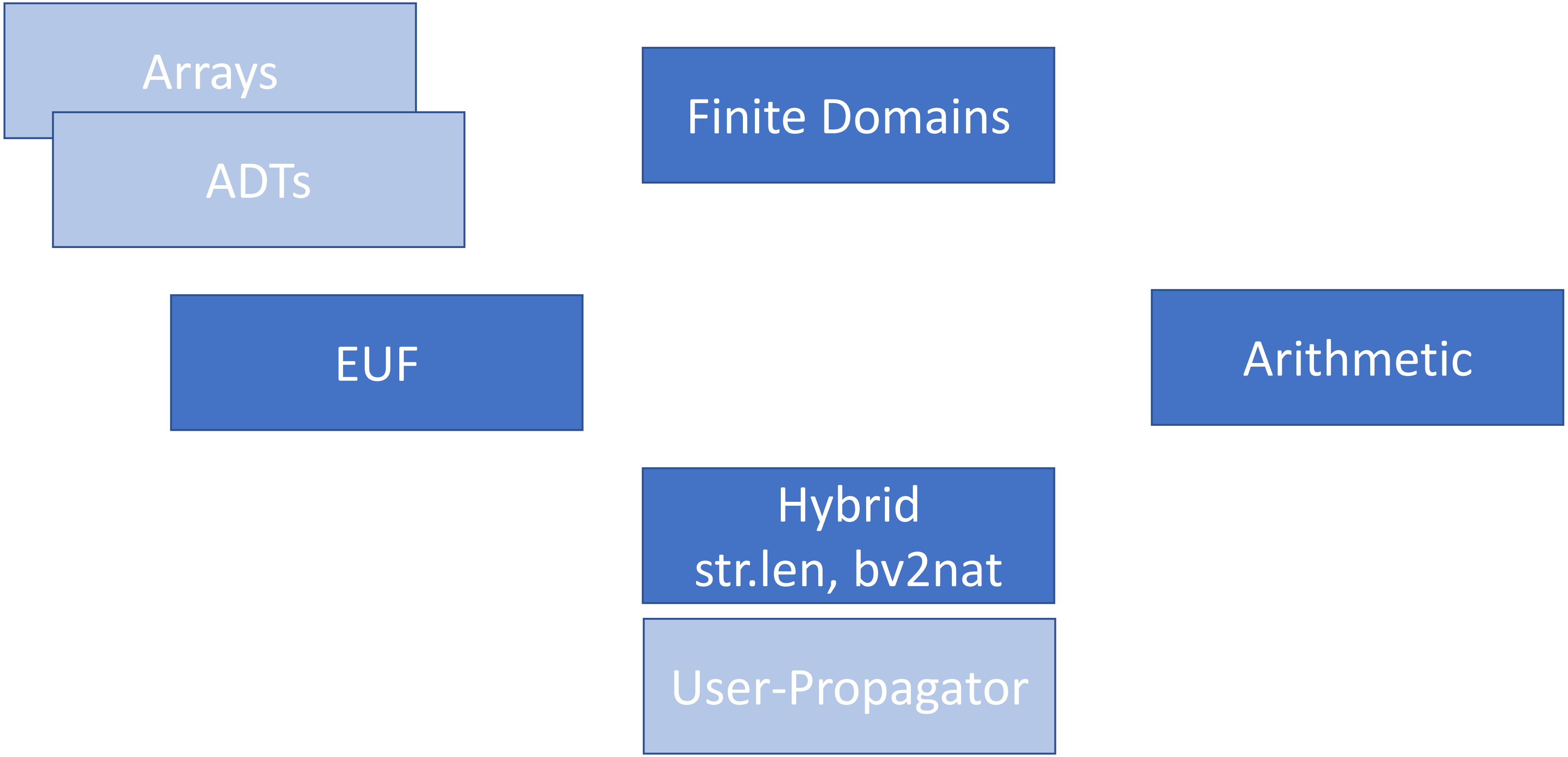
```
def CDCL():

  while True:
    if [] in clauses:         return UNSAT
    elif in_conflict():       learn(); backtrack()
    elif not free_vars:       if theory.delay_propagate() return SAT
    elif should_propagate(): propagate(); theory.propagate()
    elif should_simplify():  simplify(); theory.simplify()
    elif should_restart():   restart()
    elif should_gc():        gc(); theory.gc()
    else:
      theory.push()
      var  = choose_var(free_vars)
      sign = choose_sign(var)
      assign(var, sign)
      theory.assign(var, sign)
```

Custom Theories

- fixed: The CDCL core assigned a boolean/bit-vector value to a registered expression.
- eq: The EUF solver merged two equivalence classes. The two merged representatives will be reported.
- created: A new instance of a function symbol is encountered. e.g., $f(x)$ was instantiated to $f(a)$.
- final: The solver got a consistent assignment to all boolean variables. All theories get the final chance to intervene.
- Further: push, pop, fresh, decide, and diseq



Classification



Finite Domain Theories

```
v = BitVec('v',32)
mask = v >> 31
prove(If(v > 0, v, -v) == (v + mask) ^ mask)
```

```
p, q, r, u = Bools('p q r u')
solve(AtMost(p, q, r, 1), u,
      Implies(u, AtLeast(And(p, r), Or(p, q), r, 2)))
```

```
Color, (red, green, blue) = EnumSort('Color', ['red','green','blue'])
clr = Const('clr', Color)
solve(clr != red, clr != green)
```

Finite Domains and CDCL(T)

```
v = BitVec('v',32)
mask = v >> 31
prove(If(v > 0, v, -v) == (v + mask) ^ mask)
```

Compile to SAT

Compile to SAT + E

Word level

Finite Domains and CDCL(T)

$$p + q + r \leq 1 \wedge$$
$$(u \Rightarrow (p \wedge q) + (p \vee q) + r \geq 2)$$

```
p, q, r, u = Bools('p q r u')
solve(AtMost(p, q, r, 1), u,
      Implies(u, AtLeast(And(p, r), Or(p, q), r, 2)))
```

Compile to CDCL

Compile to CDCL + PB propagation

EUF

The *empty theory* of first-order logic.

$$\frac{}{s \simeq s}\ \text{refl} \qquad \frac{s \simeq t \qquad t \simeq u}{s \simeq u}\ \text{trans}$$

$$\frac{t \simeq s}{s \simeq t}\ \text{symm} \qquad \frac{ts_1 \simeq ts_1', \ldots, ts_k \simeq ts_k'}{f(ts) \simeq f(ts')}\ \text{cong}$$

EUF

$a = f(f(a)), \quad a = f(f(f(a))), \quad a \neq f(a)$

- Produce Proofs
- Incremental Updates
- Propagate Literals

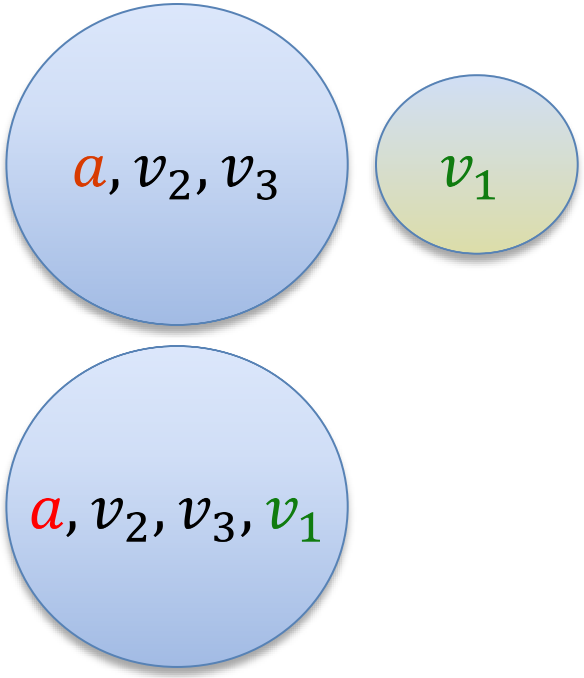
$a = v_2, a = v_3, a \neq v_1,$

$v_1 \equiv f(a), v_2 \equiv f(v_1), v_3 \equiv f(v_2)$

Step 1: Equivalence classes from equalities

Step 2: Apply Congruence Rule:

$a \simeq v_2$ implies $f(a) \simeq f(v_2)$: $\quad v_1 \simeq v_3$



Union Find

E-graph

EUF – data-structure

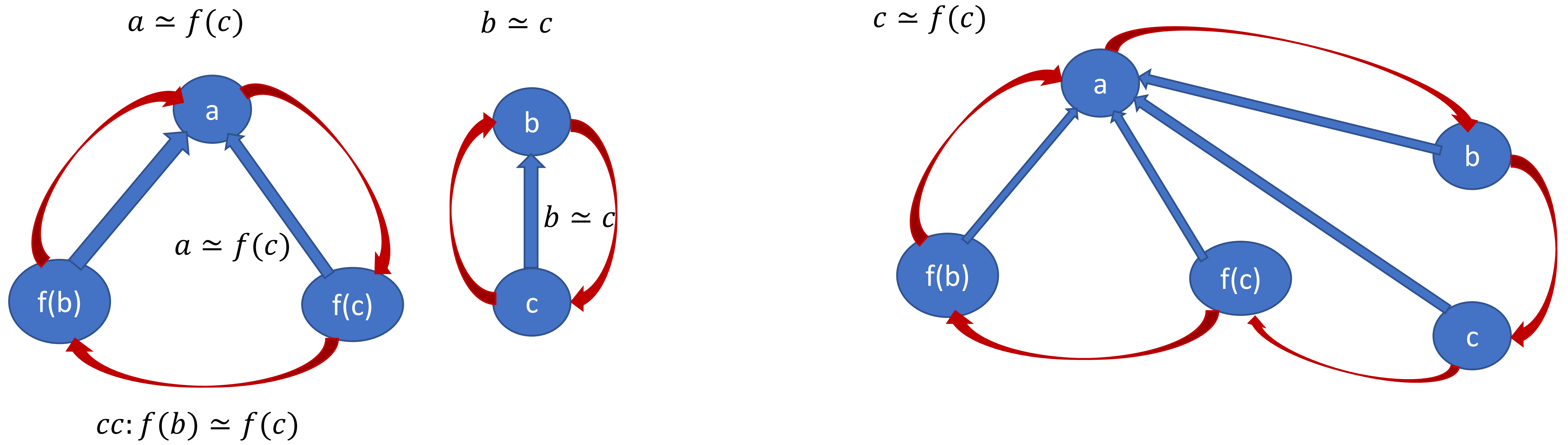
- E-Node:

$$
n : \langle \begin{array}{lll}
f : & Func & \text{function symbol} \\
ts : & N^* & \text{arguments} \\
\mathit{find} : & N & \text{link to representative} \\
P : & N^* & \text{list of parents} \\
cg : & N & \text{congruence representative} \\
j : & null \mid Just \times N & \text{pointer to justification and node}
\end{array} \rangle
$$

- Union-find: $find(n)$ - set $n \leftarrow n.\mathit{find}$ until $n = n.\mathit{find}$.
- etable: $(n.f, find(n.ts)) \mapsto cg$

EUF – union-find w. path compression, siblings

- **Z3 uses path compression** to ensure roots are within a single hop of *find*
- Maintain separate singly linked cyclic list of siblings



EUF - internalize

$$f(g(a), g(a), a)$$

$$\begin{array}{lcl}
n_1 & := & \langle f = a,\ ts = [],\ \mathit{find} = (n_1, n_1, 1),\ P = [n_2, n_3],\ cg = n_1,\ j = \mathit{null}\,\rangle \\
n_2 & := & \langle f = g,\ ts = [n_1],\ \mathit{find} = (n_2, n_2, 1),\ P = [n_3],\ cg = n_2,\ j = \mathit{null}\,\rangle \\
n_3 & := & \langle f = f,\ ts = [n_2, n_2, n_1],\ \mathit{find} = (n_3, n_3, 1),\ P = [],\ cg = n_3,\ j = \mathit{null}\,\rangle
\end{array}$$

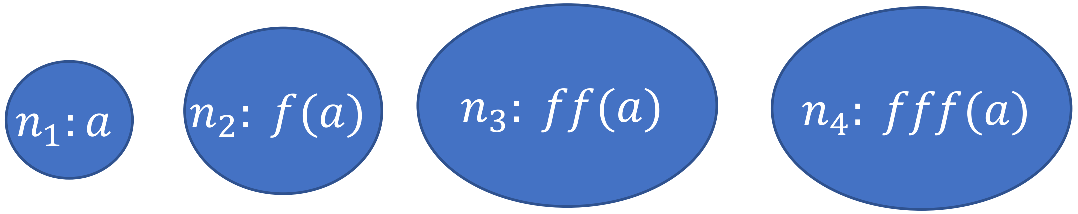
Terms are “hash-consed”

Roots are initialized to self

etable: $[a \mapsto n_1, g([n_1]) \mapsto n_2, f([n_2, n_2, n_1]) \mapsto n_3]$

EUF - merge

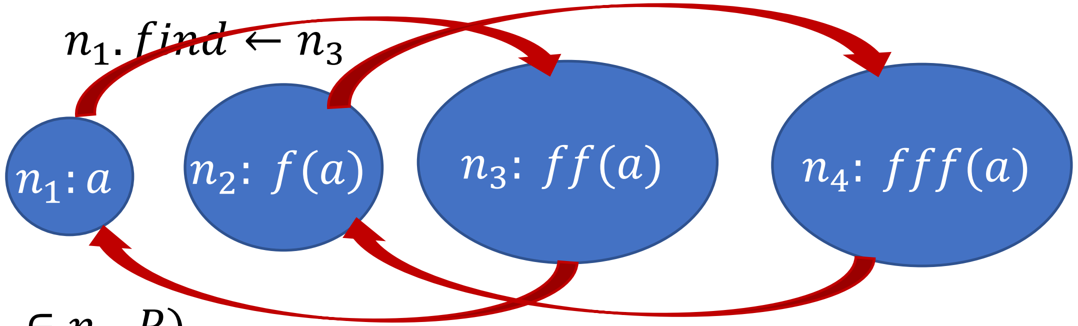
$\boxed{a = f(f(a)),}$ $\quad a = f(f(f(a))), \quad a \neq f(a)$



$merge(n_1, n_3)$

$etable[f, n_1] \leftarrow null$ (since $n_2 \in n_1.P$)

$n_1.find \leftarrow n_3$



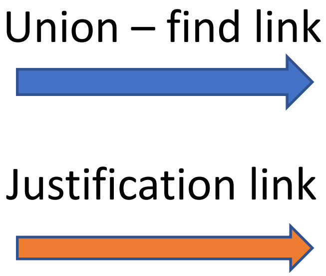
($n_2 \in n_1.P$)

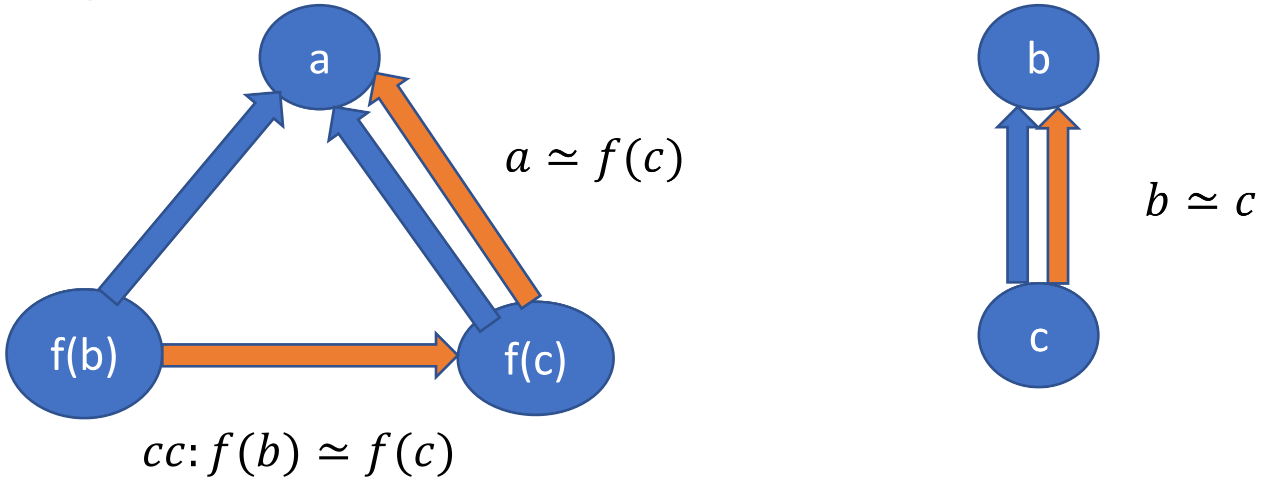
$n_2.cg \leftarrow etable[f, root(n_1)] = n_4$

add $\langle n_2, n_4 \rangle$ to *tomerge*

Roots	$r_1 \leftarrow root(n_1), r_2 \leftarrow root(n_2)$
	assume $r_1 \neq r_2$
	assume $r_1.sz \leq r_2.sz$
Erase	**for each** $p \in r_1.P$ **where** $p.cg = p$:
	erase $etable[p.f, root(p.ts)]$
Update Root	$r_1.find \leftarrow r_2$
Justify	justify(r_1, r_2, j)
Insert	**for each** $p \in r_1.P$:
	if $etable[p.f, root(p.ts)] = null$ **then**
	$etable[p.f, root(p.ts)] \leftarrow p$
	$p.cg \leftarrow etable[p.f, root(p.ts)]$
	if $p.cg = p$ **then**
	append p to $r_2.P$
	else
	add $\langle p.cg, p, cc \rangle$ to *tomerge*

EUF - Justifications

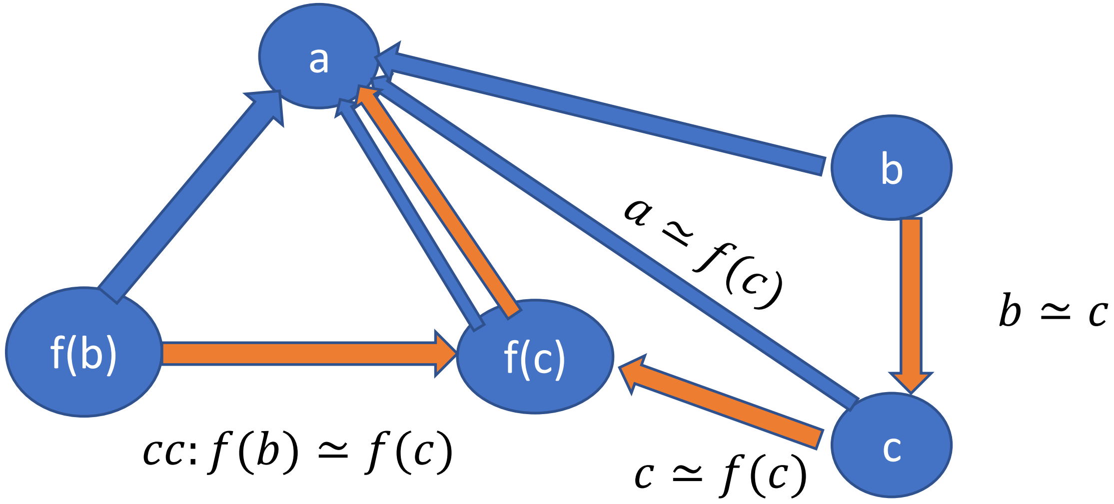




After merge $c \simeq f(c)$

$root(c) \leftarrow root(f(c))$

Justification path from
old root(c) to c got reversed



EUF - internals

- Nodes from Boolean literals contain fields:
  - value : { true, false, undef}
  - boolVar : a number referring to Boolean variable as known by SAT solver

$a \neq f(a)$

$$n_5 = \langle \simeq, [n_1, n_2], (n_5, n_5, 1), P = \epsilon, cg = n_5, j = nil, value = false,\ boolVar = 27 \rangle$$

- Equality nodes are *special:* When n1, n2 are merged, the parent n5 is equality, value = false -> conflict

EUF – internals: equalities and values

Values: If a node comes from a term that denotes a value (5, 42, 2/3, cons(1,nil)), it is always a root

- When two roots with terms based on different values are merged -> conflict

Who reasons about equalities of Booleans? EUF vs. CDCL

- E-nodes based on Bool do by default not merge with other nodes.
- Default is overridden if E-node occurs under a non-connective.

E-graph propagates equalities to theories and Booleans to SAT

$n_5 = \langle \simeq, [n_1, n_2], (n_5, n_5, 1), P = \epsilon, cg = n_5, j = nil, value = undef,\ boolVar = 27 \rangle$

$n_1.find \leftarrow n_2$ -> value is set to true and assignment to boolVar 27 is propagated to CDCL core

Arrays

Reducible reduce to base theories

```
A = Array('A', IntSort(), IntSort())
solve(A[x] == x, Store(A, x, y) == A, x != y)
```

```
solve(A[x] == x, Store(A, x, y) == A, x != y,
      Store(A, x, y)[x] == y, ...)
```

Compile into EUF

UNSAT

Arrays

Reducible reduce to base theories

```
A = Array('A', IntSort(), IntSort())
solve(A[x] == x, Store(A, x, y) == A)
```

Convert EUF solution to solution over arrays

Compile into EUF

```
solve(A[x] == x, Store(A, x, y) == A,
      Store(A, x, y)[x] == y,
      Store(A, x, y)[Diff(Store(A, x, y),A)] == A[Diff(Store(A, x, y),A)],
      Or(x != Diff(Store(A, x, y),A), y == Store(A, x, y)[Diff(Store(A, x, y),A)]))
```

A Solver for Unicode Characters

Unicode Theory

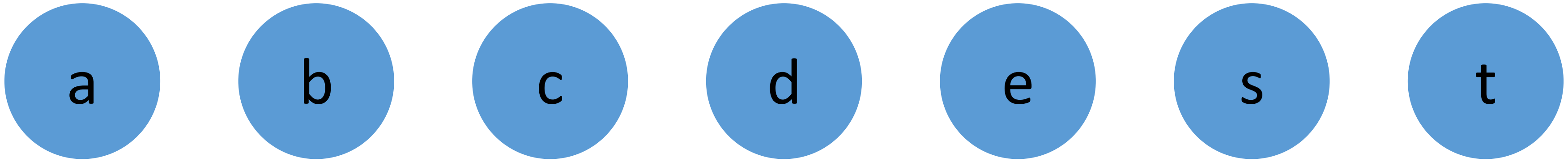
$\langle U, \leq, =: U \times U \rightarrow Bool \rangle \quad |U| = 196608$

Operator $\leq$ is used sparingly

So common case is theory of $=, \neq$

- Engine: Union-find + Lazy reduction to bit-vectors
- Inferior alternatives: pure bit-vectors, linear arithmetic, difference arithmetic

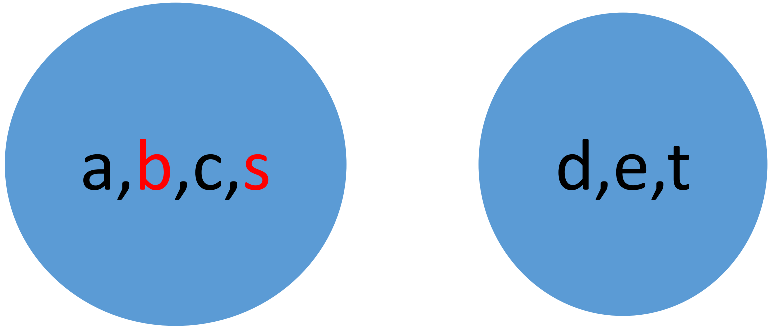
A Solver for $\langle U, \leq, =: U \times U \rightarrow Bool \rangle$ $|U| = 196608$



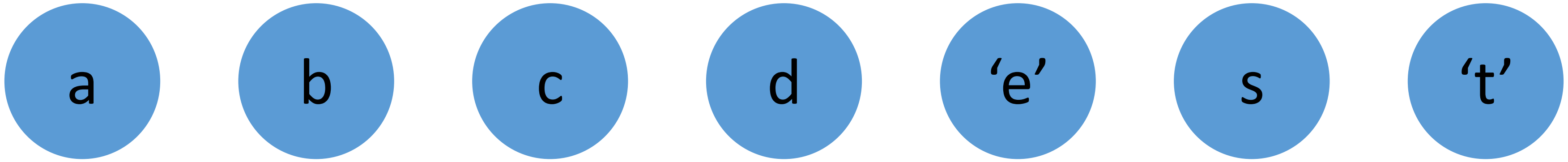
Equality

$a = b,\ b = c,\ d = e,\ b = s,\ d = t,\ a \neq e,\ a \neq s$

Union Find



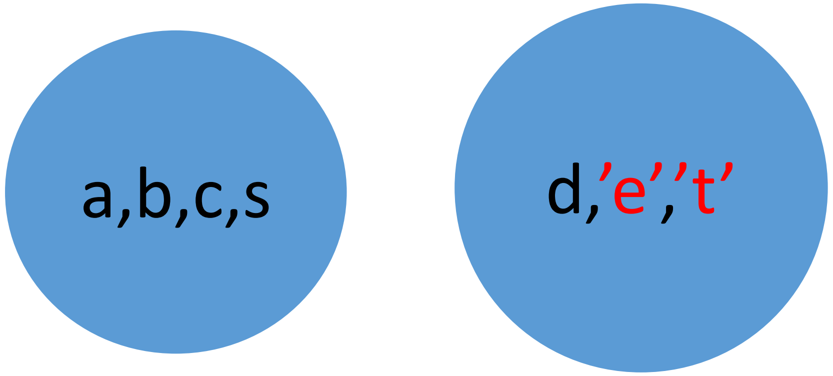
A Solver for $\langle U, \leq, =: U \times U \rightarrow Bool \rangle$ $|U| = 196608$



Equality

Union Find

a = b, b = c, d = ‘e’, b = s, d = ‘t’, a≠ e



A Solver for $\langle U, \leq, =: U \times U \rightarrow Bool \rangle$ $|U| = 196608$

$$a \leq b$$

Inequality

Bit-blasting

$$a[17:0] \leq b[17:0] \leftrightarrow \begin{matrix} (a[17] \rightarrow b[17]) \\ \wedge \begin{pmatrix} (a[17] \leftrightarrow b[17]) \rightarrow \\ a[16:0] \leq b[16:0] \end{pmatrix} \end{matrix}$$

$$\vdots$$

$$a[0:0] \leq b[0:0] \leftrightarrow (a[0] \rightarrow b[0])$$

A Solver for $\langle U, \leq, =: U \times U \to Bool \rangle$ $|U| = 196608$

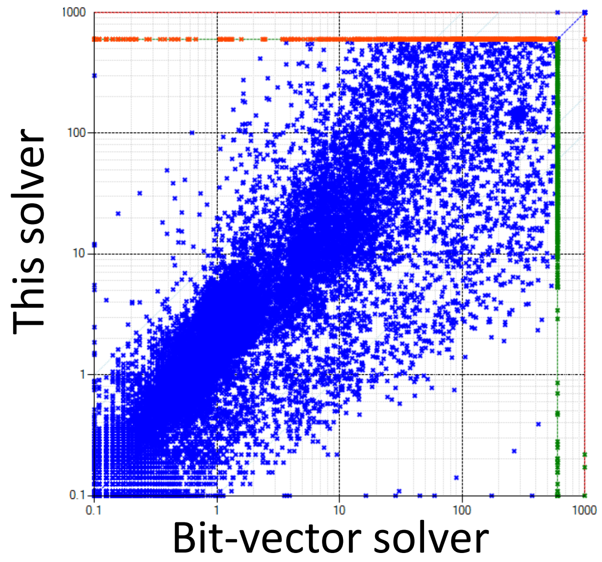
Combining two views

$$a = b$$

Equality View

$$a \leq b$$

Bit-Blast View

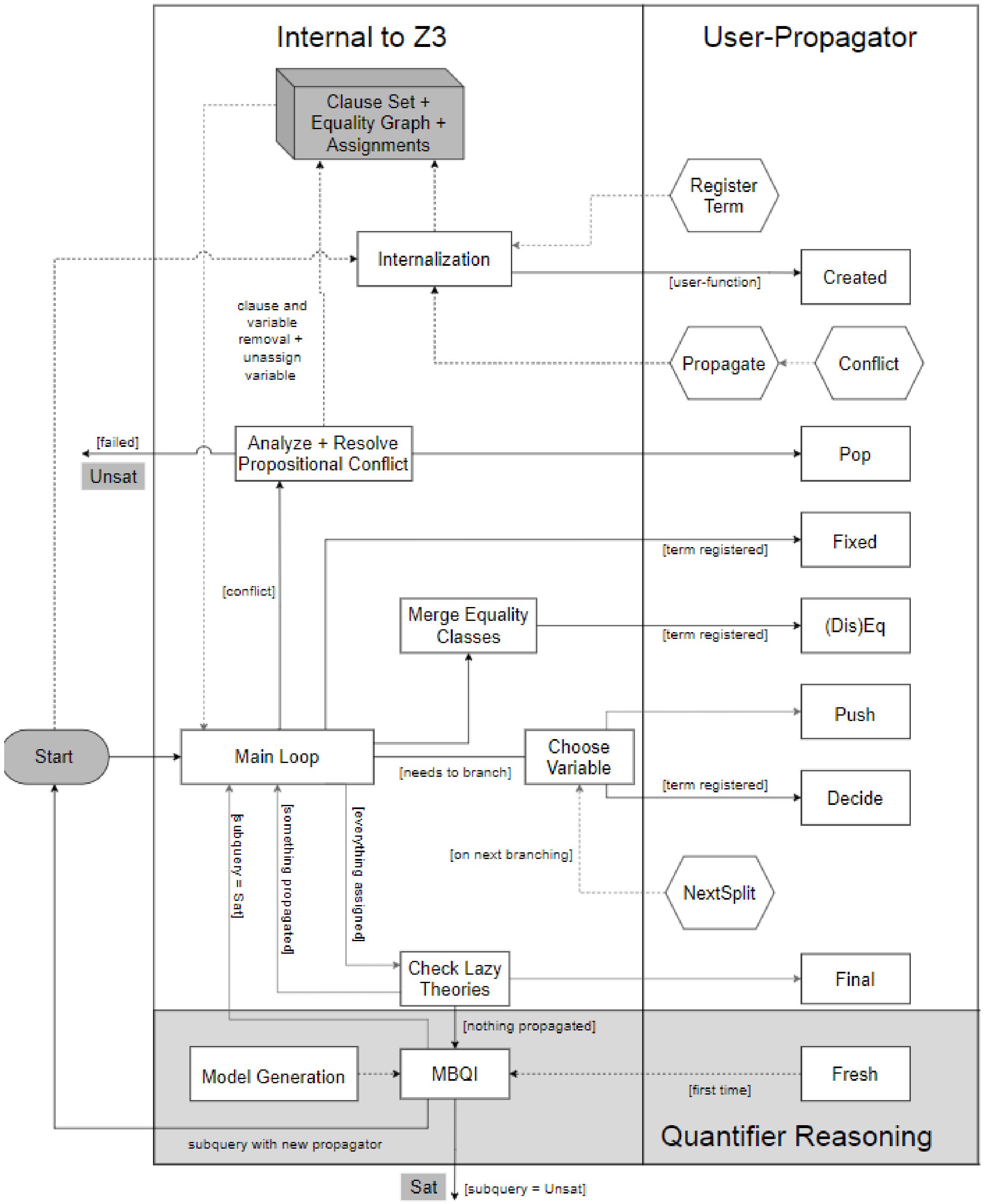


$$\langle U, \leq, =: U \times U \to Bool, bv2char : Bool^{18} \to U,$$
$$[0]: U \to Bool, [1]: U \to Bool, \ldots, [17]: U \to Bool \rangle$$

$$bv2char(a[17], \ldots, a[0]) = a$$

Custom Theories

- fixed: The CDCL core assigned a boolean/bit-vector value to a registered expression.
- eq: The EUF solver merged two equivalence classes. The two merged representatives will be reported.
- created: A new instance of a function symbol is encountered. e.g., $f(x)$ was instantiated to $f(a)$.
- final: The solver got a consistent assignment to all boolean variables. All theories get the final chance to intervene.
- Further: push, pop, fresh, decide, and diseq



Finite Sets

Let us develop a solver for finite sets

Core functionality

- Determine feasibility of conjunctions

$S_i = T_i, S_i \neq T_i, \quad x_j \in S_j, x_j \notin S_j,$

$S_i \subseteq T_i, S_i \subsetneq T_i, \quad S_i \subset T_i, S_i \not\subset T_i$ + Arithmetic over |S|

Enforce that all operators have consistent interpretations.

- Example: If |S| = 5, then S really has precisely 5 elements.

Representation

- Every term of type $Set\ \alpha$ is tracked by finite_set_theory by a *theory variable*

$$v_1 \equiv \emptyset, v_2 \equiv X, v_3 \equiv Y, v_4 \equiv v_2 \cap v_3, v_5 \equiv x \in v_4$$

Consistent interpretations are enforced using *theory axioms*

Theory Axioms

Base

$$x \notin \emptyset$$
$$x \in S \cup T \Leftrightarrow x \in S \vee x \in T$$
$$x \in S \cap T \Leftrightarrow x \in S \wedge x \in T$$
$$x \in S \setminus T \Leftrightarrow x \in S \wedge x \notin T$$
$$x \in \{y\} \Leftrightarrow x = y$$
$$s \neq t \Rightarrow \delta(s,t) \in s \neq \delta(s,t) \in t$$

Filters

$$x \in S \Rightarrow f(x) \in map(f,S)$$
$$x \in map(f,S) \Rightarrow map^{-1}(f,x,S) \in S \wedge f\left(map^{-1}(f,x,S)\right) = x$$
$$x \in select(p,S) \Leftrightarrow x \in S \wedge p(x)$$

Range

$$x \in [lo,hi] \Leftrightarrow lo \leq x \leq hi$$

Minimality

- Express map using select

Use Built-in functions for existential axioms

$$s \neq t \Rightarrow \exists x \,.\, x \in s \neq x \in t$$

Skolemize:

$$s \neq t \Rightarrow \delta(s,t) \in s \neq \delta(s,t) \in t$$

$$x \in map(f,S) \Rightarrow \exists y \,.\, y \in S \wedge f(y) = x$$

Skolemize:

$$x \in map(f,S) \Rightarrow map^{-1}(f,x,S) \in S \wedge f\left(map^{-1}(f,x,S)\right) = x$$

Theory Axiom Saturation – for Base

$$\forall x\ S, T\ .\ x \in S \cup T \Leftrightarrow x \in S \vee x \in T$$

$$\frac{x \in U,\ U \sim S \cup T}{x \in S \cup T \Leftrightarrow x \in S \vee x \in T} \qquad \frac{x \in U \equiv v \qquad U \sim S \qquad S \cup T \in parents(U)}{x \in S \cup T \Leftrightarrow x \in S \vee x \in T}$$

After axiom saturation

$M(s) := \{ x \mid x \in s \equiv \top \}$ is a consistent interpretation

Because after saturation

$x \sim y, s \sim t \ \Rightarrow x \in s \Leftrightarrow y \in t$

Theory axioms are satisfied: then M satisfies every asserted literal

Model Construction and Saturation

- We will build model M such that
  - For variables x, y that are shared: M(x) = M(y) iff x ~ y
  - M(s) = { M(x) | (set.in x s) ~ true }

- Base Claim: Saturation with respect to ~ and axioms for Base ensures this
  - x ~ y, (set.in y (set.union s t)) is an atom then

    (set.in x (set.union s t)) iff (or (set.in x s) (set.in x t))

Frugal Axiom Saturation

Do we have to saturate all axioms to ensure consistent interpretations?

- Limit extensionality axioms to sets that have to be disequal for interpretation to be correct.

- Limit axiom instantiation for operators by evaluation

$$\frac{x \in U \equiv \top \qquad U \sim S \quad S \cup T \in parents(S)}{x \in S \Rightarrow x \in S \cup T}$$

Hidden Axiom Saturation

- Option 1:
  - assert axioms to the CDCL(T) core directly.
  - Prefer unit propagation eagerly
  - Defer axioms with new unassigned literals lazily

- Option 2:
  - Propagate axioms inside of the Finite Set theory solver
  - Resolve conflicts within the theory solver before telling CDCL(T) core what the conflicts are

Consistent Interpretations for Ranges

$M(s) := \{ x \mid x \in s \equiv \top \}$ does not work for ranges

$$s = [l..l+9]$$

Then $\mathrm{M}(s)$ must have 10 elements.

Can you construct a consistent interpretation after saturation with Base + Range?

Boolean Algebras

Set inclusion forms a Boolean Algebra

$S_i \subseteq T_i, S_i \subsetneq T_i$, $\quad S_i \subset T_i, S_i \not\subset T_i$ also characterized by $T \cap S, T \cup S$

Suppose a formula only uses strict and non-strict set inclusion and negations: What is a good way to check consistency of a conjunction of set inclusions?

BAPA – Boolean Algebra Presburger Arithmetic

- Recall, we admit “set.size” or |S|.
- Suppose we have set variables $s_1, \ldots, s_n$.
- Form the full Venn-diagram of the variables (2^n disjoint regions).
- Rewrite every expression using the Venn-diagram regions.
- |S| is now a sum of disjoint regions
- Every region is either unconstrained or comes from singleton or empty sets.
- Figuring out number of elements in regions is reduced to Arithmetic.

BAPA

- Regions that matter.
  - Do we really have to consider all regions over $s_1, \dots, s_n$?
  - Disjoint regions on demand:
  - $|S \cup T|$
  - $S \cup T = (S \setminus T) \cup (S \cap T) \cup (T \setminus S)$
  - $|S \cup T| = |S \setminus T| + |S \cap T| + |T \setminus S|$