presentation-only

Conflict-Driven XNF SAT Solving

Bringing **XOR** to **CDCL**

Julian Danner

Pragmatics of SAT
Glasgow, UK

August 12, 2025

UNIVERSITY OF PASSAU

XOR-OR-AND Normal Form

XNF

```
p xnf 3 4
-1+3 2 0
1+2+3 -3 0
3 1+2 0
```

$\longleftrightarrow$

Lineral

$$\bigwedge \begin{array}{l} \boxed{(\neg X_1 \oplus X_3)} \vee X_2 \\ (X_1 \oplus X_2 \oplus X_3) \vee \neg X_3 \\ \boxed{X_3 \vee (X_1 \oplus X_2)} \end{array}$$

XNF clause

Remark informal: **CNF** $\subseteq$ **CNF-XOR** $\subseteq$ **XNF**

Proposition Every formula is equisatisfiable to a formula in 2-XNF.

$\longrightarrow$ allows implication graph based solving

XNFs allow compact encodings of **XOR**-rich problems

as they naturally occur in, e.g., cryptography or approximate model counting

see **SAT Solving Using XOR-OR-AND Normal Forms**, *Math.Comput.Sci.* 18, 20 (**2024**).

Equivalent XNF Clauses

Logic		Algebra
satisfying assignments		zeros
$X_1 \oplus X_2$		$x_1 + x_2 + 1$
$(X_1 \oplus X_2) \vee (\neg X_3 \oplus X_4)$	$\longleftrightarrow$	$(x_1 + x_2 + 1) \cdot (x_3 + x_4)$
$\{ X_1 \oplus X_2,\ \neg X_3 \oplus X_4 \}$		$\{ x_1 + x_2 + 1,\ x_3 + x_4 \}$

Definition *literals*: $\mathbb{X}_n = \{ x_1, \dots, x_n \} \cup \{ x_1 + 1, \dots, x_n + 1 \}$
linerals: $\mathbb{L}_n = \langle x_1, \dots, x_n, 1 \rangle_{\mathbb{F}_2}$

Definition XNF clauses $C \subseteq \mathbb{L}_n$; **associated subspace** $V_C = \langle \underbrace{1 + C}_{\{\ell + 1 \mid \ell \in C\}} \rangle_{\mathbb{F}_2}$

Proposition $1 \in V_C \iff C \equiv \{0\}$ is a tautology

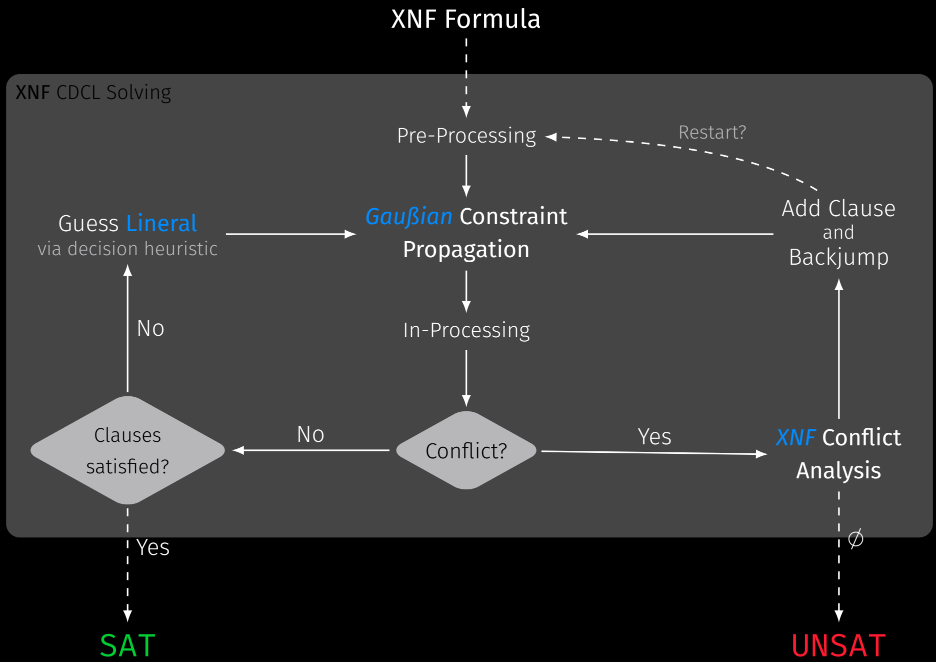
Proposition $C_1 \equiv C_2 \iff V_{C_1} = V_{C_2}$ or $1 \in V_{C_1} \cap V_{C_2}$

Corollary For $i \neq j$: $\{ \ell_1, \dots, \ell_k \} \equiv \{ \ell_1, \dots, \ell_i + \ell_j + 1, \dots, \ell_k \}$

XNF clauses are not uniquely determined by their satisfying assignments!

CDXCL

Conflict-Driven XNF Solving



reduction *modulo* $\langle L \rangle_{\mathbb{F}_2}$ / projection onto *non-pivot* inds

Definition **XNF** clause $C \subseteq \mathbb{L}_n$; set of linerals $L \subseteq \mathbb{L}_n$

- **L-assignment**: $\alpha_L : \mathbb{L}_n \to \mathbb{L}_n, \quad \ell \mapsto \mathrm{NF}(\ell, L),$
- C *under* L: $C|_L := \alpha_L(C),$
- **conflict clause under** L: $C|_L \equiv \{\,1\,\},$
- **reason clause for** u **under** L: $C|_L \equiv \{\,u\,\}.$

Gaußian Constraint Propagation (GCP)

- **GCP** of a set of linerals $L \subseteq \mathbb{L}_n$ to an **XNF** formula F

  `while there is` $C \in F$ `with` $C|_L \equiv \{\ell\}$, `add` ℓ `to` L

- admits a **lineral trail**

$$\left[\, u_1^{C_1}, \dots, u_k^{C_k} \,\right]$$

where C_i is a reason clause for u_i under $L \cup \{u_1, \dots, u_{i-1}\}$.

Proposition C is a conflict clause under $L \iff V_C \subseteq \langle L\rangle_{\mathbb{F}_2}$

Proposition C is a reason clause for u under L

$$\iff$$

there is $\underbrace{u_C \in \mathbb{L}_n}_{\text{reason lineral}}$ with $u_C|_L = u$ and $\underbrace{V_C = (V_C \cap \langle L\rangle_{\mathbb{F}_2}) \oplus \langle u_C + 1\rangle_{\mathbb{F}_2}}_{\text{reason clause decomposition}}$

XNF Conflict Analysis

Proposition lineral ℓ, set of linerals $L \subseteq \mathbb{L}_n$, **XNF** clauses $C, R \subseteq \mathbb{L}_n$, where

- C is a conflict clause under $L \cup \{\ell\}$ but not under L, and
- R is a reason clause for ℓ under L with a reason lineral u_R.

(a) Then C is a reason clause for $\ell + 1$ under L with a reason lineral u_C, and

(b) every **XNF** clause $C' \subseteq \mathbb{L}_n$ with

$$V_{C'} = \big(V_C \cap \langle L\rangle_{\mathbb{F}_2}\big) + \big(V_R \cap \langle L\rangle_{\mathbb{F}_2}\big) + \langle u_C + u_R + 1\rangle_{\mathbb{F}_2},$$

satisfies $R, C \models C'$, and C' is a conflict clause under L.

This allows a *direct* generalization of **CNF** conflict learning methods!

Remark

- learning based on the proof system **XRES**; consisting of **weak**, **res**, and

$$\frac{C \cup \{\ell_1, \ell_2\}}{C \cup \{\ell_1, \ell_1 + \ell_2 + 1\}} \; (\textbf{rewr})$$

- **XRES** is p-equivalent to **RES(⊕)**, which is exp stronger than **RES**

BUT **no** lazy data structures for GCP & learning is *computationally expensive*!

$\Longrightarrow$ CDXCL useless in practice?

$\text{CDXCL}^{\text{Lite}}$

- Literal Decisions
- Propagation only literals with $\textbf{GCP}^{\textbf{Lite}}$ as follows

`while there is` $C \in F$ `with` $C|_{L \cap \mathbb{X}_n} \equiv \{\ell\}$`, add` ℓ `to` L

`while there is` $\ell \in L$ `with` $\ell|_{L \cap \mathbb{X}_n} = \ell' \in \mathbb{X}_n$`, add` ℓ' `to` L

- **Conflict Learning** using *literal trail* as in **CDXCL**!

AND *there are* lazy data structures & learning requires *almost* no overhead!

$\Longrightarrow$ $\text{CDXCL}^{\text{Lite}}$ *might* be useful in practice!

Xorricane

Prototype Implementation of CDXCL$^{\text{Lite}}$ in C++

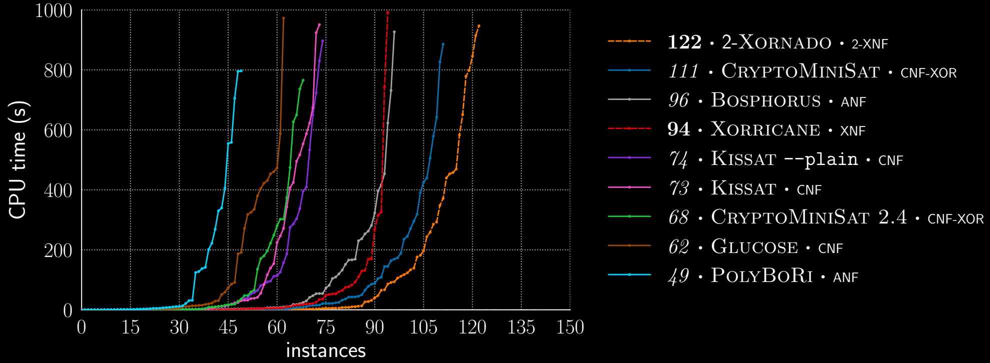
- Gaußian Constraint Propagation Lite
  - lazy data structures (*watching structures*) ✓
  - reduction with equivalences ✓
  - lazy Gauß-Jordan Elimination for unit clauses ✓
- **XNF** Conflict Analysis
  - lazy data structures (*filtrations*) ✓
  - clause minimization ×
- heuristics $\longrightarrow$ fine-tuning required!
  - EVSIDS for decisions ✓
  - LBD-based dynamic (blocking/forcing) restarts ~
  - tier-based deletion heuristic ~
- graph-based preprocessing ✓
- Gauß-Jordan in-processing ✓

Competing Solvers

ANF	PolyBoRi (2023)	*Boolean Gröbner bases*
	Bosphorus 3.0 (2020)	***CDCL(XOR)***, *ElimLin*, *Linearization*
XNF	2-Xornado (2023)	*graph-based **DPLL** 2-**XNF** Solving*
	Xorricane (2024)	***CDXCL***^{Lite} *with some lazy GJE, little pre-/in-proc*
CNF-XOR	CryptoMiniSat 2.4 (2010)	***CDCL(XOR)*** *w/o lazy GJE, Winner SAT'10*
	CryptoMiniSat 5.11 (2024)	***CDCL(XOR)*** *with lazy GJE, lots of in-proc*
CNF	Glucose 2.0 (2011)	*little pre-proc, Winner SAT'12*
	Kissat `sc2024` (2024)	*advanced **CDCL**, Winner SAT'24*
	Kissat `--plain` (2024)	***CDCL*** *w/o advanced techniques*

Random Benchmarks

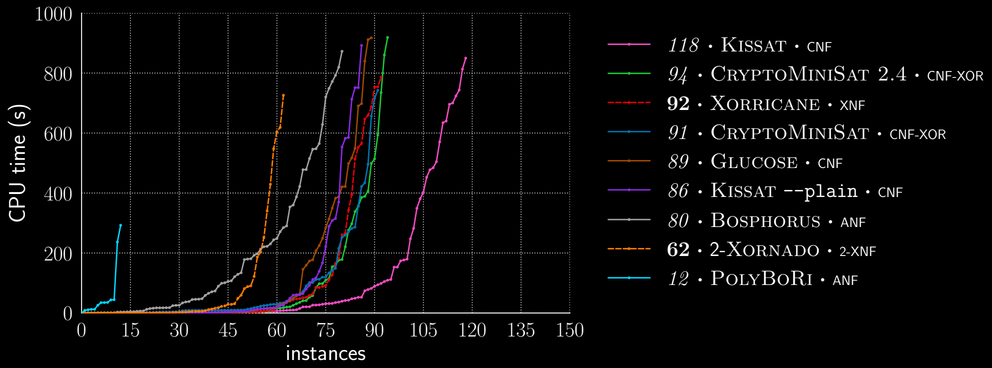
- random *underdetermined* systems of multivariate quadratic polynomials
- random *overdetermined* systems of multivariate quadratic polynomials
- random sparse 2-**XNF** with dense **XOR** constraints



150 satisfiable random instances

Cryptographic Benchmarks

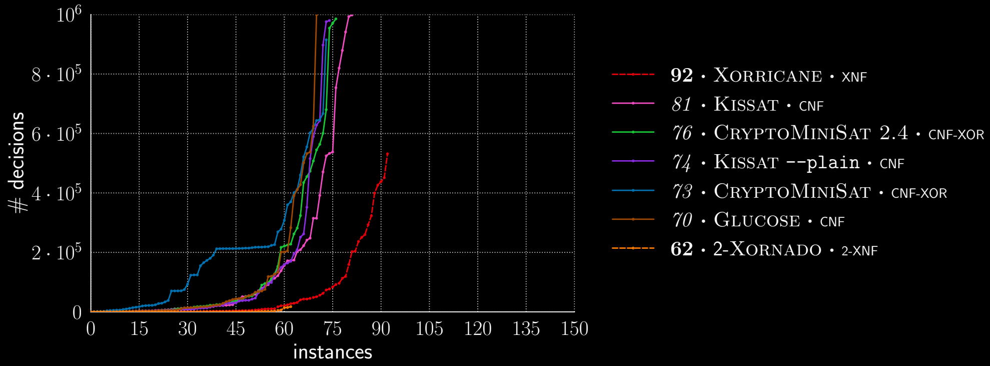
- sponge cipher: round-reduced key-recovery attack on `Ascon`
- block cipher: key-recovery attack on toy cipher `CTC2`
- stream cipher: state-recovery on toy cipher `Bivium`



150 satisfiable cryptographic instances

Cryptographic Benchmarks

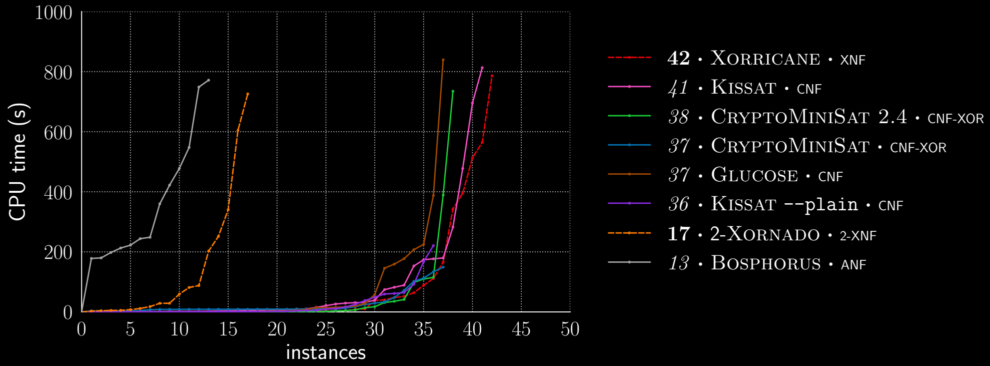
- sponge cipher: round-reduced key-recovery attack on `Ascon`
- block cipher: key-recovery attack on toy cipher `CTC2`
- stream cipher: state-recovery on toy cipher `Bivium`



capped at 10^6 conflicts

Cryptographic Benchmark

Bivium



Try it yourself!

Implementation and benchmark instances available at

`github.com/j-danner/xnf_sat_solving`

Paper **SAT Solving Using XOR-OR-AND Normal Forms**
Math.Comput.Sci. *18, 20 (**2024**).*

Paper **Conflict-Driven SAT Solving Using XOR-OR-AND Normal Forms**
to be submitted very soon