A Characteristic Set Method for Solving Boolean Equations

Chun-Ming Yuan

joint work with F.J. Chai & X.S. Gao

Institute of Systems Science
Chinese Academy of Sciences

MAP, ICTP, TRIESTE 2008.8.27

Outline

- Background
- A Characteristic Set Method for Boolean Equations
- Implementation and Variation
- Experimental Result with a Class of Stream Ciphers
- Conclusion

Characteristic Set Method

$$\begin{array}{lcl} P_1(x_1, \ldots, x_n) & & A_1(u_1, \ldots, u_q, y_1) \\ P_2(x_1, \ldots, x_n) & & A_2(u_1, \ldots, u_q, y_1, y_2) \\ & \Rightarrow & \quad \ldots \\ P_m(x_1, \ldots, x_n) & & A_p(u_1, \ldots, u_q, y_1, \ldots, y_p) \end{array}$$

Polynomial system $\Rightarrow$ Triangular set

Characteristic Set Method: An Example

Example (Zhu Shijie)

$$
\begin{aligned}
P_1 &= xyz - xy^2 - z - x - y, \\
P_2 &= xz - x^2 - z - y + x, \\
P_3 &= z^2 - x^2 - y^2.
\end{aligned}
$$

Characteristic Set Method: An Example

Example (Zhu Shijie)

$$P_1 = xyz - xy^2 - z - x - y,$$
$$P_2 = xz - x^2 - z - y + x,$$
$$P_3 = z^2 - x^2 - y^2.$$

We have:

$$\mathrm{Zero}(\{P_1, P_2, P_3\}) = \mathrm{Zero}(\mathcal{C}_1) \cup \mathrm{Zero}(\mathcal{C}_2) \cup \mathrm{Zero}(\mathcal{C}_3).$$

$\mathcal{C}_1 = x - 3, y - 4, z - 5;$ One solution

$\mathcal{C}_2 = x - 1, y, z + 1;$ One solution

$\mathcal{C}_3 = x, y + z;$ Dimension one

Existing Work on CS Method

- **Algebraic Equation over $\mathcal{C}$:** the most basic case, lots of work since the pioneering paper of Wu in 1978.
- **Differential Equations:** Ritt 1930s, Kolchin 1930-70s, Wu 1970s, etc. Also extensively studied.
- **Difference Equations:** Theory: Ritt 1930s, Cohn 1950s. Algorithms: Gao et al, since 2004.
- **Finite Fields, in particular, Boolean equations:** ?.

Solving Boolean Equation Systems

Motivation.

- Design and formal verification of hardware.
- Cryptanalysis.
- Deciding whether a Boolean polynomial system has solutions is NP-complete.

Solving Boolean Equation Systems

Motivation.

- Design and formal verification of hardware.
- Cryptanalysis.
- Deciding whether a Boolean polynomial system has solutions is NP-complete.

Methods to solve Boolean equation systems.

- Logic approaches: Quine normal form, Davis-Putnam, et all.
- Methods based on graphs: BDD/ZDD.
- Probability and approximate methods.
- Methods based on elimination: Boole's method, Gröbner basis, and the Characteristic set method.

Solving Boolean Equations with Characteristic Set Method

Boolean Ring: Notations

$\mathbf{F}_2 = \mathbf{Z}/(2) = \{0, 1\}$.
$\mathbb{X} = \{x_1, \ldots, x_n\}$ a set of indeterminants
$\mathbb{H} = \{x_1^2 + x_1, \ldots, x_n^2 + x_n\}$
A Boolean Ring:

$$\mathbb{R}_2 = \mathbb{R}_{2,n} = \mathbf{F}_2[\mathbb{X}]/(\mathbb{H})$$

Boolean Ring: Notations

$\mathbf{F}_2 = \mathbf{Z}/(2) = \{0, 1\}$.
$\mathbb{X} = \{x_1, \ldots, x_n\}$ a set of indeterminants
$\mathbb{H} = \{x_1^2 + x_1, \ldots, x_n^2 + x_n\}$

A Boolean Ring:

$$\mathbb{R}_2 = \mathbb{R}_{2,n} = \mathbf{F}_2[\mathbb{X}]/(\mathbb{H})$$

Connection between Boolean Ring and Boolean Algebra:

Boolean Algebra $\Rightarrow$ Boolean Ring:

$f \wedge g \Rightarrow f \cdot g$
$f \vee g \Rightarrow f \cdot g + f + g$

Boolean Ring $\Rightarrow$ Boolean Algebra:

$f \cdot g \Rightarrow f \wedge g$
$f + g \Rightarrow \bar{f} \wedge g \vee f \wedge \bar{g}$

Zeros of Boolean Polynomials

Variety: $\overline{\mathrm{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\mathrm{Zero}}(\mathbb{P}/D) = \overline{\mathrm{Zero}}(\mathbb{P}) \setminus \overline{\mathrm{Zero}}(D)$.

Zeros of Boolean Polynomials

Variety: $\overline{\mathrm{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\mathrm{Zero}}(\mathbb{P}/D) = \overline{\mathrm{Zero}}(\mathbb{P}) \setminus \overline{\mathrm{Zero}}(D)$.

Basic Properties.

Let $U, V, D \in \mathbb{R}_2$ and $\mathbb{P} \subset \mathbb{R}_2$. We have

$$U \neq 1 \Rightarrow \overline{\mathrm{Zero}}(U) \neq \emptyset.$$

Zeros of Boolean Polynomials

Variety: $\overline{\mathrm{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\mathrm{Zero}}(\mathbb{P}/D) = \overline{\mathrm{Zero}}(\mathbb{P}) \setminus \overline{\mathrm{Zero}}(D)$.

Basic Properties.

Let $U, V, D \in \mathbb{R}_2$ and $\mathbb{P} \subset \mathbb{R}_2$. We have

$$U \neq 1 \Rightarrow \overline{\mathrm{Zero}}(U) \neq \emptyset.$$

$$|\overline{\mathrm{Zero}}(\mathbb{P})| = 1 \Leftrightarrow (\mathbb{P}) = (x_1 - a_1, \ldots, x_n - a_n).$$

Zeros of Boolean Polynomials

Variety: $\overline{\mathrm{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\mathrm{Zero}}(\mathbb{P}/D) = \overline{\mathrm{Zero}}(\mathbb{P}) \setminus \overline{\mathrm{Zero}}(D)$.

Basic Properties.

Let $U, V, D \in \mathbb{R}_2$ and $\mathbb{P} \subset \mathbb{R}_2$. We have

$$U \neq 1 \Rightarrow \overline{\mathrm{Zero}}(U) \neq \emptyset.$$

$$|\overline{\mathrm{Zero}}(\mathbb{P})| = 1 \Leftrightarrow (\mathbb{P}) = (x_1 - a_1, \ldots, x_n - a_n).$$

$$\overline{\mathrm{Zero}}(UV + 1) = \overline{\mathrm{Zero}}(\{U + 1, V + 1\}).$$

Zeros of Boolean Polynomials

Variety: $\overline{\mathrm{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\mathrm{Zero}}(\mathbb{P}/D) = \overline{\mathrm{Zero}}(\mathbb{P}) \setminus \overline{\mathrm{Zero}}(D)$.

Basic Properties.

Let $U, V, D \in \mathbb{R}_2$ and $\mathbb{P} \subset \mathbb{R}_2$. We have

$$U \neq 1 \Rightarrow \overline{\mathrm{Zero}}(U) \neq \emptyset.$$
$$|\overline{\mathrm{Zero}}(\mathbb{P})| = 1 \Leftrightarrow (\mathbb{P}) = (x_1 - a_1, \ldots, x_n - a_n).$$
$$\overline{\mathrm{Zero}}(UV + 1) = \overline{\mathrm{Zero}}(\{U + 1, V + 1\}).$$
$$\overline{\mathrm{Zero}}(UV + U + V) = \overline{\mathrm{Zero}}(\{U, V\}).$$

Zeros of Boolean Polynomials

Variety: $\overline{\text{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\text{Zero}}(\mathbb{P}/D) = \overline{\text{Zero}}(\mathbb{P}) \setminus \overline{\text{Zero}}(D)$.

Basic Properties.

Let $U, V, D \in \mathbb{R}_2$ and $\mathbb{P} \subset \mathbb{R}_2$. We have

$$U \neq 1 \Rightarrow \overline{\text{Zero}}(U) \neq \emptyset.$$

$$|\overline{\text{Zero}}(\mathbb{P})| = 1 \Leftrightarrow (\mathbb{P}) = (x_1 - a_1, \ldots, x_n - a_n).$$

$$\overline{\text{Zero}}(UV + 1) = \overline{\text{Zero}}(\{U + 1, V + 1\}).$$

$$\overline{\text{Zero}}(UV + U + V) = \overline{\text{Zero}}(\{U, V\}).$$

$$\overline{\text{Zero}}(\emptyset/D) = \overline{\text{Zero}}(D + 1).$$

Zeros of Boolean Polynomials

Variety: $\overline{\mathrm{Zero}}(\mathbb{P}) = \{\alpha \in \mathbf{F}_2^n, s.t. \quad \forall P \in \mathbb{P}, P(\alpha) = 0\}$.

Quasi Variety: $\overline{\mathrm{Zero}}(\mathbb{P}/D) = \overline{\mathrm{Zero}}(\mathbb{P}) \setminus \overline{\mathrm{Zero}}(D)$.

Basic Properties.

Let $U, V, D \in \mathbb{R}_2$ and $\mathbb{P} \subset \mathbb{R}_2$. We have

$$U \neq 1 \Rightarrow \overline{\mathrm{Zero}}(U) \neq \emptyset.$$
$$|\overline{\mathrm{Zero}}(\mathbb{P})| = 1 \Leftrightarrow (\mathbb{P}) = (x_1 - a_1, \ldots, x_n - a_n).$$
$$\overline{\mathrm{Zero}}(UV + 1) = \overline{\mathrm{Zero}}(\{U + 1, V + 1\}).$$
$$\overline{\mathrm{Zero}}(UV + U + V) = \overline{\mathrm{Zero}}(\{U, V\}).$$
$$\overline{\mathrm{Zero}}(\emptyset/D) = \overline{\mathrm{Zero}}(D + 1).$$
$$\overline{\mathrm{Zero}}(\mathbb{P}) = \overline{\mathrm{Zero}}(\mathbb{P} \cup \{U\}) \cup \overline{\mathrm{Zero}}(\mathbb{P} \cup \{U + 1\}).$$

Zeros of Triangular Sets

Monic Triangular Set:

$$\mathcal{A} = \begin{cases} A_1 = x_{c_1} + U_1(\mathbb{U}) \\ \cdots \\ A_p = x_{c_p} + U_p(\mathbb{U}) \end{cases} \tag{1}$$

Parameter set: $\mathbb{U} = \{x_i | i \neq c_j\}$.
Dimension of $\mathcal{A}$: $\mathbf{dim}(\mathcal{A}) = |\mathbb{U}| = n - |\mathcal{A}|$.

Zeros of Triangular Sets

Monic Triangular Set:

$$\mathcal{A} = \begin{cases} A_1 = x_{c_1} + U_1(\mathbb{U}) \\ \cdots \\ A_p = x_{c_p} + U_p(\mathbb{U}) \end{cases} \quad (1)$$

Parameter set: $\mathbb{U} = \{x_i | i \neq c_j\}$.
Dimension of $\mathcal{A}$: $\mathbf{dim}(\mathcal{A}) = |\mathbb{U}| = n - |\mathcal{A}|$.

Lemma

Let $\mathcal{A}$ be a monic triangular set. Then $|\overline{\mathrm{Zero}}(\mathcal{A})| = 2^{\mathbf{dim}(\mathcal{A})}$.

Zeros of Triangular Sets

Monic Triangular Set:

$$\mathcal{A} = \begin{cases} A_1 = x_{c_1} + U_1(\mathbb{U}) \\ \cdots \\ A_p = x_{c_p} + U_p(\mathbb{U}) \end{cases} \quad (1)$$

Parameter set: $\mathbb{U} = \{x_i | i \neq c_j\}$.
Dimension of $\mathcal{A}$: $\mathbf{dim}(\mathcal{A}) = |\mathbb{U}| = n - |\mathcal{A}|$.

Lemma

Let $\mathcal{A}$ be a monic triangular set. Then $|\overline{\mathrm{Zero}}(\mathcal{A})| = 2^{\mathbf{dim}(\mathcal{A})}$.

A chain $\mathcal{A}$ is called **conflict** if $\mathbf{I}_{\mathcal{A}} = 0$.

Lemma

Let $\mathcal{A}$ be a non-conflict chain. Then $\overline{\mathrm{Zero}}(\mathcal{A}/\mathbf{I}_{\mathcal{A}}) \neq \emptyset$.

Characteristic Set

Ordering: $\mathcal{A} = A_1, \ldots, A_r, \quad \mathcal{B} = B_1, \ldots, B_s$

$\mathcal{A} \prec \mathcal{B}$ if

either $\exists k$ st $A_1 \sim B_1, \ldots, A_{k-1} \sim B_{k-1}$, and $A_k \prec B_k$;

or $r > s$ and $A_1 \sim B_1, \ldots, A_s \sim B_s$.

Characteristic Set

Ordering: $\mathcal{A} = A_1, \ldots, A_r, \quad \mathcal{B} = B_1, \ldots, B_s$

$\mathcal{A} \prec \mathcal{B}$ if

either $\exists k$ st $A_1 \sim B_1, \ldots, A_{k-1} \sim B_{k-1}$, and $A_k \prec B_k$;

or $r > s$ and $A_1 \sim B_1, \ldots, A_s \sim B_s$.

Lemma

A sequence of triangular sets steadily lower in ordering is finite.
Let $\mathcal{A}_1 \succ \mathcal{A}_2 \succ \cdots \succ \mathcal{A}_m$. Then $m \leq 2^n$.

Characteristic Set

Ordering: $\mathcal{A} = A_1, \ldots, A_r, \quad \mathcal{B} = B_1, \ldots, B_s$

$\mathcal{A} \prec \mathcal{B}$ if

either $\exists k$ st $A_1 \sim B_1, \ldots, A_{k-1} \sim B_{k-1}$, and $A_k \prec B_k$;

or $r > s$ and $A_1 \sim B_1, \ldots, A_s \sim B_s$.

Lemma

A sequence of triangular sets steadily lower in ordering is finite.
Let $\mathcal{A}_1 \succ \mathcal{A}_2 \succ \cdots \succ \mathcal{A}_m$. Then $m \leq 2^n$.

Definition (Characteristic Set)

$\mathbb{P}$ be a set of Boolean polynomials. The smallest triangular set in $\mathbb{P}$ is called the CS of $\mathbb{P}$.

Pseudo-remainder

Pseudo-remainder of Boolean Polynomials

$P = Ix_c + U$ with $\mathrm{cls}(P) = c$.

$Q = I_1 x_c + U_1$.

Pseudo-remainder: $R = \mathrm{prem}(Q, P) = IU_1 + I_1 U$.

Remainder Formula: $\mathrm{init}(P)Q = BP + R$.

Reduced: R is reduced wrt P: x_c does not occur in R.

Pseudo-remainder

Pseudo-remainder of Boolean Polynomials

$P = Ix_c + U$ with $\mathrm{cls}(P) = c$.

$Q = I_1 x_c + U_1$.

Pseudo-remainder: $R = \mathrm{prem}(Q, P) = IU_1 + I_1 U$.

Remainder Formula: $\mathrm{init}(P)Q = BP + R$.

Reduced: R is reduced wrt P: x_c does not occur in R.

Pseudo-remainder of Boolean Polynomials wrt TS

$R = \mathrm{prem}(Q, \mathcal{A}) = \mathrm{prem}(\mathrm{prem}(Q, A_r), A_1, \ldots, A_{r-1})$

Remainder Formula: $\mathbf{I}_{\mathcal{A}} G = \sum_i Q_i A_i + R$

$\mathbf{I}_{\mathcal{A}}$: product of the initials of the polynomials in $\mathcal{A}$.

Well-Ordering Principle

Let $\mathbb{P}_0$ be a finite Boolean polynomial set.

$$\begin{array}{rcl} \mathbb{P} & = & \mathbb{P}_0 \quad \mathbb{P}_1 \quad \cdots \quad \mathbb{P}_i \quad \cdots \quad \mathbb{P}_m \\ & & \mathcal{C}_0 \quad \mathcal{C}_1 \quad \cdots \quad \mathcal{C}_i \quad \cdots \quad \mathcal{C}_m = \mathcal{C} \\ & & \mathbb{R}_0 \quad \mathbb{R}_1 \quad \cdots \quad \mathbb{R}_i \quad \cdots \quad \mathbb{R}_m = \emptyset \end{array} \tag{2}$$

$\mathcal{C}_i =$ a characteristic set of $\mathbb{P}_i$
$\mathbb{R}_i = \text{prem}(\mathbb{P}_i, \mathcal{C}_i)$
$\mathbb{P}_{i+1} = \mathbb{P}_i \cup \mathbb{R}_i$

Well-Ordering Principle

Let $\mathbb{P}_0$ be a finite Boolean polynomial set.

$$
\begin{array}{lllllll}
\mathbb{P} = & \mathbb{P}_0 & \mathbb{P}_1 & \cdots & \mathbb{P}_i & \cdots & \mathbb{P}_m \\
 & \mathcal{C}_0 & \mathcal{C}_1 & \cdots & \mathcal{C}_i & \cdots & \mathcal{C}_m = \mathcal{C} \\
 & \mathbb{R}_0 & \mathbb{R}_1 & \cdots & \mathbb{R}_i & \cdots & \mathbb{R}_m = \emptyset
\end{array} \quad (2)
$$

$\mathcal{C}_i$ = a characteristic set of $\mathbb{P}_i$

$\mathbb{R}_i = \text{prem}(\mathbb{P}_i, \mathcal{C}_i)$

$\mathbb{P}_{i+1} = \mathbb{P}_i \cup \mathbb{R}_i$

Fact. $m \leq 2^n$.

Well-Ordering Principle

Let $\mathbb{P}_0$ be a finite Boolean polynomial set.

$$\begin{array}{rcl} \mathbb{P} & = & \mathbb{P}_0 \quad \mathbb{P}_1 \quad \cdots \quad \mathbb{P}_i \quad \cdots \quad \mathbb{P}_m \\ & & \mathcal{C}_0 \quad \mathcal{C}_1 \quad \cdots \quad \mathcal{C}_i \quad \cdots \quad \mathcal{C}_m = \mathcal{C} \\ & & \mathbb{R}_0 \quad \mathbb{R}_1 \quad \cdots \quad \mathbb{R}_i \quad \cdots \quad \mathbb{R}_m = \emptyset \end{array} \quad (2)$$

$\mathcal{C}_i =$ a characteristic set of $\mathbb{P}_i$

$\mathbb{R}_i = \mathrm{prem}(\mathbb{P}_i, \mathcal{C}_i)$

$\mathbb{P}_{i+1} = \mathbb{P}_i \cup \mathbb{R}_i$

Fact. $m \leq 2^n$.

Wu Characteristic Set of $\mathbb{P}$: $\mathcal{C}$

(1) $\forall P \in \mathbb{P}$, $\mathrm{prem}(P, \mathcal{C}) = 0$.

(2) $\mathcal{C} \subset (\mathbb{P})$.

Fact: $\mathcal{C}_m$ is a Wu CS of $\mathbb{P}$.

Zero Decomposition Theorem

$\mathbb{P}$: a finite Boolean polynomial set.

Theorem (Well-ordering principle (1))

Let $\mathcal{C} = C_1, \ldots, C_p$ be a Wu CS of $\mathbb{P}$. Then

$$\overline{\mathrm{Zero}}(\mathbb{P}) = \overline{\mathrm{Zero}}(\mathcal{C}/\mathbf{I}_{\mathcal{C}}) \bigcup \cup_{i=1}^{p} \overline{\mathrm{Zero}}(\mathbb{P} \cup \mathcal{C} \cup \{I_i\})$$

where $I_i = init(C_i)$.

Fact. $\mathbf{I}_{\mathcal{C}} P = \sum_i B_i C_i$, for $P \in \mathbb{P}$.

Zero Decomposition Theorem

$\mathbb{P}$: a finite Boolean polynomial set.

Theorem (Well-ordering principle (1))

Let $\mathcal{C} = C_1, \ldots, C_p$ be a Wu CS of $\mathbb{P}$. Then

$$\overline{\mathrm{Zero}}(\mathbb{P}) = \overline{\mathrm{Zero}}(\mathcal{C}/\mathbf{I}_{\mathcal{C}}) \bigcup \cup_{i=1}^{p} \overline{\mathrm{Zero}}(\mathbb{P} \cup \mathcal{C} \cup \{I_i\})$$

where $I_i = init(C_i)$.

Fact. $\mathbf{I}_{\mathcal{C}} P = \sum_i B_i C_i$, for $P \in \mathbb{P}$.

Theorem (Zero Decomposition Theorem)

We can construct chains $\mathcal{A}_j, j = 1, \ldots, s$ such that

$$\overline{\mathrm{Zero}}(\mathbb{P}) = \cup_{j=1}^{s} \overline{\mathrm{Zero}}(\mathcal{A}_j/\mathbf{I}_{\mathcal{A}_j}).$$

Monic Zero Decomposition Theorem

$\mathbb{P}$: a finite Boolean polynomial set.

Theorem (Well-ordering principle (2))

Let $\mathcal{C} = C_1, \ldots, C_p$ be a Wu CS of $\mathbb{P}$ with $I_i = init(C_i)$. Then

$$\overline{\mathrm{Zero}}(\mathbb{P}) = \overline{\mathrm{Zero}}(\mathcal{C} \cup \{I_1 + 1, \ldots, I_p + 1\}) \cup_{i=1}^{p} \overline{\mathrm{Zero}}(\mathbb{P} \cup \mathcal{C} \cup \{I_i\})$$

Fact. $\overline{\mathrm{Zero}}(/\mathbf{I}_{\mathcal{C}}) = \overline{\mathrm{Zero}}(\mathbf{I}_{\mathcal{C}} + 1) = \overline{\mathrm{Zero}}(I_1 + 1, \ldots, I_p + 1)$

Monic Zero Decomposition Theorem

$\mathbb{P}$: a finite Boolean polynomial set.

Theorem (Well-ordering principle (2))

Let $\mathcal{C} = C_1, \ldots, C_p$ be a Wu CS of $\mathbb{P}$ with $I_i = init(C_i)$. Then

$$\overline{\text{Zero}}(\mathbb{P}) = \overline{\text{Zero}}(\mathcal{C} \cup \{I_1 + 1, \ldots, I_p + 1\}) \cup_{i=1}^{p} \overline{\text{Zero}}(\mathbb{P} \cup \mathcal{C} \cup \{I_i\})$$

Fact. $\overline{\text{Zero}}(/\mathbf{I}_\mathcal{C}) = \overline{\text{Zero}}(\mathbf{I}_\mathcal{C} + 1) = \overline{\text{Zero}}(I_1 + 1, \ldots, I_p + 1)$

Theorem (Monic Zero Decomposition Theorem)

We can construct monic chains $\mathcal{A}_j, j = 1, \ldots, t$ such that

$$\overline{\text{Zero}}(\mathbb{P}) = \cup_{j=1}^{t} \overline{\text{Zero}}(\mathcal{A}_j).$$

Example

Let $P = x_1x_2x_3 + 1$.

By ZDT, $\overline{\text{Zero}}(P) = \overline{\text{Zero}}(P/x_1x_2) \neq \emptyset$.

By MZDT,

$$\begin{aligned} \overline{\text{Zero}}(P) &= \overline{\text{Zero}}(x_1 + 1, x_2 + 1, P) \cup \overline{\text{Zero}}(x_1, P) \cup \overline{\text{Zero}}(x_2, P) \\ &= \overline{\text{Zero}}(x_1 + 1, x_2 + 1, x_3 + 1). \end{aligned}$$

Well-ordering principle

$\mathbb{P}$: a finite Boolean polynomial set.

Theorem (Well-ordering principle)

Let $\mathcal{C} = C_1, \ldots, C_p$ *be a Wu CS of* $\mathbb{P}$. *Then*

$$\begin{aligned}\overline{\mathrm{Zero}}(\mathbb{P}) &= \overline{\mathrm{Zero}}(\mathcal{C} \cup \{I_1 + 1, \ldots, I_p + 1\}) \cup \\ &\quad \overline{\mathrm{Zero}}(\mathbb{Q} \cup \{I_1\}) \cup \overline{\mathrm{Zero}}(\mathbb{Q} \cup \{I_1 + 1, I_2\}) \cup \cdots \\ &\quad \overline{\mathrm{Zero}}(\mathbb{Q} \cup \{I_1 + 1, \ldots, I_{p-1} + 1, I_p\})\end{aligned}$$

where $I_i = init(C_i)$, $\mathbb{Q} = \mathbb{P} \cup \mathcal{C}$.

Fact. $\overline{\mathrm{Zero}}(\{P\}) \cup \overline{\mathrm{Zero}}(\{Q\}) = \overline{\mathrm{Zero}}(P) \cup \overline{\mathrm{Zero}}(Q/P)$
Note that every pair of components is disjoint.

Disjoint Monic Zero Decomposition Theorem

Theorem (DMZDT)

We can find monic chains $\mathcal{A}_j, j = 1, \ldots, s$ *such that*

$$\overline{\mathrm{Zero}}(\mathbb{P}) = \cup_{i=1}^{s} \overline{\mathrm{Zero}}(\mathcal{A}_i)$$

and $\overline{\mathrm{Zero}}(\mathcal{A}_i) \cap \overline{\mathrm{Zero}}(\mathcal{A}_j) = \emptyset$ *for* $i \neq j$*.*
As a consequence,

$$|\overline{\mathrm{Zero}}(\mathbb{P})| = \sum_{i=1}^{s} 2^{\mathbf{dim}(\mathcal{A}_i)}.$$

Example

$\mathbb{P} = \{x_1 x_2 + x_2 + x_1 + 1\}$.

We have, $\overline{\mathrm{Zero}}(\mathbb{P}) = \overline{\mathrm{Zero}}(\mathcal{A}_1) \cup \overline{\mathrm{Zero}}(\mathcal{A}_2)$,

$$\mathcal{A}_1 = x_1, x_2 + 1;$$
$$\mathcal{A}_2 = x_1 + 1.$$

Then, $|\overline{\mathrm{Zero}}(\mathbb{P})| = 2^0 + 2^1 = 3$.

Complexity of Modified Well-ordering Principle

Modified Well-ordering Principle

$$
\begin{array}{llllllll}
\mathbb{P} = & \mathbb{P}_0 & \mathbb{P}_1 & \cdots & \mathbb{P}_i & \cdots & \mathbb{P}_m \\
& \mathcal{C}_0 & \mathcal{C}_1 & \cdots & \mathcal{C}_i & \cdots & \mathcal{C}_m = \mathcal{C} \\
& \mathbb{R}_0 & \mathbb{R}_1 & \cdots & \mathbb{R}_i & \cdots & \mathbb{R}_m = \emptyset
\end{array} \tag{3}
$$

$\mathcal{C}_i =$ a characteristic set of $\mathbb{P}_i$

$\mathbb{R}_i = \text{prem}(\mathbb{P}_i, \mathcal{C}_i)$

$\mathbb{P}_{i+1} = \mathcal{C}_i \cup \mathbb{R}_i$; $(\mathbb{P}_{i+1} = \mathbb{P}_i \cup \mathbb{R}_i)$

Complexity of Modified Well-ordering Principle

Modified Well-ordering Principle

$$\begin{array}{ccccccccc} \mathbb{P} = & \mathbb{P}_0 & \mathbb{P}_1 & \cdots & \mathbb{P}_i & \cdots & \mathbb{P}_m & \\ & \mathcal{C}_0 & \mathcal{C}_1 & \cdots & \mathcal{C}_i & \cdots & \mathcal{C}_m = \mathcal{C} \\ & \mathbb{R}_0 & \mathbb{R}_1 & \cdots & \mathbb{R}_i & \cdots & \mathbb{R}_m = \emptyset \end{array} \quad (3)$$

$\mathcal{C}_i =$ a characteristic set of $\mathbb{P}_i$

$\mathbb{R}_i = \mathrm{prem}(\mathbb{P}_i, \mathcal{C}_i)$

$\mathbb{P}_{i+1} = \mathcal{C}_i \cup \mathbb{R}_i$; $(\mathbb{P}_{i+1} = \mathbb{P}_i \cup \mathbb{R}_i)$

Theorem

Let $l = |\mathbb{P}|$. In the modified well-ordering principle, we have

- $m \leq n$,
- *need $O(n^2 l)$ polynomial multiplications.*

Theorem (Modified well-ordering principle)

Let $I_1, \ldots, I_s$ be the initials of the polynomials in $\mathcal{C}_m, \ldots, \mathcal{C}_0$, $H_j = prem(I_i, \mathcal{C}), j = 1, \ldots, s$, and J_m the product for all the H_j. Then,

$$\begin{aligned} & \overline{\mathrm{Zero}}(\mathbb{P}) \\ = \; & \overline{\mathrm{Zero}}(\mathcal{C}/J_m) \bigcup \cup_{i=1}^{s} \overline{\mathrm{Zero}}(\mathbb{P} \cup \mathcal{C} \cup \{H_1 + 1, \ldots, H_{i-1} + 1, H_i\}) \\ = \; & \overline{\mathrm{Zero}}(\mathcal{C} \cup \{I_1 + 1, \ldots, I_s + 1\}) \bigcup \cup_i^s \overline{\mathrm{Zero}}(\mathbb{P} \cup \mathcal{C} \cup \{I_i\}) \end{aligned}$$

Comments of the CS Methods

- Compare to the general CS method:
  - We have a disjoint monic zero decomposition.
  - Well ordering principle needs a polynomial number of arithmetic operations.

Comments of the CS Methods

- Compare to the general CS method:
  - We have a disjoint monic zero decomposition.
  - Well ordering principle needs a polynomial number of arithmetic operations.
- The method tries to find a balance point between space growth and the branch growth:
  - To find one Wu CS needs a polynomial number of arithmetic operations.
  - If the Wu CS is conflict, split the problem into smaller ones.

Comments of the CS Methods

- Compare to the general CS method:
  - We have a disjoint monic zero decomposition.
  - Well ordering principle needs a polynomial number of arithmetic operations.
- The method tries to find a balance point between space growth and the branch growth:
  - To find one Wu CS needs a polynomial number of arithmetic operations.
  - If the Wu CS is conflict, split the problem into smaller ones.
- The method gives a clear and compact way to represent the solutions of Boolean equation systems.

Implementation and Variations of the Method

Implementation

System and Data Structure

Using C, both in Linux and Windows (VC++) systems.

Implementation

System and Data Structure

Using C, both in Linux and Windows (VC++) systems.

- Principle Balance Between Sizes and Branches.
- Boolean polynomial representation
  - Polynomial: Linked list of monomials.
  - Recursive representation: $P = Ix_c + U$.
  - SZDD.
- Parallel implementation

Solving Boolean Equations: Two Extreme Cases

Truth Table: 2^n

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Solving Boolean Equations: Two Extreme Cases

Truth Table: 2^n

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Reduce to One Equation

- $f(x_1, \ldots, x_n) = g(x_1, \ldots, x_n)$
  $\Leftrightarrow$
  $h = \bar{f} \wedge g \vee \bar{g} \wedge f = 0.$
- $f_1 = f_2 = \cdots f_m = 0$
  $\Leftrightarrow$
  $f = f_1 \vee f_2 \vee \cdots \vee f_m = 0.$
- **Quine Normal Form**:
  $f = 0$ has a unique solution
  $\Leftrightarrow$
  $f = x_1 \vee \bar{x_2} \vee \cdots \vee x_n.$

Balance Between Sizes and Branches

Comparison.

- **Truth Table.** Need to test many cases, but to test one case is fast.
- **Quine Normal Form.** Need to test one case, but generally will produce large polynomial.

Balance Between Sizes and Branches

Comparison.

- **Truth Table.** Need to test many cases, but to test one case is fast.
- **Quine Normal Form.** Need to test one case, but generally will produce large polynomial.

Principle of Balance Between Sizes and Branches. Try to produce as few branches as possible under the constraint that the memory of the computers to be sufficiently used.

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a “simple" initial.

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a “simple" initial.
3. $P = Ix_c + U$ with $\mathrm{cls}(P) = c$, $\mathrm{init}(P) = I$.

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a "simple" initial.
3. $P = Ix_c + U$ with $\text{cls}(P) = c$, $\text{init}(P) = I$.
4. If $I = 1$, then we can eliminate x_c:

$$\text{Zero}(\mathbb{H}) = \text{Zero}(\{P\} \cup \{\mathbb{H}|_{x_c=U})\})$$

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a “simple" initial.
3. $P = Ix_c + U$ with $\mathrm{cls}(P) = c$, $\mathrm{init}(P) = I$.
4. If $I = 1$, then we can eliminate x_c:

$$\mathrm{Zero}(\mathbb{H}) = \mathrm{Zero}(\{P\} \cup \{\mathbb{H}|_{x_c=U})\})$$

5. If $I \neq 1$, then

$$\mathrm{Zero}(\mathbb{H}) = \mathrm{Zero}(\mathbb{H} \cup \{I+1\}) \cup \mathrm{Zero}(\mathbb{H} \cup \{I\})$$

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a “simple" initial.
3. $P = Ix_c + U$ with $\mathrm{cls}(P) = c$, $\mathrm{init}(P) = I$.
4. If $I = 1$, then we can eliminate x_c:

$$\mathrm{Zero}(\mathbb{H}) = \mathrm{Zero}(\{P\} \cup \{\mathbb{H}|_{x_c=U})\})$$

5. If $I \neq 1$, then

$$\begin{aligned} \mathrm{Zero}(\mathbb{H}) &= \mathrm{Zero}(\mathbb{H} \cup \{I+1\}) \cup \mathrm{Zero}(\mathbb{H} \cup \{I\}) \\ &= \mathrm{Zero}((\mathbb{H} \setminus \{P\}) \cup \{x_c + U, I+1\}) \cup \end{aligned}$$

Top-Down Algorithm for Zero Decomposition (I)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a “simple" initial.
3. $P = Ix_c + U$ with $\text{cls}(P) = c$, $\text{init}(P) = I$.
4. If $I = 1$, then we can eliminate x_c:

$$\text{Zero}(\mathbb{H}) = \text{Zero}(\{P\} \cup \{\mathbb{H}|_{x_c=U})\})$$

5. If $I \neq 1$, then

$$\begin{aligned} \text{Zero}(\mathbb{H}) &= \text{Zero}(\mathbb{H} \cup \{I+1\}) \cup \text{Zero}(\mathbb{H} \cup \{I\}) \\ &= \text{Zero}((\mathbb{H} \setminus \{P\}) \cup \{x_c + U, I+1\}) \cup \\ &\quad \text{Zero}((\mathbb{H} \setminus \{P\}) \cup \{I, U\}). \end{aligned}$$

Top-Down Algorithm for Zero Decomposition (II)

TDZDT. Input: $\mathbb{P}$ a finite Boolean polynomial set.

1. $\mathbb{H}$: the polynomials with the highest class in $\mathbb{P}$.
2. P: a polynomial in $\mathbb{H}$ with a “simple" initial.
3. $P = Ix_c + U$ with $\mathrm{cls}(P) = c$, $\mathrm{init}(P) = I$.
4. If $I = 1$, then we can eliminate x_c:

$$\mathrm{Zero}(\mathbb{H}) = \mathrm{Zero}(\{P\} \cup \{\mathbb{H}|_{x_c=U})\})$$

5. If $I \neq 1$, then

$$\begin{aligned}\mathrm{Zero}(\mathbb{H}) &= \mathrm{Zero}(\mathbb{H} \cup \{I+1\}) \cup \mathrm{Zero}(\mathbb{H} \cup \{I\}) \\ &= \mathrm{Zero}((\mathbb{H} \setminus \{P\}) \cup \{x_c + U, I+1\}) \cup \\ &\quad \mathrm{Zero}((\mathbb{H} \setminus \{P\}) \cup \{IU + U + I\}).\end{aligned}$$

Properties of the Top-Down Algorithm

- It gives a disjoint monic decomposition:

$$\overline{\text{Zero}}(\mathbb{P}) = \cup_{i=1}^{s} \overline{\text{Zero}}(\mathcal{A}_i)$$

$$|\overline{\text{Zero}}(\mathbb{P})| = \sum_{i=1}^{s} 2^{\textbf{dim}(\mathcal{A}_i)}.$$

- The algorithm does not need polynomial multiplications and the degree of all the polynomials occurring in the algorithm is bounded by $\max_{P \in \mathbb{P}} \textbf{deg}(P)$.

Properties of the Top-Down Algorithm(II)

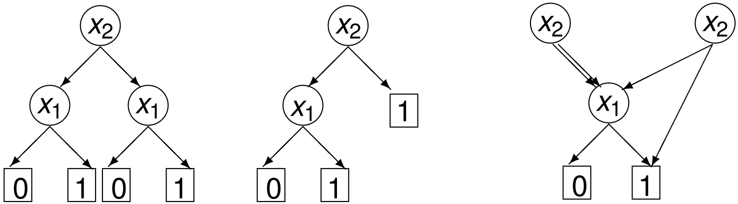
- It gives a disjoint monic decomposition:

$$\overline{\mathrm{Zero}}(\mathbb{P}) = \cup_{i=1}^{s} \overline{\mathrm{Zero}}(\mathcal{A}_i)$$

$$|\overline{\mathrm{Zero}}(\mathbb{P})| = \sum_{i=1}^{s} 2^{\mathbf{dim}(\mathcal{A}_i)}.$$

- One round of elimination from x_n to x_1 needs $O(nl)$ polynomial arithmetic operations where $l = |\mathbb{P}|$.

Shared Zero-suppressed BDD: SZDD



$P_1 = x_2 x_1 + x_1$ $P_2 = x_2 + x_1$ SZDD for $\{P_1, P_2\}$

Figure: SZDD for a polynomial set

Minto, S. Zero-Sppressed BDDs for Set Manipulation, *Proc. ACM Design Automation*, 1993.

Experimental Results with a Class of Stream Ciphers

Nonlinear Filter Generators

LFSR of length L:

Initial State: $S_0 = (s_0, s_1, \dots, s_{L-1}) \in \mathbf{F}_2^L$

An infinite sequence satisfying

$s_i = c_1 s_{i-1} + c_2 s_{i-2} + \cdots c_L s_{i-L}, i = L, L+1, \cdots$.

Nonlinear Filter.

$f(x_1, \dots, x_m)$: a Boolean polynomial with m variables.

A new sequence: $z_i = f(s_{i-m}, \dots, s_{i-1}), i = m, m+1, \cdots$.

The Test Problem. Given f, c_i, and $z_m, z_{m+1}, \dots, z_{r \cdot m}$, recover the initial state S_0 from the following algebraic equations:

$$z_i = f(s_{i-m}, \dots, s_{i-1}), i = m, m+1, \cdots, r \cdot m.$$

Filtering Functions Used in the Experiments

- CanFil 1, $x_1x_2x_3 + x_1x_4 + x_2x_5 + x_3$
- CanFil 2, $x_1x_2x_3 + x_1x_2x_4 + x_1x_2x_5 + x_1x_4 + x_2x_5 + x_3 + x_4 + x_5$
- CanFil 3, $x_2x_3x_4x_5 + x_1x_2x_3 + x_2x_4 + x_3x_5 + x_4 + x_5$
- CanFil 4, $x_1x_2x_3 + x_1x_4x_5 + x_2x_3 + x_1$
- CanFil 5, $x_2x_3x_4x_5 + x_2x_3 + x_1$
- CanFil 6, $x_1x_2x_3x_5 + x_2x_3 + x_4$
- CanFil 7, $x_1x_2x_3 + x_2x_3x_4 + x_2x_3x_5 + x_1 + x_2 + x_3$
- CanFil 8, $x_1x_2x_3 + x_2x_3x_6 + x_1x_2 + x_3x_4 + x_5x_6 + x_4 + x_5$
- CanFil 9,
  $x_2x_4x_5x_7 + x_2x_5x_6x_7 + x_3x_4x_6x_7 + x_1x_2x_4x_7 + x_1x_3x_4x_7 + x_1x_3x_6x_7 + x_1x_4x_5x_7 + x_1x_2x_5x_7 + x_1x_2x_6x_7 + x_1x_4x_6x_7 + x_3x_4x_5x_7 + x_2x_4x_6x_7 + x_3x_5x_6x_7 + x_1x_3x_5x_7 + x_1x_2x_3x_7 + x_3x_4x_5 + x_3x_4x_7 + x_3x_6x_7 + x_5x_6x_7 + x_2x_6x_7 + x_1x_4x_6 + x_1x_5x_7 + x_2x_4x_5 + x_2x_3x_7 + x_1x_2x_7 + x_1x_4x_5 + x_6x_7 + x_4x_6 + x_4x_7 + x_5x_7 + x_2x_5 + x_3x_4 + x_3x_5 + x_1x_4 + x_2x_7 + x_6 + x_5 + x_2 + x_1$
- CanFil 10, $x_1x_2x_3 + x_2x_3x_4 + x_2x_3x_5 + x_6x_7 + x_3 + x_2 + x_1$

Main Efficiency Issues

- Large Expressions.

  Currently, not the major problem.

  Improvement Techniques:

  - Using SZDD to represent Boolean polynomials
  - Using annihilator to reduce the degree
  - Using monic polynomials to keep the degree low

Main Efficiency Issues

- Large Expressions.
  Currently, not the major problem.
  Improvement Techniques:
  - Using SZDD to represent Boolean polynomials
  - Using annihilator to reduce the degree
  - Using monic polynomials to keep the degree low
- Branch Control/Number of Solutions.
  Number of branches/solutions strongly related to speed.
  c/s mostly ranges from 1/20 to 4.

Main Efficiency Issues

- Large Expressions.
  Currently, not the major problem.
  Improvement Techniques:
  - Using SZDD to represent Boolean polynomials
  - Using annihilator to reduce the degree
  - Using monic polynomials to keep the degree low
- Branch Control/Number of Solutions.
  Number of branches/solutions strongly related to speed.
  c/s mostly ranges from 1/20 to 4.
  Our current system works fine:
  But, this is the major time consuming part.

Main Efficiency Issues

- Large Expressions.

  Currently, not the major problem.
  Improvement Techniques:
  - Using SZDD to represent Boolean polynomials
  - Using annihilator to reduce the degree
  - Using monic polynomials to keep the degree low
- Branch Control/Number of Solutions.

  Number of branches/solutions strongly related to speed.
  c/s mostly ranges from 1/20 to 4.
  Our current system works fine:
  But, this is the major time consuming part.

  Solutions:
  - Using Parallel computation.
  - Find new techniques to reduce the branch.

Cryptanalysis of stream ciphers based on nonlinear filter generators can be reduced to solving equations over $\mathbf{F}_2$.
CS Method: Algorithm **TDZDTA** implemented with C++.
GB Method: F4 algorithm in Magma.
Machine: PC with a 3.19G CPU and 2G memory

	L (# of variables)	40	60	81	100	128
CanFil1 Deg=3	time for CS	0.04	0.00	0.01	0.05	0.06
	time for GB	0.91	0.43	8.12	3.61	1997.22
	# of polynomials	1.3L	1.9L	1.9L	1.4L	1.8L
CanFil2 Deg=3	time for CS	0.03	0.05	0.02	0.10	0.07
	time for GB	0.92	30.65	0.02	55.09	•
	# of polynomials	1.1L	1.2L	1.7L	1.4L	1.7L
CanFil3 Deg=4	time for CS	1.77	0.01	0.29	0.76*	1.27*
	time for GB	178.57	1.68	•	1.99*	•
	# of polynomials	1.6L	1.9L	2L	1.2L	L
CanFil4 Deg=3	time for CS	0.63	0.01	0.01	0.01*	0.02*
	time for GB	0.65	2.24	0.39	0.99*	22.57*
	# of polynomials	1.5L	2.8L	1.9L	1.5L	1.4L
CanFil5 Deg=4	time for CS	0.00	0.00	0.00	0.01	0.01
	time for GB	0.10	0.06	0.10	0.50	0.85
	# of polynomials	L	L	L	L	L

•: Memory overflow.

CanFil6 Deg=4	time for CS	0.01	0.00	0.01	0.03	0.06
	time for GB	0.24	0.09	0.01	0.65	•
	# of polynomials	1.3L	1.8L	1.8L	1.6L	1.8L
CanFil7 Deg=3	time for CS	0.01	0.01	0.01	0.07	0.07
	time for GB	0.27	0.40	0.01	831.89	•
	# of polynomials	L	2L	1.9L	1.5L	1.7L
CanFil8 Deg=3	time for CS	0.02	0.03	0.02	0.23	0.22
	time for GB	0.88	0.56	92.51	20.03	•
	# of polynomials	1.1L	L	1.9L	1.4L	1.7L
CanFil9 Deg=4	time for CS	4.83*	0.56	1.63	1.93	50.78*
	time for GB	•	90.49	1.63	•	•
	# of polynomials	1.2L	1.7L	1.4L	1.1L	1.7L
CanFil10 Deg=3	time for CS	0.17	0.06	0.06	0.10	0.32
	time for GB	28.72	2.21	492.16	•	•
	# of polynomials	1.1L	1.5L	1.5L	1.4L	1.6L

•: Memory overflow.

Observations

- r ranges from 1 to 2.8: we need at most $3L$ equations in order to find a unique solution.
- For the system with rL equations, it is much faster than the system with L equations.
- Using SZDD significantly reduces the speed.
- Our algorithm produces many branches which share many polynomials.

Conclusion

1. We give the monic and disjoint monic zero decomposition theorems for polynomial equations over $\mathbf{F}_2$.
2. We may compute a Wu characteristic set of a Boolean polynomial system with a polynomial number of arithmetic operations.

Conclusion

1. We give the monic and disjoint monic zero decomposition theorems for polynomial equations over $\mathbf{F}_2$.
2. We may compute a Wu characteristic set of a Boolean polynomial system with a polynomial number of arithmetic operations.
3. The method is comparable with F5 for moderately large size polynomial systems.
4. For very large systems, we still need improvements.

Further Work

1. CS Program System: Better techniques of branch control; good parallel strategies.

Further Work

1. CS Program System: Better techniques of branch control; good parallel strategies.
2. CS Method for finite fields.

   Gao and Huang, A Characteristic Set Method for Equation Solving in Finite Fields, MM-Preprints, Vol. 26, 2008.

Further Work

1. CS Program System: Better techniques of branch control; good parallel strategies.
2. CS Method for finite fields.

   Gao and Huang, A Characteristic Set Method for Equation Solving in Finite Fields, MM-Preprints, Vol. 26, 2008.
3. Approximate/probabilistic/quantum algorithms.

   Is there a polynomial approximate/probabilistic/quantum algorithm to solve Boolean equations?

Thanks !