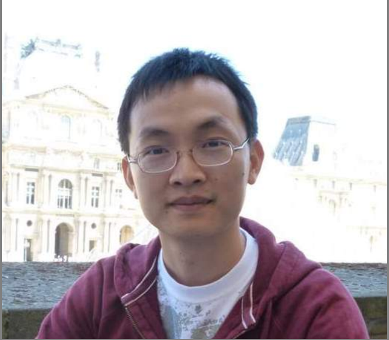
A Formal Theory for the Complexity Class Associated with the Stable Marriage Problem

Stephen Cook

Joint work with Dai Tri Man Lê and Yuli Ye

Department of Computer Science
University of Toronto
Canada

CSL 2011



Dai Lê

Bounded Reverse Mathematics [Cook-Nguyen '10]

Motivation

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

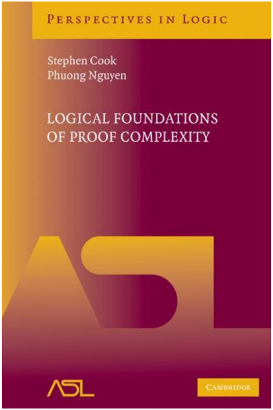
Program in Chapter 9

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

$$\mathsf{AC}^0 \subseteq \mathsf{C} \subseteq \mathsf{P}$$

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

$$\mathsf{VC} \vdash \tau$$



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

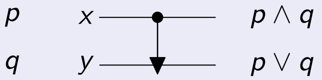
Outline of the talk

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

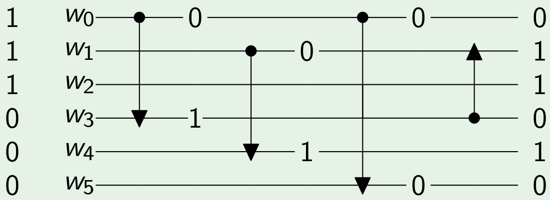
Comparator Circuits

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

Comparator gate

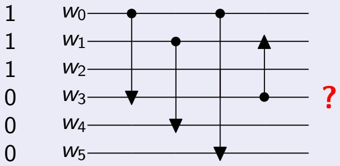


Example



Comparator Circuit Value ($\textsc{Ccv}$) Problem (decision)

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

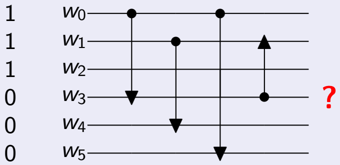


Complexity classes

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

Comparator Circuit Value ($\textsc{Ccv}$) Problem (decision)

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



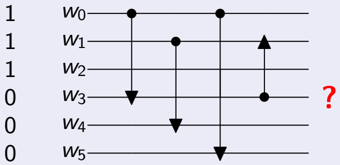
Complexity classes

1. $\mathsf{CC}^{\mathsf{Subr}} = \{$decision problems log-space many-one-reducible to $\textsc{Ccv}\}$
   - [Subramanian's PhD thesis '90], [Mayr-Subramanian '92]
2. $\mathsf{CC} = \{$decision problems AC^0 many-one-reducible to $\textsc{Ccv}\}$
   - Complete problems: stable marriage, lex-first maximal matching...
3. $\mathsf{CC}^* = \{$decision problems AC^0 oracle-reducible to $\textsc{Ccv}\}$
   - Needed when developing a Cook-Nguyen style theory for CC
   - The function class FCC^* is closed under compostion

$$\boxed{\mathsf{NC}^1 \subseteq \mathsf{NL} \subseteq \mathsf{CC} \subseteq \mathsf{CC}^{\mathsf{Subr}} \subseteq \mathsf{CC}^* \subseteq \mathsf{P}}$$

Comparator Circuit Value ($\textsc{Ccv}$) Problem (decision)

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



Complexity classes

1. $\mathsf{CC}^{\mathsf{Subr}} = \{$decision problems log-space many-one-reducible to $\textsc{Ccv}\}$
   - [Subramanian's PhD thesis '90], [Mayr-Subramanian '92]
2. $\mathsf{CC} = \{$decision problems AC^0 many-one-reducible to $\textsc{Ccv}\}$
   - Complete problems: stable marriage, lex-first maximal matching...
3. $\mathsf{CC}^* = \{$decision problems AC^0 oracle-reducible to $\textsc{Ccv}\}$
   - Needed when developing a Cook-Nguyen style theory for CC
   - The function class FCC^* is closed under compostion

$$\boxed{\mathsf{NC}^1 \subseteq \mathsf{NL} \subseteq \mathsf{CC} \subseteq \mathsf{CC}^{\mathsf{Subr}} \subseteq \mathsf{CC}^* \subseteq \mathsf{P}}$$

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

Men:

a	x	y
b	y	x

Women:

x	a	b
y	a	b

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

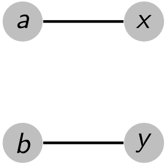
Preference lists

Men:

a	x	y
b	y	x

Women:

x	a	b
y	a	b



stable marriage

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

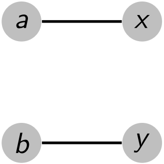
Preference lists

Men:

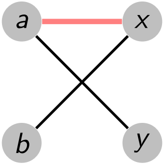
a	x	y
b	y	x

Women:

x	a	b
y	a	b



stable marriage



unstable marriage

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

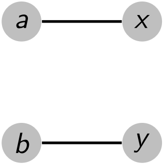
Preference lists

Men:

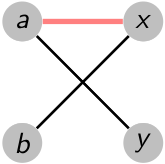
a	x	y
b	y	x

Women:

x	a	b
y	a	b



stable marriage



unstable marriage

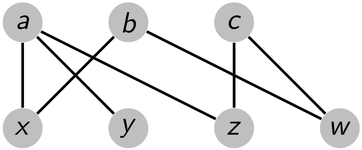
Stable Marriage Problem (decision version)

Is a given pair of (m, w) in the man-optimal (woman-optimal) stable marriage?

Lex-first maximal matching problem

Lex-first maximal matching

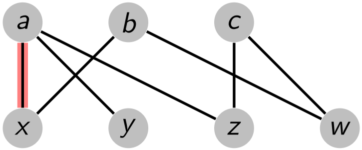
- Let G be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node



Lex-first maximal matching problem

Lex-first maximal matching

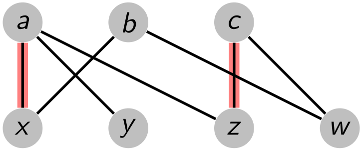
- Let G be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node



Lex-first maximal matching problem

Lex-first maximal matching

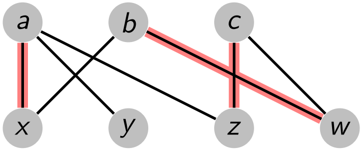
- Let G be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node



Lex-first maximal matching problem

Lex-first maximal matching

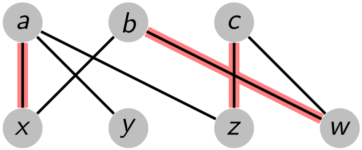
- Let G be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node



Lex-first maximal matching problem

Lex-first maximal matching

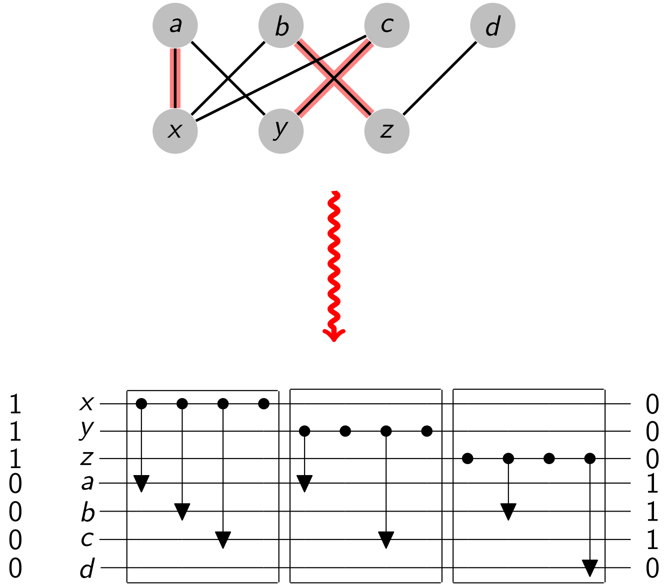
- Let G be a bipartite graph.
- Successively match the bottom nodes $x, y, z, \ldots$ to the least available top node



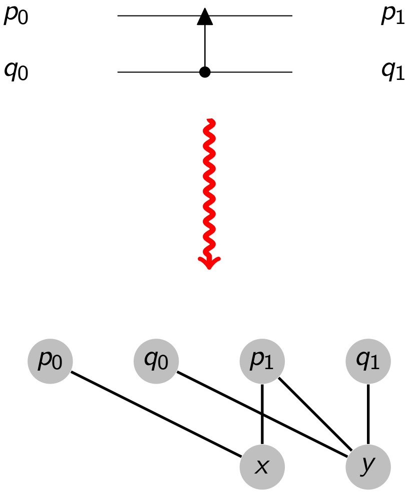
Lex-first maximal matching problem (decision)

Is a given edge $\{u, v\}$ in the lex-first maximal matching of G?

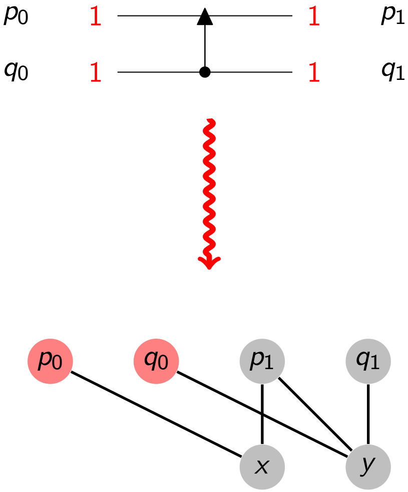
Reducing lex-first maximal matching to CCV



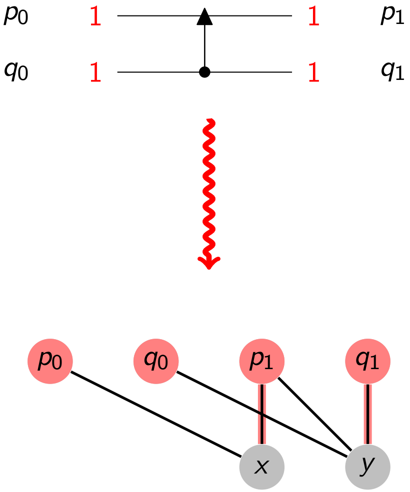
Reducing Ccv to lex-first maximal matching



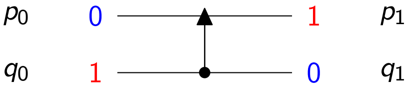
Reducing CCV to lex-first maximal matching

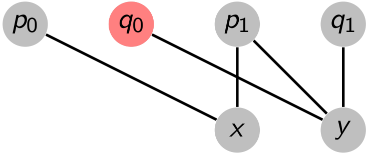


Reducing CCV to lex-first maximal matching

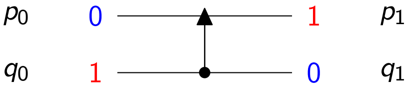


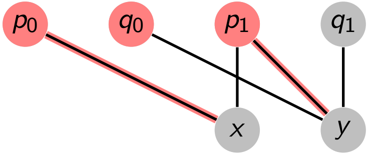
Reducing $\textsc{Ccv}$ to lex-first maximal matching





Reducing $\textsc{Ccv}$ to lex-first maximal matching





Outline of the talk

1. The complexity class CC
   - Interesting natural complete problems: stable marriage, lex-first maximal matching, comparator circuit value problem. . .
2. $\boxed{\text{Use the Cook-Nguyen method to define a theory for CC}}$
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 $\mathbb{N}$
- $|X|$ = 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 $\mathcal{L}_A^2 = [0, 1, +, \cdot, |\ |; \in, \leq, =_1, =_2]$

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

Note

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

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

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

Note

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

- "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 $X, Y, Z, \ldots$

Definition (Σ_0^B 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 $\mathcal{O}(\log n)$ with a constant number of alternations.

Σ_0^B-Representation Theorem [Zambella '96, Cook-Nguyen]

$R(\vec{x}, \vec{X})$ is in AC^0 iff it is represented by a Σ_0^B-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. Σ_0^B-**COMP** (Comprehension): for every Σ_0^B-formula $\varphi(z)$ without X,
$$\exists X \leq y \, \forall z < y \big(X(z) \leftrightarrow \varphi(z)\big)$$

Theorem

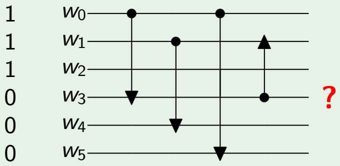
1. Σ_0^B-**IND:** $\big[\varphi(0) \wedge \forall x\big(\varphi(x) \rightarrow \varphi(x+1)\big)\big] \rightarrow \forall x \varphi(x)$, *where* $\varphi \in \Sigma_0^B$.
2. *The provably total functions in* V^0 *are precisely* FAC^0.

Note: Theories, developed using Cook-Nguyen method, extend V^0.

The theory VCC^* for CC^*

Comparator Circuit Value ($\textsc{Ccv}$) Problem (decision)

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

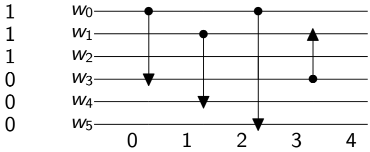


Recall that $\mathsf{CC}^* = \{$decision problems AC^0 oracle-reducible to $\textsc{Ccv}\}$

The two-sorted theory VCC^* [using the Cook-Nguyen method]

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

Asserting the existence of a solution to Ccv



- 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 Σ_0^B formula $\delta_{\mathsf{CCV}}(m, n, X, Y, Z)$ states that Z encodes the correct values of all the layers of the Ccv instance encoded in X and Y:

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

$$(X)^i = \langle x, y\rangle \to \left[\begin{array}{ll} & Z(i+1,x) \leftrightarrow \big(Z(i,x) \land Z(i,y)\big) \\ \land & Z(i+1,y) \leftrightarrow \big(Z(i,x) \lor Z(i,y)\big) \\ \land & \forall j < m\Big[(j \neq x \land j \neq y) \to \big(Z(i+1,j) \leftrightarrow Z(i,j)\big)\Big] \end{array}\right]$$

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

Conclusion

Summary

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

$$\boxed{\mathsf{NC}^1 \subseteq \mathsf{NL} \subseteq \mathsf{CC} \subseteq \mathsf{CC}^{\mathsf{Subr}} \subseteq \mathsf{CC}^* \subseteq \mathsf{P}}$$

Conclusion

Summary

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

$$\boxed{\text{NC}^1 \subseteq \text{NL} \subseteq \text{CC} \subseteq \text{CC}^{\text{Subr}} \subseteq \text{CC}^* \subseteq \text{P}}$$

2. Promote the use of Σ_0^B-formulas when working with AC^0 functions or relations.

Conclusion

Summary

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

$$\boxed{\mathrm{NC}^1 \subseteq \mathrm{NL} \subseteq \mathrm{CC} \subseteq \mathrm{CC}^{\mathrm{Subr}} \subseteq \mathrm{CC}^* \subseteq \mathrm{P}}$$

2. Promote the use of Σ_0^B-formulas when working with AC^0 functions or relations.
3. Introduce the two-sorted theory VCC^* that "captures" CC^*. We show that

$$\boxed{\mathrm{VNC}^1 \subseteq \mathrm{VNL} \subseteq \mathrm{VCC}^* \subseteq \mathrm{VP}}$$

Conclusion

Summary

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

$$\boxed{\mathsf{NC}^1 \subseteq \mathsf{NL} \subseteq \mathsf{CC} \subseteq \mathsf{CC}^{\mathsf{Subr}} \subseteq \mathsf{CC}^* \subseteq \mathsf{P}}$$

2. Promote the use of Σ_0^B-formulas when working with AC^0 functions or relations.
3. Introduce the two-sorted theory VCC^* that "captures" CC^*. We show that

$$\boxed{\mathsf{VNC}^1 \subseteq \mathsf{VNL} \subseteq \mathsf{VCC}^* \subseteq \mathsf{VP}}$$

4. Sharpen and simplify Subramanian's results: we show the following problems are CC-complete (under AC^0-many-one-reduction):
   - 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)

Conclusion

Summary

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

$$\boxed{\mathrm{NC}^1 \subseteq \mathrm{NL} \subseteq \mathrm{CC} \subseteq \mathrm{CC}^{\mathrm{Subr}} \subseteq \mathrm{CC}^* \subseteq \mathrm{P}}$$

2. Promote the use of Σ_0^B-formulas when working with AC^0 functions or relations.
3. Introduce the two-sorted theory VCC^* that "captures" CC^*. We show that

$$\boxed{\mathrm{VNC}^1 \subseteq \mathrm{VNL} \subseteq \mathrm{VCC}^* \subseteq \mathrm{VP}}$$

4. Sharpen and simplify Subramanian's results: we show the following problems are CC-complete (under AC^0-many-one-reduction):
   - 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 CC and CC^*, which are AC^0-many-one-closure and AC^0-oracle-closure of Ccv respectively.
$$\boxed{NC^1 \subseteq NL \subseteq CC \subseteq CC^{Subr} \subseteq CC^* \subseteq P}$$
2. Promote the use of Σ_0^B-formulas when working with AC^0 functions or relations.
3. Introduce the two-sorted theory VCC^* that "captures" CC^*. We show that
$$\boxed{VNC^1 \subseteq VNL \subseteq VCC^* \subseteq VP}$$
4. Sharpen and simplify Subramanian's results: we show the following problems are CC-complete (under AC^0-many-one-reduction):
    - 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^*.

Open Problems

1. $CC = CC^{Subr} = CC^*$?
2. Do universal comparator circuits exist?
3. $CC^* = P$?
4. Do the complete problems in CC have NC or RNC algorithms?