A formal framework for Stringology

Neerja Mhaskar and Michael Soltys

August 2015

Summary

- A new formal framework for Stringology is proposed, which consists of a three-sorted logical theory S designed to capture the combinatorial reasoning about finite words.
- A witnessing theorem is proven which demonstrates how to extract algorithms for constructing strings from their proofs of existence.
- Various other applications of the theory are shown.
- The long term goal of this line of research is to introduce the tools of Proof Complexity to the analysis of strings.

Language

Formal	Informal	Intended Meaning
		Index
0_{index}	0	the integer zero
1_{index}	1	the integer one
$+_{\text{index}}$	$+$	integer addition
$-_{\text{index}}$	$-$	bounded integer subtraction
$\cdot_{\text{index}}$	$\cdot$	integer multiplication (we also just use juxtaposition)
$\text{div}_{\text{index}}$	div	integer division
$\text{rem}_{\text{index}}$	rem	remainder of integer division
$<_{\text{index}}$	$<$	less-than for integers
$=_{\text{index}}$	$=$	equality for integers
		Alphabet symbol
$\mathbf{0}_{\text{symbol}}$	**0**	default symbol in every alphabet
σ_{symbol}	σ	unary function for generating more symbols
$<_{\text{symbol}}$	$<$	ordering of alphabet symbols
$\text{cond}_{\text{symbol}}$	cond	a conditional function
$=_{\text{symbol}}$	$=$	equality for alphabet symbols
		String
$\|\|_{\text{string}}$	$\|\|$	unary function for string length
e_{string}	e	binary fn. for extracting the i-th symbol from a string
$=_{\text{string}}$	$=$	string equality

λ string constructors

The string 000 can be represented by:

$$\lambda i\langle 1+1+1, \mathbf{0}\rangle.$$

Given an integer n, let $\hat{n}$ abbreviate the term $1+1+\cdots+1$ consisting of n many 1s. Using this convenient notation, a string of length 8 of alternating 1s and 0s can be represented by:

$$\lambda i\langle \hat{8}, \mathrm{cond}(\exists j \leq i(j+j=i), \mathbf{0}, \sigma\mathbf{0})\rangle.$$

Let U be a binary string, and suppose that we want to define $\bar{U}$, which is U with every 0 (denoted $\mathbf{0}$) flipped to 1 (denote $\sigma\mathbf{0}$), and every 1 flipped to 0. We can define $\bar{U}$ as follows:

$$\bar{U} := \lambda i\langle |U|, \mathrm{cond}(e(U,i)=\mathbf{0}, \sigma\mathbf{0}, \mathbf{0}\rangle.$$

Index Axioms

Index Axioms	
B1. $i+1 \neq 0$	B9. $i \leq j, j \leq i \rightarrow i = j$
B2. $i+1 = j+1 \rightarrow i = j$	B10. $i \leq i+j$
B3. $i+0 = i$	B11. $0 \leq i$
B4. $i+(j+1) = (i+j)+1$	B12. $i \leq j \vee j \leq i$
B5. $i \cdot 0 = 0$	B13. $i \leq j \leftrightarrow i < j+1$
B6. $i \cdot (j+1) = (i \cdot j)+i$	B14. $i \neq 0 \rightarrow \exists j \leq i(j+1 = i)$
B7. $i \leq j, i+k = j \rightarrow j-i = k$	B15. $i \not\leq j \rightarrow j-i = 0$
B8. $j \neq 0 \rightarrow \mathrm{rem}(i,j) < j$	B16. $j \neq 0 \rightarrow i = j \cdot \mathrm{div}(i,j) + \mathrm{rem}(i,j)$

Symbol and String Axioms

Alphabet Axioms
B17. $u \lneqq \sigma u$
B18. $u < v, v < w \rightarrow u < w$
B19. $\alpha \rightarrow \mathrm{cond}(\alpha, u, v) = u$
B20. $\neg\alpha \rightarrow \mathrm{cond}(\alpha, u, v) = v$

String Axioms
B21. $\lvert\lambda i\langle t, s\rangle\rvert = t$
B22. $j < t \rightarrow e(\lambda i\langle t, s\rangle, j) = s(j/i)$
B23. $\lvert U\rvert \leq j \rightarrow e(U, j) = \mathbf{0}$
B24. $\lvert U\rvert = \lvert V\rvert, \forall i < \lvert U\rvert e(U, i) = e(V, i) \rightarrow U = V$

Conclusion

- If we can prove $\forall X \exists Y \alpha(X, Y)$, then we can compute the Y in polynomial time. (*Witnessing Theorem*.) So we can extract algorithms from proofs!
- Utilize the sophisticated tools of Proof Complexity for a combinatorial analysis of strings.

  > *the richness of the field of Stringology arises from the fact that a string U is a map $I \longrightarrow \Sigma$, where I can be arbitrarily large, while Σ is "small." This produces repetitions and patterns that are the object of study for Stringology. On the other hand, Proof Complexity has studied in depth the varied versions of the Pigeonhole Principle that is responsible for these repetitions.*

Conclusion Cont'd

- The formalization allows us to see explicitly what is the engine of reasoning behind combinatorics on words.
  The Alphabet and String Axioms are *definitional*; they state the definitions of the objects.
  However, the Axioms for Indices provide the reasoning power.
  They show that combinatorics on words uses number theory on indices in order to prove its results.

Please visit:

http://soltys.cs.csuci.edu

and/or email us to discuss this further:

michael.soltys@csuci.edu

pophlin@mcmaster.ca