Symmetric Functions Capture General Functions

36th MFCS, 2011, EaGL Workshop 9/11/11

Richard J. Lipton\(^1\) Kenneth W. Regan\(^2\) Atri Rudra\(^3\)

Georgia Tech University at Buffalo (SUNY)

September 11, 2011

\(^1\)Research connected to blog “Gödel’s Lost Letter and P = NP”
\(^2\)Ditto
\(^3\)Supported by NSF CAREER grant CCF-0844796
Symmetric Functions Capture General Functions

Symmetric Functions Are...

**Hard:**
- Parity ∉ AC$^0$.
- Majority is complete for TC$^0$.

**Easy:**
- Over $x \in \{0, 1\}^n$, depend only on $\#1(x)$.
- ACC$^0 \subset \text{symm}(\text{quasi-poly many } \land)$ (Beigel-Tarui)
- The *elementary symmetric functions* are easy even in $\mathbb{Z}_m$ (Gromulsz).

**Main Theorems:** Senses in which every function $f$ is complexity-equivalent to some symmetric function $g$.

Why care? Symmetric functions have great algebraic structure.
Symmetric Functions Over Fields (And Rings \( R \))

- \( f : R^n \rightarrow R \) is symmetric if for all permutations \( \pi \) on \([n]\), \( f(\pi x) = f(x) \).
- Symmetric functions closed under +, *.
- Hence for any symmetric functions \( \sigma_1, \ldots, \sigma_n : R_1^N \rightarrow R_0 \) and polynomials \( f : R_0^n \rightarrow R \), the function \( f' : R_1^N \rightarrow R_0 \) is symmetric, where

\[
f'(y_1, \ldots, y_N) = f(\sigma_1(\bar{y}), \ldots, \sigma_n(\bar{y})).
\]

- Provided each \( \sigma_i(y_1, \ldots, y_N) \) is easy to compute, \( f' \leq f \).
- When does \( f \leq f' \)?
- Note: if \( F \) is a finite field then every function from \( F^n \) to \( F \) is a polynomial.
Fast Symmetrization

Goal: Compute $f(a_1, \ldots, a_n)$ over $R_0$.

Given: Can compute $f'(\tilde{b}) = f(\sigma_1(\tilde{b}), \ldots, \sigma_n(\tilde{b}))$ for any $b \in R_1^N$.

Task: Pick the $\sigma_i$ so that given any $\tilde{a} \in R_0^n$ one can efficiently find $\tilde{b} \in R_1^N$ such that

$$a_1 = \sigma_1(\tilde{b}), a_2 = \sigma_2(\tilde{b}), \ldots, a_n = \sigma_n(\tilde{b})$$

Then

$$f(\tilde{a}) = f'(\tilde{b}).$$

So $f \leq f'$. 
Coding Via Symmetric Functions

We want $\Sigma = (\sigma_1, \ldots, \sigma_n)$, so that $\Sigma : R_1^N \rightarrow R_0^n$, to be onto $R_0^n$ and efficiently invertible as well as computable.

Complexity considerations:

- Size of $R_1$ and $N$? Define $s = 1 + \log_{|R_0|}(|R_1|^N/|R_0|^n)$.
  - If $N = n$, and $R_0$ is a field $F$, then $R_1$ can be the field extension $F^s$.
- Degree $d'$ of $f'$ as a symmetric polynomial, vs. degree $d$ of $f$.
- Time $u(n)$ to invert $\Sigma$, i.e. to compute
  \[ \Sigma^{-1}(\bar{a}) = \bar{b}. \]
- Time $t(n)$ to compute $\Sigma$.

Two main constructions in paper give different tradeoffs.
1. Elementary Symmetrization

- The *elementary symmetric polynomials* $s_1, s_2, \ldots, s_n : \mathbb{R}^n \to \mathbb{R}$ are defined by

  $$s_i(b_1, \ldots, b_n) = \sum_{J \subseteq [n], |J|=i} \prod_{j \in J} b_j.$$

  So $s_1(\vec{b}) = b_1 + \cdots + b_n$,
  $s_2(\vec{b}) = b_1 b_2 + \cdots + b_1 b_n + \cdots b_2 b_3 + \cdots + b_{n-1} b_n$, and
  $s_n = b_1 b_2 \cdots b_n$.

- Form an algebra basis for all symmetric polynomials on $\mathbb{R}^n$.
- Idea is to define the following, which gives degree $d' = dn$:

  $$f'(b_1, \ldots, b_n) = f(s_1(\vec{b}), \ldots, s_n(\vec{b})).$$

- By counting, *cannot* have $|R_1| = |R_0| = q$, so $s > 1$. Theorem: $s \geq \lceil \log_2 n \rceil - 3$. 

Symmetric Functions Capture General Functions
Simple Example

The $2 \times 2$ permanent polynomial $ad + bc$ undergoes the substitutions

\[
\begin{align*}
    a & \mapsto e + f + g + h \\
    b & \mapsto ef + eg + eh + fg + fh + gh \\
    c & \mapsto efg + efh + egh + fgh \\
    d & \mapsto efgh
\end{align*}
\]

to yield

\[
\begin{align*}
e^2f^2g + e^2fg^2 + ef^2g^2 + e^2f^2h + e^2g^2h + f^2g^2h \\
+ e^2fh^2 + ef^2h^2 + e^2gh^2 + f^2gh^2 + eg^2h^2 + fg^2h^2 \\
+ 4e^2fgh + 4ef^2gh + 4efg^2h + 4efgh^2
\end{align*}
\]
Symmetric Functions Capture General Functions

Elementary Facts

For a formal single variable $x$,

\[
\prod_{i=1}^{n}(x + b_i) = x^n + \sum_{i=1}^{n} s_i(b_1, \ldots, b_n)x^{i-1}. \tag{1}
\]

**Fact:** All $s_i(\vec{b})$ are computed in $O(n(\log n)^2)$ time by using FFT to multiply out the product on the left-hand side of (1).

For inversion, given $(a_1, \ldots, a_n)$, we want $\vec{b} = (b_1, \ldots, b_n)$ such that for each $i$, $a_i = s_i(\vec{b})$. Define

\[
\phi = \phi_{\vec{a}}(x) = x^n + \sum_{i=1}^{n} a_i x^{i-1}.
\]

By fact (1), our goal is to split $\phi$ into linear factors:

\[
\phi = \prod_{i}(x + b_i).
\]

This will make $a_i = s_i(\vec{b})$ for each $i$. 
The problem is that $\phi$ may not—indeed by the counting, generally will not—split into linear factors over $R_0$. We need $R_0$ to be a field $F$, and $R_1$ to be an extension $F^s$. How large must $s$ be?

**Lemma (well-known)**

The minimum $s$ equals the least common multiple of the degrees of all irreducible factors of $\phi$ over $F$.

Alas, this $s$ can be as high as $n^{O(\sqrt{n})}$, making the extension field elements themselves have exponential size.

**Theorem (also known)**

$$\Pr_{\tilde{a} \in F^n} [\log s > \log^2 n] < 2^{-\Omega(\sqrt{\log n})}.$$  

Thus there are exp-few bad $\tilde{a}$ that make $s$ larger than $n^{O(\log n)}$. 


Theorem (paper has more-general form)

If the symmetric function $f$ is in time $v(n)$, then $f \in \text{RTIME}[dv(n) + n^{O(\log n)}q^{O(1)}]$. □
2. Second Symmetrization

- Can we do better than quasi-polynomial time overhead?
- Answer is yes, but degree of $f'$ becomes higher: $d' = q^2 dn \log_q n$.
- Still needs an extension field, but $s \leq 1 + \lceil \log_q n \rceil$.
- Less algebraically simple to define, but running time basically cannot be beat:

**Theorem**

Every function $f : F_q^n \rightarrow F_q$ is equivalent to a symmetric function $f' : F_{q^s}^n \rightarrow F_q$ with above parameters, up to $\tilde{O}(n)$ deterministic time complexity (plus $\text{poly}(q, s)$ pre-processing to represent $F_{q^s}$).

Note that $f'$ maps from the extension field into the original field.
Idea: How to encode information symmetrically?

Recall the task is to pick symmetric $\sigma_i$ so that given any $\tilde{a} \in R_0^n$ one can efficiently find $\tilde{b} \in R_1^N$ such that

$$a_1 = \sigma_1(\tilde{b}), \ a_2 = \sigma_2(\tilde{b}), \ldots, \ a_n = \sigma_n(\tilde{b})$$

so that

$$f'(b_1, \ldots, b_n) = f(\sigma_1(\tilde{b}), \ldots, \sigma_n(\tilde{b})).$$

Idea is to encode $b_i = \langle i, a_i \rangle$. In general we have pairs $\langle j, a \rangle$. How do we know which index $j$ gives us $a_i$? We need to create a Kronecker delta function $\delta_i(j)$. Then each $a_i$ can be represented symmetrically as a sum

$$a_i = \sum_{j=1}^{n} \delta_i(j) a_j.$$ 

Over finite fields, all this can be done with polynomials.
Proof of Second Main Theorem

- Pre-process to represent $F_{q^s}$ by an irreducible polynomial with formal root $\gamma$, giving every element $\alpha$ of the extension field as

$$\alpha = \sum_{\ell=s-1}^{0} \alpha_\ell \gamma^\ell = (\alpha_{s-1}, \ldots, \alpha_0).$$

- By choice of $s$, $n \leq q^{s-1}$, so embed $[n]$ into first $s - 1$ places.
- Next construct polynomials $\pi_k$ that project out the $k$-th place:

$$\pi_k(\alpha) = \alpha_k.$$ 

- To do so, define $V$ to be the Vandermonde matrix whose row $\ell$, $0 \leq \ell \leq s - 1$, comprises the first $s$ powers of $\gamma^{q^\ell}$. Then using column vectors,

$$V(\alpha_{s-1}, \ldots, \alpha_0) = (\alpha^{q^{s-1}}, \ldots, \alpha^{q^2}, \alpha^q, \alpha),$$

so $\alpha_k$ is obtained by invering $V$ and dotting its $k$-th row with the right-hand side. Use polynomial closed-form for $V^{-1}$ to get $\pi_k$. 
Key Coding Lemma

Abbreviate $F_{q^s}$ to $E$ and $F_q$ to $F$, and let $\alpha_-$ stand for $\alpha$ minus its $\alpha_0$ co-ordinate, which may be an embedded value in $[n]$.

Lemma

For each $j \in [n]$ we can construct a symmetric polynomial $\phi_j : E^n \rightarrow F$ of degree at most $sq^s$ such that for any elements $\alpha^1, \ldots, \alpha^n$ in $E^n$,

$$
\phi_j(\alpha^1, \ldots, \alpha^n) = \sum_{i \in [n] : \alpha_=-j} \alpha^i.
$$

The proof picks apart $j$ into the $s - 1$ co-ordinates $(j_{s-1}, \ldots, j_1)$ of its embedded value in $F^{s-1}$. First idea is to represent the Kronecker delta function on the embedded values, namely $\delta_j(i) = 1$ if $i = j$ and 0 otherwise.
This formula makes $\delta_j(j) = 1$ since the fractions are identically 1:

$$\delta_j(u_{s-1}, \ldots, u_1) = \prod_{\ell=1}^{s-1} \prod_{\beta \in F \setminus \{j_\ell\}} \frac{u_\ell - \beta}{j_\ell - \beta}.$$ 

And $\delta_j(i) = 0$ for $i \neq j$ because the numerator hits a zero. Now define:

$$\phi_j(z_1, \ldots, z_n) = \sum_{i=1}^{n} \delta_j(\pi_{s-1}(z_i), \ldots, \pi_1(z_i)) \cdot \pi_0(z_i).$$

This picks out only those $\alpha^i_0$ for which the first $s - 1$ co-ordinates yield $j$, thus proving the lemma’s equation. Moreover $\phi_j$ is symmetric, thus proving the lemma.
Completing the Construction

Finally we define \( f' : E^n \rightarrow F \) by

\[
f'(\vec{b}) = f(\phi_1(\vec{b}), \phi_2(\vec{b}), \ldots, \phi_n(\vec{b})).
\]

Since each \( \phi_j \) has degree at most \( sq^s \), and \( s \) is chosen to make \( q^{s-1} \leq nq \), \( f' \) has degree at most \( sndq^2 \).

To compute \( f' \) from \( f \), one linear scan of \( \vec{b} \) can identify all the terms that will contribute to the sums in the Lemma, giving the arguments of \( f \).

To compute \( f(\vec{a}) \) with arguments from the base field \( F' \), we need to find \( \vec{b} \) over the extension field such that \( \phi_j(\vec{b}) = a_j \), and find it efficiently. This is done by using the embedded natural numbers, which pick out indices, as co-ordinates:

\[
b_i = (i_{s-1}, \ldots, i_1, a_i).
\]

Then for all \( j \), \( \phi_j(\vec{b}) = \pi_0(b_j) = a_j \), as needed. This is done in \( O(sn) \) time treating entries as units, which gives \( \tilde{O}(n) \) time overall. \( \square \)
The elementary symmetrization works over any field.
The second one does not, because the coding tricks require finite fields.
Different coding tricks work over the reals or complex numbers, but do not yield polynomials.
Paper gives a result over the reals.
Open Questions

- Can we prove that no symmetrization by polynomials over an infinite field gives $\tilde{O}(n)$ time?
- Can the possibility $N > n$ be used to improve either symmetrization?
- Can either symmetrization be used in a positive way to enable more-structured analysis of, say, symmetrized permanent polynomials?
- Can the idea be used to derive more (conditional) lower bounds?
- Are fields needed? What can be done over the rings $\mathbb{Z}_m$ for $m$ composite?