Unification modulo Chaining

Siva Anantharaman[1] Christopher Bouchard[2]
Paliath Narendran[2] Michael Rusinowitch[3]

[1]LIFO - Université d'Orléans (France), siva@univ-orleans.fr

[2]University at Albany–SUNY (USA), dran@cs.albany.edu

[3]Loria-INRIA Lorraine, Nancy (France), rusi@loria.fr

Language and Automata Theory and Applications 2012

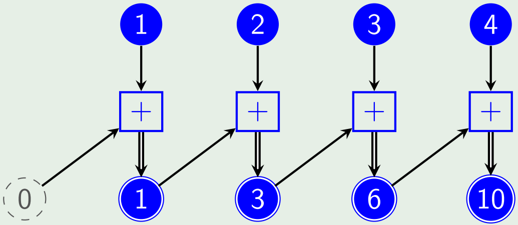
Section 1

Introduction

What is Chaining?

- Apply an operation to each element of a list
- Operation takes the current element and the previous result

Example



Cipher Block Chaining

- Technique for encryption
- Masks each message block with the previous result before encrypting

Example

$$[\, a,\ b,\ c \,]$$

$$\Downarrow$$

$$[\, e_k(a \oplus x),\ e_k(b \oplus e_k(a \oplus x)),\ e_k(c \oplus e_k(b \oplus e_k(a \oplus x))) \,]$$

Cipher Block Chaining

- Technique for encryption
- Masks each message block with the previous result before encrypting

Example

$$[\,a,\ b,\ c\,]$$

$$\Downarrow$$

$$[\,e_k(a \oplus x),\ e_k(b \oplus e_k(a \oplus x)),\ e_k(c \oplus e_k(b \oplus e_k(a \oplus x)))\,]$$

Cipher Block Chaining

- Technique for encryption
- Masks each message block with the previous result before encrypting

Example

$$[\, a,\ b,\ c \,]$$

$$\Downarrow$$

$$[\, e_k(a \oplus x),\ e_k(b \oplus e_k(a \oplus x)),\ e_k(c \oplus e_k(b \oplus e_k(a \oplus x))) \,]$$

Cipher Block Chaining

- Technique for encryption
- Masks each message block with the previous result before encrypting

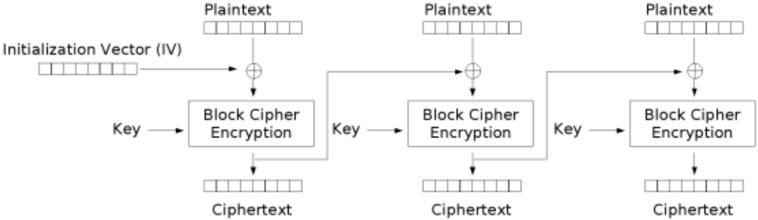
Example

$$[\,a,\ b,\ c\,]$$

$$\Downarrow$$

$$[\,e_k(a \oplus x),\ e_k(b \oplus e_k(a \oplus x)),\ e_k(c \oplus e_k(b \oplus e_k(a \oplus x)))\,]$$

Cipher Block Chaining



Cipher Block Chaining (CBC) mode encryption

Unification

- Given set of equations over terms
- Find a satisfying assignment for variables: Unifier

Example

$$f(x, c) \stackrel{?}{=} f(g(a, b), y)$$

Unification

- Given set of equations over terms
- Find a satisfying assignment for variables: Unifier

Example

$$f(x, c) \stackrel{?}{=} f(g(a, b), y)$$

Unification

- Given set of equations over terms
- Find a satisfying assignment for variables: Unifier

Example

$$f(x, c) \stackrel{?}{=} f(g(a, b), y)$$

$$\sigma = \{\, x := g(a, b),\ y := c \,\}$$

Unification

- Given set of equations over terms
- Find a satisfying assignment for variables: Unifier

Example

$$f(g(a, b), c) = f(g(a, b), c)$$

$$\sigma = \{ x := g(a, b),\ y := c \}$$

Equational Unification

- Unification modulo a set of axioms E
- Given a set of equations $\mathcal{EQ} = \{\, s_1 \stackrel{?}{=} t_1, \ldots, s_n \stackrel{?}{=} t_n \,\}$
- σ is an *E-unifier* of $\mathcal{EQ}$ iff:

$$\sigma(s_1) =_E \sigma(t_1), \ldots, \sigma(s_n) =_E \sigma(t_n)$$

Equational Unification

Unification modulo Commutativity:

Example

$$a + x \stackrel{?}{=}_C b + y$$

Equational Unification

Unification modulo Commutativity:

Example

$$a + x \stackrel{?}{=}_C b + y$$

Equational Unification

Unification modulo Commutativity:

Example

$$a + x \stackrel{?}{=}_{C} b + y$$

Equational Unification

Unification modulo Commutativity:

Example

$$a + x \stackrel{?}{=}_C \; b + y$$

$$\sigma = \{\, x := b,\; y := a \,\}$$

Equational Unification

Unification modulo Commutativity:

Example

$$a + b =_C b + a$$

$$\sigma = \{ x := b,\ y := a \}$$

Equational Unification

Unification modulo Commutativity:

Example

$$a + b =_C a + b$$

$$\sigma = \{ x := b,\ y := a \}$$

Rewrite System

- Axioms:

$$bc(nil, z) = nil$$

$$bc(cons(x, Y), z) = cons(h(x, z),\ bc(Y, h(x, z)))$$

Rewrite System

- Rewrite Rules:

$$bc(nil,\ z) \to nil$$

$$bc(cons(x, Y),\ z) \to cons(h(x, z),\ bc(Y,\ h(x, z)))$$

- Confluent, terminating
- We show unification modulo this theory is finitary

Notation

- Two types: Elements (τ_e) and Lists (τ_l)
- Elements use lowercase variables (e.g. x)
- Lists use uppercase variables (e.g. Y)
- Function symbols are typed:

$$bc : \tau_l \times \tau_e \to \tau_l \qquad cons : \tau_e \times \tau_l \to \tau_l$$
$$h : \tau_e \times \tau_e \to \tau_e \qquad nil : \tau_l$$

Standard Form

- Equations given in a standard form:

$$U \stackrel{?}{=} V \qquad u \stackrel{?}{=} v$$
$$U \stackrel{?}{=} nil \qquad u \stackrel{?}{=} c$$
$$U \stackrel{?}{=} cons(v, W) \qquad u \stackrel{?}{=} h(v, w)$$
$$U \stackrel{?}{=} bc(V, w)$$

Inference Rules

- Reason over sets of equations
- Preserve unifiability

Example

$$\frac{\mathcal{EQ} \uplus \{\, u \stackrel{?}{=} f(v,w),\ u \stackrel{?}{=} f(x,y) \,\}}{\mathcal{EQ} \cup \{\, u \stackrel{?}{=} f(v,w),\ x \stackrel{?}{=} v,\ y \stackrel{?}{=} w \,\}}$$

Inference Rules

- Reason over sets of equations
- Preserve unifiability

Example

$$\frac{\mathcal{E}\mathcal{Q} \uplus \{\, u \stackrel{?}{=} f(v, w),\ u \stackrel{?}{=} f(x, y) \,\}}{\mathcal{E}\mathcal{Q} \cup \{\, u \stackrel{?}{=} f(v, w),\ x \stackrel{?}{=} v,\ y \stackrel{?}{=} w \,\}}$$

How to Interpret h?

- h could be:
  - Uninterpreted: $\mathcal{BC}_0$
  - Interpreted as encryption with XOR mask: $\mathcal{BC}_1$

- Semi-cancellative

$$\frac{u \stackrel{?}{=} h(v,w), u \stackrel{?}{=} h(v,x)}{u \stackrel{?}{=} h(v,w), x \stackrel{?}{=} w} \qquad \frac{u \stackrel{?}{=} h(v,w), u \stackrel{?}{=} h(x,w)}{u \stackrel{?}{=} h(v,w), x \stackrel{?}{=} v}$$

How to Interpret h?

- h could be:
  - Uninterpreted: $\mathcal{BC}_0$
  - Interpreted as encryption with XOR mask: $\mathcal{BC}_1$ (CBC)

- Semi-cancellative

$$\frac{u \stackrel{?}{=} h(v, w), u \stackrel{?}{=} h(v, x)}{u \stackrel{?}{=} h(v, w), x \stackrel{?}{=} w} \qquad \frac{u \stackrel{?}{=} h(v, w), u \stackrel{?}{=} h(x, w)}{u \stackrel{?}{=} h(v, w), x \stackrel{?}{=} v}$$

How to Interpret h?

- h could be:
  - Uninterpreted: $\mathcal{BC}_0$
  - Interpreted as encryption with XOR mask: $\mathcal{BC}_1$ (CBC)

- Semi-cancellative

$$\frac{u \stackrel{?}{=} h(v, w), u \stackrel{?}{=} h(v, x)}{u \stackrel{?}{=} h(v, w), x \stackrel{?}{=} w} \qquad \frac{u \stackrel{?}{=} h(v, w), u \stackrel{?}{=} h(x, w)}{u \stackrel{?}{=} h(v, w), x \stackrel{?}{=} v}$$

Section 2

Algorithm

Algorithm

- Given: A set of equations in standard notation
- Goal: Get list equations into dag-solved form

Dag-Solved Form

A system of equations

$$\mathcal{EQ} = \{\, x_1 \overset{?}{=} t_1,\, x_2 \overset{?}{=} t_2,\, \ldots,\, x_n \overset{?}{=} t_n \,\}$$

is in dag-solved form iff:

- $\forall i : x_i$ is a variable
- $\forall i, j : i \neq j \Rightarrow x_i \neq x_j$
- $\forall i \leq j : x_i \notin \mathrm{Var}(t_j)$

Dag-Solved Form

A system of equations

$$\mathcal{EQ} = \{\, x_1 \stackrel{?}{=} t_1,\, x_2 \stackrel{?}{=} t_2,\, \ldots,\, x_n \stackrel{?}{=} t_n \,\}$$

is in dag-solved form iff:

- $\forall i : x_i$ is a variable
- $\forall i, j : i \neq j \Rightarrow x_i \neq x_j$
- $\forall i \leq j : x_i \notin \mathrm{Var}(t_j)$

Dag-Solved Form

Example

$$\{\, U \stackrel{?}{=} bc(V, w),\ V \stackrel{?}{=} cons(x, Y),\ w \stackrel{?}{=} a \,\}$$

Not Dag-Solved

$$\{\, U \stackrel{?}{=} cons(v, W),\ W \stackrel{?}{=} cons(x, U) \,\}$$

$$\{\, U \stackrel{?}{=} cons(v, W),\ U \stackrel{?}{=} bc(X, y) \,\}$$

Dag-Solved Form

Example

$$\{\, U \stackrel{?}{=} bc(V, w),\ V \stackrel{?}{=} cons(x, Y),\ w \stackrel{?}{=} a \,\}$$

Not Dag-Solved

$$\{\, U \stackrel{?}{=} cons(v, W),\ W \stackrel{?}{=} cons(x, U) \,\}$$

$$\{\, U \stackrel{?}{=} cons(v, W),\ U \stackrel{?}{=} bc(X, y) \,\}$$

Dag-Solved Form

Example

$$\{\, U \stackrel{?}{=} bc(V, w),\ V \stackrel{?}{=} cons(x, Y),\ w \stackrel{?}{=} a \,\}$$

Not Dag-Solved

$$\{\, U \stackrel{?}{=} cons(v, W),\ W \stackrel{?}{=} cons(x, U) \,\}$$

$$\{\, {\color{red}U} \stackrel{?}{=} cons(v, W),\ {\color{red}U} \stackrel{?}{=} bc(X, y) \,\}$$

Algorithm

(L1) Variable Elimination:

$$\frac{\mathcal{EQ} \uplus \{\, U \overset{?}{=} V \,\}}{[V/U](\mathcal{EQ}) \cup \{\, U \overset{?}{=} V \,\}} \quad \text{if } U \in \mathrm{Var}(\mathcal{EQ})$$

(L2) Cancellation on *cons*:

$$\frac{\mathcal{EQ} \uplus \{\, U \overset{?}{=} \mathit{cons}(v, W),\ U \overset{?}{=} \mathit{cons}(x, Y) \,\}}{\mathcal{EQ} \cup \{\, U \overset{?}{=} \mathit{cons}(v, W),\ x \overset{?}{=} v,\ Y \overset{?}{=} W \,\}}$$

Algorithm

(L3a) Nil Solution 1:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} nil \,\}}{\mathcal{EQ} \cup \{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil \,\}}$$

(L3b) Nil Solution 2:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ V \stackrel{?}{=} nil \,\}}{\mathcal{EQ} \cup \{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil \,\}}$$

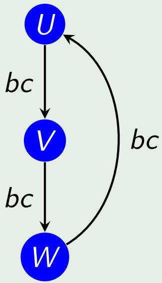
(L3c) Nil Solution 3:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w) \,\}}{\mathcal{EQ} \cup \{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil \,\}} \quad \text{if } V >_{bc}^{*} U$$

nil Solutions

Example

$$\{\, U \stackrel{?}{=} bc(V, x),\ V \stackrel{?}{=} bc(W, y),$$
$$W \stackrel{?}{=} bc(U, z) \,\}$$



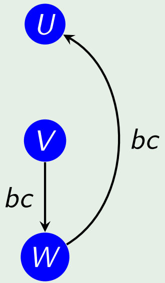
nil Solutions

Example

$$\{\, U \overset{?}{=} bc(V, x),\; V \overset{?}{=} bc(W, y),$$
$$W \overset{?}{=} bc(U, z)\,\}$$

$$\Downarrow$$

$$\{\, U \overset{?}{=} \textcolor{red}{nil},\; V \overset{?}{=} \textcolor{red}{nil},\; V \overset{?}{=} bc(W, y),$$
$$W \overset{?}{=} bc(U, z)\,\}$$



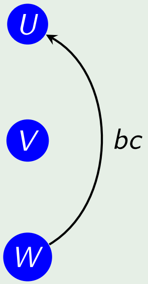
nil Solutions

Example

$$\{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil,\ V \stackrel{?}{=} bc(W, y),\ W \stackrel{?}{=} bc(U, z) \,\}$$

$$\Downarrow$$

$$\{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil,\ W \stackrel{?}{=} nil,\ W \stackrel{?}{=} bc(U, z) \,\}$$



nil Solutions

Example

$$\{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil,\ W \stackrel{?}{=} nil,$$
$$W \stackrel{?}{=} bc(U, z) \,\}$$

$$\Downarrow$$

$$\{\, U \stackrel{?}{=} nil,\ V \stackrel{?}{=} nil,\ W \stackrel{?}{=} nil \,\}$$

Algorithm

(L4a) Semi-cancellation on *bc*:

$$\frac{\mathcal{E}\mathcal{Q} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, w) \,\}}{\mathcal{E}\mathcal{Q} \cup \{\, U \stackrel{?}{=} bc(V, w),\ X \stackrel{?}{=} V \,\}}$$

Algorithm

(L4a) Semi-cancellation on bc:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, w) \,\}}{\mathcal{EQ} \cup \{\, U \stackrel{?}{=} bc(V, w),\ X \stackrel{?}{=} V \,\}}$$

Algorithm

(L4b) Pushing *bc* below *cons*:

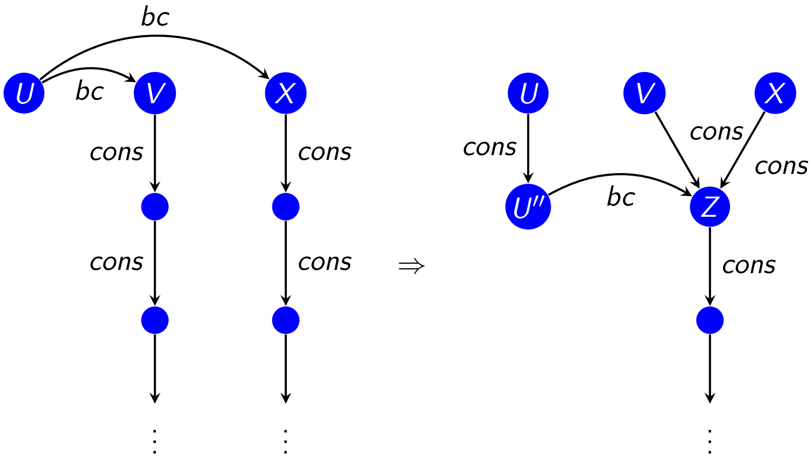
$$\frac{\mathcal{E}\mathcal{Q} \uplus \{\, U \overset{?}{=} bc(V, w),\ U \overset{?}{=} bc(X, y) \,\}}{\begin{array}{r}\mathcal{E}\mathcal{Q} \cup \{\, V \overset{?}{=} cons(v', Z),\ X \overset{?}{=} cons(x', Z), \\ U \overset{?}{=} cons(u', U''),\ U'' \overset{?}{=} bc(Z, u'), \\ u' \overset{?}{=} h(v', w),\ u' \overset{?}{=} h(x', y) \,\} \quad \end{array}} \quad \text{if } U \in \textbf{nonnil}$$

Algorithm

(L4b) Pushing *bc* below *cons*:

$$\frac{\mathcal{E}\mathcal{Q} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{r}\mathcal{E}\mathcal{Q} \cup \{\, V \stackrel{?}{=} cons(v', Z),\ X \stackrel{?}{=} cons(x', Z), \\ U \stackrel{?}{=} cons(u', U''),\ U'' \stackrel{?}{=} bc(Z, u'), \\ u' \stackrel{?}{=} h(v', w),\ u' \stackrel{?}{=} h(x', y) \,\}\end{array}} \quad \text{if } U \in \textbf{nonnil}$$

Pushing *bc* below *cons*



Algorithm

(L5) Splitting:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} cons(v, W),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{c}\mathcal{EQ} \cup \{\, U \stackrel{?}{=} cons(v, W),\ W \stackrel{?}{=} bc(V', v), \\ X \stackrel{?}{=} cons(z, V'),\ v \stackrel{?}{=} h(z, y) \,\}\end{array}}$$

Algorithm

(L5) Splitting:

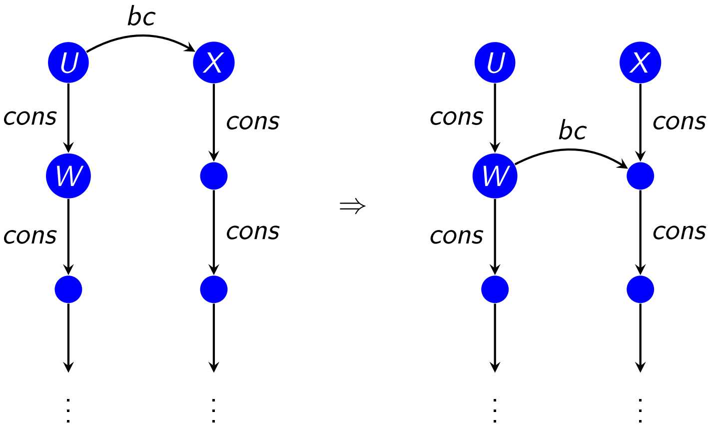
$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} cons(v, W),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{c}\mathcal{EQ} \cup \{\, U \stackrel{?}{=} cons(v, W),\ W \stackrel{?}{=} bc(V', v), \\ X \stackrel{?}{=} cons(z, V'),\ v \stackrel{?}{=} h(z, y) \,\}\end{array}}$$

Algorithm

(L5) Splitting:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} cons(v, W),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{c}\mathcal{EQ} \cup \{\, U \stackrel{?}{=} cons(v, W),\ W \stackrel{?}{=} bc(V', v), \\ X \stackrel{?}{=} cons(z, V'),\ v \stackrel{?}{=} h(z, y) \,\}\end{array}}$$

Splitting



Nondeterminism

- Need to explore space of unifiers
- Remaining rules are nondeterministic

Nondeterminism

- Need to explore space of unifiers
- Remaining rules are nondeterministic
- Stop here for unifiability of $\mathcal{BC}_0$

Algorithm

(L8) Nil Solution Branch:

$$\frac{\mathcal{EQ} \uplus \{\, U \overset{?}{=} bc(V, w),\ U \overset{?}{=} bc(X, y) \,\}}{\mathcal{EQ} \cup \{\, U \overset{?}{=} nil,\ V \overset{?}{=} nil,\ X \overset{?}{=} nil \,\}}$$

Algorithm

(L9) Cancellation Branch on bc:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{r}\mathcal{EQ} \cup \{\, V \stackrel{?}{=} cons(v', Z),\ X \stackrel{?}{=} cons(x', Z), \\ U \stackrel{?}{=} cons(u', U''),\ U'' \stackrel{?}{=} bc(Z, u'), \\ u' \stackrel{?}{=} h(v', w),\ u' \stackrel{?}{=} h(x', y) \,\}\end{array}}$$

Algorithm

(L9) Cancellation Branch on *bc*:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{r} \mathcal{EQ} \cup \{\, V \stackrel{?}{=} cons(v', Z),\ X \stackrel{?}{=} cons(x', Z), \\ U \stackrel{?}{=} cons(u', U''),\ U'' \stackrel{?}{=} bc(Z, u'), \\ u' \stackrel{?}{=} h(v', w),\ u' \stackrel{?}{=} h(x', y) \,\} \end{array}} \quad \text{if } U \in \textbf{nonnil}$$

Algorithm

(L9) Cancellation Branch on bc:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, y) \,\}}{\begin{array}{r}\mathcal{EQ} \cup \{\, V \stackrel{?}{=} cons(v', Z),\ X \stackrel{?}{=} cons(x', Z), \\ U \stackrel{?}{=} cons(u', U''),\ U'' \stackrel{?}{=} bc(Z, u'), \\ u' \stackrel{?}{=} h(v', w),\ u' \stackrel{?}{=} h(x', y) \,\} \quad\quad\end{array}} \qquad \text{~~if } U \in \textbf{nonnil}\text{~~}$$

Algorithm

(L10) Standard Unification Branch on *bc*:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, y) \,\}}{\mathcal{EQ} \cup \{\, U \stackrel{?}{=} bc(V, w),\ X \stackrel{?}{=} V,\ y \stackrel{?}{=} w \,\}}$$

Algorithm

(L10) Standard Unification Branch on *bc*:

$$\frac{\mathcal{EQ} \uplus \{\, U \stackrel{?}{=} bc(V, w),\ U \stackrel{?}{=} bc(X, y) \,\}}{\mathcal{EQ} \cup \{\, U \stackrel{?}{=} bc(V, w),\ X \stackrel{?}{=} V,\ y \stackrel{?}{=} w \,\}}$$

Complexity Results

	Unifiability	Unification
$\mathcal{BC}_0$	P	NP-Complete
$\mathcal{BC}_1$	NP-Complete	NP-Complete

Section 3

Example

Example

- Suppose we have the following protocol:

$$\mathcal{A} \rightarrow \mathcal{B} : \{A, m\}_{kb}$$
$$\mathcal{B} \rightarrow \mathcal{A} : \{B, m\}_{ka}$$

- One block: Secure

- Cipher Block Chaining: Insecure

Example

- Interpret $h(u, v)$ as $e_{kb}(u \oplus v)$

$$\mathcal{A} \rightarrow \mathcal{B} : bc([\, a,\ m\,], c)$$

Example

- Interpret $h(u, v)$ as $e_{kb}(u \oplus v)$

$$\mathcal{A} \rightarrow \mathcal{B} : bc(cons(a, cons(m, nil)), c)$$

Example

- Interpret $h(u, v)$ as $e_{kb}(u \oplus v)$

$$\mathcal{A} \rightarrow \mathcal{B} : bc(cons(a, cons(m, nil)), c)$$

- Intruder sees

$$[\, h(a, c),\ h(m, h(a, c))\,]$$

Example

- Interpret $h(u, v)$ as $e_{kb}(u \oplus v)$

$$\mathcal{A} \rightarrow \mathcal{B} : bc(cons(a, cons(m, nil)), c)$$

- Intruder sees

$$cons(h(a, c), cons(h(m, h(a, c)), nil))$$

Example

- Interpret $h(u, v)$ as $e_{kb}(u \oplus v)$

$$\mathcal{A} \rightarrow \mathcal{B} : bc(cons(a, cons(m, nil)), c)$$

- Intruder sees

$$cons(h(a, c), cons(\color{red}{h(m, h(a, c))}, nil))$$

Example

- Intruder can send to B

$$bc(cons(i, L), d)$$

- To find a suitable attack message L, solve

$$bc(L, h(i, d)) \stackrel{?}{=} cons(h(m, h(a, c)), nil)$$

Example

- Convert to standard form:

$$bc(L, h(i, d)) \stackrel{?}{=} cons(h(m, h(a, c)), nil)$$

$$\Downarrow$$

$$\{ U \stackrel{?}{=} bc(L, w),\ U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),\ x \stackrel{?}{=} h(v_m, y),\ y \stackrel{?}{=} h(v_a, v_c),\ v_a \stackrel{?}{=} a,\ v_c \stackrel{?}{=} c,\ v_d \stackrel{?}{=} d,\ v_i \stackrel{?}{=} i,\ v_m \stackrel{?}{=} m \}$$

Example

- Convert to standard form:

$$bc(L, h(i, d)) \stackrel{?}{=} cons(h(m, h(a, c)), nil)$$

$$\Downarrow$$

$$\{\, U \stackrel{?}{=} bc(L, w),\ U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),\ x \stackrel{?}{=} h(v_m, y),\ y \stackrel{?}{=} h(v_a, v_c),\ v_a \stackrel{?}{=} a,\ v_c \stackrel{?}{=} c,\ v_d \stackrel{?}{=} d,\ v_i \stackrel{?}{=} i,\ v_m \stackrel{?}{=} m \,\}$$

Example

- Convert to standard form:

$$bc(L, h(i, d)) \stackrel{?}{=} cons(h(m, h(a, c)), nil)$$

$$\Downarrow$$

$$\{\, U \stackrel{?}{=} bc(L, w),\ U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),$$
$$x \stackrel{?}{=} h(v_m, y),\ y \stackrel{?}{=} h(v_a, v_c),\ \ldots \}$$

Example

$$\{\, U \stackrel{?}{=} bc(L, w),\ U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),$$
$$x \stackrel{?}{=} h(v_m, y),\ y \stackrel{?}{=} h(v_a, v_c),\ \ldots\}$$

$\Downarrow$ (L5) Splitting

$$\{\, U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),\ x \stackrel{?}{=} h(v_m, y)$$
$$y \stackrel{?}{=} h(v_a, v_c),\ N \stackrel{?}{=} bc(V_1, x),\ L \stackrel{?}{=} cons(z, V_1),$$
$$x \stackrel{?}{=} h(z, w),\ \ldots\}$$

Example

$$\{\, U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),\ x \stackrel{?}{=} h(v_m, y)$$
$$y \stackrel{?}{=} h(v_a, v_c),\ N \stackrel{?}{=} bc(V_1, x),\ L \stackrel{?}{=} cons(z, V_1),$$
$$x \stackrel{?}{=} h(z, w),\ \ldots \}$$

$\Downarrow$ (L3a) Nil Solution

$$\{\, U \stackrel{?}{=} cons(x, N),\ N \stackrel{?}{=} nil,\ w \stackrel{?}{=} h(v_i, v_d),\ x \stackrel{?}{=} h(v_m, y)$$
$$y \stackrel{?}{=} h(v_a, v_c),\ L \stackrel{?}{=} cons(z, V_1),\ x \stackrel{?}{=} h(z, w),\ V_1 \stackrel{?}{=} nil,\ \ldots \}$$

Example

- List equations now in dag-solved form
- Pass element equations to XOR unification algorithm
- Treat e_{kb} as an uninterpreted function symbol

$$\mathcal{XOR}\{\, w \stackrel{?}{=} e_{kb}(v_i \oplus v_d),\ x \stackrel{?}{=} e_{kb}(z \oplus w),\ x \stackrel{?}{=} e_{kb}(v_m \oplus y),\ y \stackrel{?}{=} e_{kb}(v_a \oplus v_c),\ \ldots\,\}$$

$$\Downarrow$$

$$\{\, z := e_{kb}(e_{kb}(m \oplus e_{kb}(a \oplus c)) \oplus e_{kb}(i \oplus d)),\ \ldots\,\}$$

Example

- Convert back to h

$$\{\, z := e_{kb}(e_{kb}(m \oplus e_{kb}(a \oplus c)) \oplus e_{kb}(i \oplus d)) \,\}$$

$$\Downarrow$$

$$\{z := h(h(m, h(a, c)), h(i, d))\}$$

- Putting everything together:

$$\sigma = \{\, L := cons(h(h(m, h(a, c)), h(i, d)), nil) \,\}$$

Example

- Convert back to h

$$\{\, z := e_{kb}(e_{kb}(m \oplus e_{kb}(a \oplus c)) \oplus e_{kb}(i \oplus d))\,\}$$

$$\Downarrow$$

$$\{z := h(h(m, h(a, c)), h(i, d))\}$$

- Putting everything together:

$$\sigma = \{\, L := cons(h(h(m, h(a, c)), h(i, d)), nil)\,\}$$

Section 4

Conclusion

Results

- Algorithm for Unification modulo Chaining
  - Sound and Complete
- Unification is finitary

Complexity Results

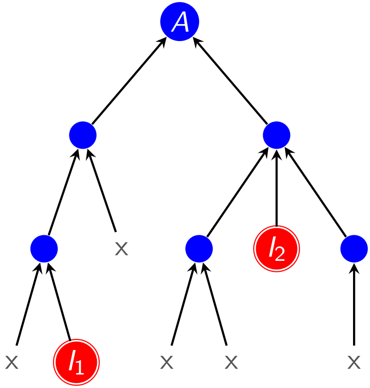
	Unifiability	Unification
$\mathcal{BC}_0$	P	NP-Complete
$\mathcal{BC}_1$	NP-Complete	NP-Complete

Future Work

- Decryption operators
  - Element decryption: $g(h(u, v), v) = u$
  - Decryption of CBC
- Implementation

Maude-NPA

- Analysis tool for cryptographic protocols
- Based on backwards narrowing and unification
- Seeking inclusion



Thank You

- cbou@cs.albany.edu
- Research supported in part by NSF grant CNS-0905286