A Formal Theory for the Complexity Class Associated with the Stable Marriage Problem

Dai Tri Man Lê

Joint work with Stephen Cook and Yuli Ye

Department of Computer Science
University of Toronto
Canada

LCC 2011
Two Aspects of Proof Complexity

1. Propositional Proof Complexity (\textit{Pitassi’s invited talk})
   - the lengths of proofs of tautologies in various proof systems

2. Bounded Arithmetic
   - the power of weak formal systems to prove theorems of interest in computer science

- Both are closely related to mainstream complexity theory
- (2) and (1) are related by “\textit{propositional translations}”
  - a proof in theory $T \rightsquigarrow$ uniform short proofs in propositional proof system $P_T$
  - bounded arithmetic = uniform version of propositional proof complexity
- “\textit{bounded}”: induction axioms are restricted to bounded formulas
Two Aspects of Proof Complexity

1. Propositional Proof Complexity (*Pitassi’s invited talk*)
   - the lengths of proofs of tautologies in various proof systems

2. Bounded Arithmetic
   - the power of weak formal systems to prove theorems of interest in computer science

- Both are closely related to mainstream complexity theory
- (2) and (1) are related by “propositional translations”
  - a proof in theory $T \rightsquigarrow$ uniform short proofs in propositional proof system $P_T$
  - bounded arithmetic = uniform version of propositional proof complexity
- “bounded”: induction axioms are restricted to bounded formulas
Motivation

Classify theorems according to the computational complexity of concepts needed to prove them.

Program in Chapter 9

1. Introduce a general method for associating a canonical minimal theory $VC$ for certain complexity classes $C$

   $$AC^0 \subseteq C \subseteq P$$

2. Given a theorem $\tau$, try to find the smallest complexity class $C$ such that

   $$VC \vdash \tau$$
“As a matter of fact, the subject of the book can almost be thought as developing the proof theory that is missing from the descriptive complexity approach to understanding complexity classes through logic.”

[Atserias ’11]
Outline of the talk

1. The complexity class CC
   - Interesting **natural complete problems**: stable marriage, lex-first maximal matching, comparator circuit value problem.

2. Use the Cook-Nguyen method to define a theory for CC

3. Discuss many open problems related to CC
Outline of the talk

1. The complexity class CC
   - Interesting natural complete problems: stable marriage, lex-first maximal matching, comparator circuit value problem.

2. Use the Cook-Nguyen method to define a theory for CC

3. Discuss many open problems related to CC
Comparator Circuits

- Originally invented for sorting, e.g.,
  - Batcher’s $O(\log^2 n)$-depth sorting networks ('68)
  - Ajtai-Komlós-Szemerédi (AKS) $O(\log n)$-depth sorting networks ('83)
- Can also be considered as boolean circuits.

**Comparator gate**

<table>
<thead>
<tr>
<th></th>
<th>$p$</th>
<th>$x$</th>
<th>$p \land q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q$</td>
<td>$y$</td>
<td></td>
<td>$p \lor q$</td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th></th>
<th>$w_0$</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$w_1$</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>$w_2$</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>$w_3$</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>$w_4$</td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>$w_5$</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Comparator Circuit Value (CCV) Problem (decision)

Given a comparator circuit with specified Boolean inputs, determine the output value of a designated wire.

Complexity classes

1. $\text{CC}^{\text{Subr}} = \{\text{decision problems log-space many-one-reducible to CCV}\}$
   - [Subramanian’s PhD thesis ’90], [Mayr-Subramanian ’92]
Comparator Circuit Value (CCv) Problem (decision)

Given a comparator circuit with specified Boolean inputs, determine the output value of a designated wire.

Complexity classes

1. $CC^{Subr} = \{ \text{decision problems log-space many-one-reducible to CCv} \}$
   - [Subramanian’s PhD thesis ’90], [Mayr-Subramanian ’92]

2. $CC = \{ \text{decision problems AC}^0 \text{ many-one-reducible to CCv} \}$
   - Complete problems: stable marriage, lex-first maximal matching...

3. $CC^* = \{ \text{decision problems AC}^0 \text{ oracle-reducible to CCv} \}$
   - Needed when developing a Cook-Nguyen style theory for CC
   - The function class FCC* is closed under composition

$NC^1 \subseteq NL \subseteq CC \subseteq CC^{Subr} \subseteq CC^* \subseteq P$
### Comparator Circuit Value ($\text{CCv}$) Problem (decision)

Given a comparator circuit with specified Boolean inputs, determine the output value of a designated wire.

<table>
<thead>
<tr>
<th>$w_0$</th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
<th>$w_4$</th>
<th>$w_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>?</td>
</tr>
</tbody>
</table>

### Complexity classes

1. $\text{CC}^{\text{Subr}} = \{ \text{decision problems log-space many-one-reducible to } \text{CCv} \}$
   - [Subramanian’s PhD thesis ’90], [Mayr-Subramanian ’92]

2. $\text{CC} = \{ \text{decision problems } \text{AC}^0 \text{ many-one-reducible to } \text{CCv} \}$
   - Complete problems: stable marriage, lex-first maximal matching...

3. $\text{CC}^* = \{ \text{decision problems } \text{AC}^0 \text{ oracle-reducible to } \text{CCv} \}$
   - Needed when developing a Cook-Nguyen style theory for \text{CC}
   - The function class $\text{FCC}^*$ is closed under composition

\[ \text{NC}^1 \subseteq \text{NL} \subseteq \text{CC} \subseteq \text{CC}^{\text{Subr}} \subseteq \text{CC}^* \subseteq \text{P} \]
Stable Marriage Problem (search version) (Gale-Shapley ’62)

- Given $n$ men and $n$ women together with their preference lists
- Find a stable marriage between men and women, i.e.,
  1. a perfect matching
  2. satisfies the stability condition: no two people of the opposite sex like each other more than their current partners

Preference lists

<table>
<thead>
<tr>
<th>Men:</th>
<th>a</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b</td>
<td>y</td>
<td>x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Women:</th>
<th>x</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>y</td>
<td>a</td>
<td>b</td>
</tr>
</tbody>
</table>
Stable Marriage Problem (search version) (Gale-Shapley ’62)

- Given $n$ men and $n$ women together with their preference lists
- Find a stable marriage between men and women, i.e.,
  1. a perfect matching
  2. satisfies the stability condition: no two people of the opposite sex like each other more than their current partners

Preference lists

<table>
<thead>
<tr>
<th>Men:</th>
<th>a</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b</td>
<td>y</td>
<td>x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Women:</th>
<th>x</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>y</td>
<td>a</td>
<td>b</td>
</tr>
</tbody>
</table>

stable marriage
The Stable Marriage Problem (search version) (Gale-Shapley ’62)

- Given \( n \) men and \( n \) women together with their preference lists
- Find a stable marriage between men and women, i.e.,
  1. a perfect matching
  2. satisfies the **stability condition**: no two people of the opposite sex like each other more than their current partners

**Preference lists**

<table>
<thead>
<tr>
<th>Men:</th>
<th>Women:</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>( x )</td>
</tr>
<tr>
<td>( a )</td>
<td>( x )</td>
</tr>
<tr>
<td>( x )</td>
<td>( a )</td>
</tr>
<tr>
<td>( y )</td>
<td>( x )</td>
</tr>
<tr>
<td>( b )</td>
<td>( y )</td>
</tr>
<tr>
<td>( y )</td>
<td>( a )</td>
</tr>
</tbody>
</table>

Stable marriage: \( a \) and \( x \) are matched, and no other pair prefers each other over their current partners.

Unstable marriage: \( b \) and \( y \) are matched, but they prefer each other over their current partners.
Stable Marriage Problem (search version) (Gale-Shapley ’62)

- Given $n$ men and $n$ women together with their preference lists
- Find a stable marriage between men and women, i.e.,
  1. a perfect matching
  2. satisfies the stability condition: no two people of the opposite sex like each other more than their current partners

Preference lists

<table>
<thead>
<tr>
<th>Men:</th>
<th>a</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b</td>
<td>y</td>
<td>x</td>
</tr>
<tr>
<td>Women:</td>
<td>x</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>a</td>
<td>b</td>
</tr>
</tbody>
</table>

Stable Marriage Problem (decision version)

Is a given pair of $(m, w)$ in the man-optimal (woman-optimal) stable marriage?
Lex-first maximal matching problem

Lex-first maximal matching

- Let $G$ be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node.
Lex-first maximal matching problem

Lex-first maximal matching

- Let $G$ be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node.
Lex-first maximal matching problem

Lex-first maximal matching

- Let $G$ be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node.
Lex-first maximal matching problem

Lex-first maximal matching

- Let $G$ be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node.
Lex-first maximal matching problem

Lex-first maximal matching

- Let $G$ be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node

Lex-first maximal matching problem (decision)

Is a given edge $\{u, v\}$ in the lex-first maximal matching of $G$?
Reducing lex-first maximal matching to $C_{cv}$
Reducing $\mathbb{C}_\text{CV}$ to lex-first maximal matching

$p_0 \quad \uparrow \quad p_1$
$q_0 \quad \downarrow \quad q_1$

$p_0 \quad q_0 \quad p_1 \quad q_1$

$x \quad y$
Reducing $\text{C}_{\text{CV}}$ to lex-first maximal matching

$p_0 \quad 1 \quad \rightarrow \quad 1 \quad p_1$
$q_0 \quad 1 \quad \rightarrow \quad 1 \quad q_1$
Reducing $\mathcal{C}_{\text{CV}}$ to lex-first maximal matching

\[
p_0 \quad 1 \quad \downarrow \quad 1 \quad p_1
\]
\[
q_0 \quad 1 \quad \downarrow \quad 1 \quad q_1
\]
Reducing $\mathbb{C}_\mathbb{C}_\mathbb{V}$ to lex-first maximal matching

\[
\begin{array}{cccc}
p_0 & 0 & \rightarrow & 1 & p_1 \\
q_0 & 1 & \rightarrow & 0 & q_1 \\
\end{array}
\]
Reducing $C_{CV}$ to lex-first maximal matching

\[ p_0 \quad 0 \quad 1 \quad p_1 \]
\[ q_0 \quad 1 \quad 0 \quad q_1 \]
Outline of the talk

1. The complexity class CC
   ▶ Interesting **natural complete problems**: stable marriage, lex-first maximal matching, comparator circuit value problem...

2. Use the Cook-Nguyen method to define a theory for CC

3. Discuss many open problems related to CC
Two-sorted language $\mathcal{L}_A^2$ (Zambella ’96)

Vocabulary $\mathcal{L}_A^2 = [0, 1, +, \cdot, | | ; \in, \leq, =_1, =_2]$

- Standard model $\mathbb{N}_2 = \langle \mathbb{N}, \text{finite subsets of } \mathbb{N} \rangle$
- $0, 1, +, \cdot, \leq, =$ have usual meaning over $\mathbb{N}$
- $|X| = \text{length of } X$
- Set membership $y \in X$

- “number” variables $x, y, z, \ldots$ (range over $\mathbb{N}$)
- “string” variables $X, Y, Z, \ldots$ (range over finite subsets of $\mathbb{N}$)
- Number terms are built from $x, y, z, \ldots, 0, 1, +, \cdot$ and $|X|, |Y|, |Z|, \ldots$
- The only string terms are variable $X, Y, Z, \ldots$
Two-sorted language $\mathcal{L}_A^2$ (Zambella '96)

Vocabulary $\mathcal{L}_A^2 = [0, 1, +, \cdot, \mid \mid ; \in, \leq, =_1, =_2]$

- Standard model $\mathbb{N}_2 = \langle \mathbb{N}, \text{finite subsets of } \mathbb{N} \rangle$
- $0, 1, +, \cdot, \leq, =$ have usual meaning over $\mathbb{N}$
- $|X| = \text{length of } X$
- Set membership $y \in X$

- “number” variables $x, y, z, \ldots$ (range over $\mathbb{N}$)
- “string” variables $X, Y, Z, \ldots$ (range over finite subsets of $\mathbb{N}$)
- Number terms are built from $x, y, z, \ldots, 0, 1, +, \cdot$ and $|X|, |Y|, |Z|, \ldots$
- The only string terms are variable $X, Y, Z, \ldots$

Note

The natural inputs for Turing machines and circuits are finite strings.
Two-sorted language $\mathcal{L}_A^2$ (Zambella ’96)

Vocabulary $\mathcal{L}_A^2 = \left[ 0, 1, +, \cdot, \mid \mid \in, \leq, =_1, =_2 \right]$
- Standard model $\mathbb{N}_2 = \langle \mathbb{N}, \text{finite subsets of } \mathbb{N} \rangle$
- $0, 1, +, \cdot, \leq, =$ have usual meaning over $\mathbb{N}$
- $|X| = \text{length of } X$
- Set membership $y \in X$

- “number” variables $x, y, z, \ldots$ (range over $\mathbb{N}$)
- “string” variables $X, Y, Z, \ldots$ (range over finite subsets of $\mathbb{N}$)
- Number terms are built from $x, y, z, \ldots, 0, 1, +, \cdot$ and $|X|, |Y|, |Z|, \ldots$
- The only string terms are variable $X, Y, Z, \ldots$

Note
The natural inputs for Turing machines and circuits are finite strings.

Definition ($\Sigma^B_0$ formula)

1. All the number quantifiers are bounded.
2. No string quantifiers (free string variables are allowed)
Two-sorted complexity classes

A two-sorted complexity class consists of relations $R(\vec{x}, \vec{X})$, where
- $\vec{x}$ are number arguments (in unary) and $\vec{X}$ are string arguments

**Definition (Two-sorted AC$^0$)**

A relation $R(\vec{x}, \vec{X})$ is in AC$^0$ iff some alternating Turing machine accepts $R$ in time $O(\log n)$ with a constant number of alternations.

**$\Sigma^B_0$-Representation Theorem [Zambella ’96, Cook-Nguyen]**

$R(\vec{x}, \vec{X})$ is in AC$^0$ iff it is represented by a $\Sigma^B_0$-formula $\varphi(\vec{x}, \vec{X})$.

**Useful consequences**

1. Don’t need to work with uniform circuit families or alternating Turing machines when defining AC$^0$ functions or relations.
2. Useful when working with AC$^0$-reductions
The theory $V^0$ for $AC^0$ reasoning

The axioms of $V^0$

1. **2-BASIC axioms**: essentially the axioms of Robinson arithmetic plus
   - the defining axioms for $\leq$ and the string length function $| |$
   - the axiom of extensionality for finite sets (bit strings).

2. $\Sigma^B_0$-COMP (Comprehension): for every $\Sigma^B_0$-formula $\varphi(z)$ without $X$,
   \[ \exists X \leq y \forall z < y (X(z) \leftrightarrow \varphi(z)) \]

Theorem

1. $\Sigma^B_0$-IND: $[\varphi(0) \land \forall x (\varphi(x) \rightarrow \varphi(x + 1))] \rightarrow \forall x \varphi(x)$, where $\varphi \in \Sigma^B_0$.

2. The provably total functions in $V^0$ are precisely $\text{FAC}^0$.

Note: Theories, developed using Cook-Nguyen method, extend $V^0$. 
The theory \textit{VCC* for CC*}

Comparator Circuit Value (\textit{Ccv}) Problem (decision)

- Given a comparator circuit with specified Boolean inputs
- Determine the output value of a designated wire.

Recall that \textit{CC*} = \{decision problems AC^0 oracle-reducible to Ccv\}

The two-sorted theory \textit{VCC*} [using the Cook-Nguyen method]

- \textit{VCC*} has vocabulary $\mathcal{L}_A^2$
- Axiom of \textit{VCC*} = Axiom of $V^0$ + one additional axiom asserting the existence of a solution to the Ccv problem.
Asserting the existence of a solution to $C_{cv}$

- $X$ encodes a comparator circuit with $m$ wires and $n$ gates
- $Y$ encodes the input sequence
- $Z$ is an $(n + 1) \times m$ matrix, where column $i$ of $Z$ encodes values layer $i$

The following $\Sigma^B_0$ formula $\delta_{CCV}(m, n, X, Y, Z)$ states that $Z$ encodes the correct values of all the layers of the $C_{cv}$ instance encoded in $X$ and $Y$:

$$\forall k < m (Y(k) \leftrightarrow Z(0, k)) \land \forall i < n \forall x < m \forall y < m,$$

$$(X)^i = \langle x, y \rangle \rightarrow \begin{bmatrix}
Z(i + 1, x) \leftrightarrow (Z(i, x) \land Z(i, y)) \\
\land \quad Z(i + 1, y) \leftrightarrow (Z(i, x) \lor Z(i, y)) \\
\land \quad \forall j < m [(j \neq x \land j \neq y) \rightarrow (Z(i + 1, j) \leftrightarrow Z(i, j))]
\end{bmatrix}$$

$V_{CC}^* = V^0 + \exists Z \leq \langle m, n + 1 \rangle + 1, \delta_{CCV}(m, n, X, Y, Z)$
Conclusion

Summary

1. Introduce the new complexity classes CC and CC*, which are AC⁰-many-one-closure and AC⁰-oracle-closure of Ccv respectively.

\[
\text{NC}^1 \subseteq \text{NL} \subseteq \text{CC} \subseteq \text{CC}^{\text{Subr}} \subseteq \text{CC}^* \subseteq \text{P}
\]

2. Promote the use of \( \Sigma^B_0 \)-formulas when working with AC⁰ functions or relations.

3. Introduce the two-sorted theory VCC* that “captures” CC*. We show that

\[
\text{VNC}^1 \subseteq \text{VNL} \subseteq \text{VCC}^* \subseteq \text{VP}
\]

4. Sharpen and simplify Subramanian’s results: we show the following problems are CC-complete:
   - lex-first maximal matching (even with degree at most 3)
   - stable-marriage (man-opt, woman-opt and search version)
   - three-valued Ccv (useful when showing the completeness of stable marriage)

5. Prove the correctness of the above reductions within VCC*.
Conclusion

Summary

1. Introduce the new complexity classes $\text{CC}$ and $\text{CC}^*$, which are $\text{AC}^0$-many-one-closure and $\text{AC}^0$-oracle-closure of $\text{Ccv}$ respectively.

\[ \text{NC}^1 \subseteq \text{NL} \subseteq \text{CC} \subseteq \text{CC}^{\text{Subr}} \subseteq \text{CC}^* \subseteq \text{P} \]

2. Promote the use of $\Sigma_0^B$-formulas when working with $\text{AC}^0$ functions or relations.

3. Introduce the two-sorted theory $\text{VCC}^*$ that “captures” $\text{CC}^*$. We show that

\[ \text{VNC}^1 \subseteq \text{VNL} \subseteq \text{VCC}^* \subseteq \text{VP} \]

4. Sharpen and simplify Subramanian’s results: we show the following problems are $\text{CC}$-complete:
   - lex-first maximal matching (even with degree at most 3)
   - stable-marriage (man-opt, woman-opt and search version)
   - three-valued $\text{Ccv}$ (useful when showing the completeness of stable marriage)

5. Prove the correctness of the above reductions within $\text{VCC}^*$.

Open Problems

1. Is $\text{CC} = \text{CC}^{\text{Subr}} = \text{CC}^*$?
2. Do universal comparator circuits exists?
3. Is $\text{CC}/\text{CC}^{\text{Subr}}/\text{CC}^*$ equal to $\text{P}$?
4. Does any of the $\text{CC}$-complete problem have an $\text{NC}$ or $\text{RNC}$ algorithm?