COSC 544 Probabilistic Proof Systems	9/14/17
The Sum-Check Protocol	
Lecturer: Justin Thaler	

1 The Sum-Check Protocol

Suppose we are given a v-variate polynomial g defined over a finite field $\mathbb{F}$. The purpose of the sum-check protocol [LFKN92] is to compute the sum:

$$H := \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_v \in \{0,1\}} g(b_1, \dots, b_v).$$

In applications, this sum will often be over a very large number of terms, so the verifier may not have the resources to compute the sum without help. Instead, she uses the sum-check protocol to force the prover to compute the sum for her.

Remark 1. In full generality, the sum-check protocol can compute the sum $\sum_{\mathbf{b} \in B^m} g(\mathbf{b})$ for any $B \subseteq \mathbb{F}$, but most of the applications covered in this survey will only require $B = \{0,1\}$.

For presentation purposes, we assume here that the verifier has oracle access to g, i.e., $\mathcal{V}$ can evaluate $g(r_1, \dots, r_v)$ for a randomly chosen vector $(r_1, \dots, r_v) \in \mathbb{F}^v$ with a single query to an oracle, though this will not be the case in applications. In our applications, $\mathcal{V}$ will either be able to efficiently evaluate $g(r_1, \dots, r_v)$ unaided, or if this is not the case, $\mathcal{V}$ will ask the prover to *tell her* $g(r_1, \dots, r_v)$, and $\mathcal{P}$ will subsequently prove this claim is correct via further applications of the sum-check protocol.

The protocol proceeds in v rounds. In the first round, the prover sends a polynomial $g_1(X_1)$, and claims that $g_1(X_1) = \sum_{(x_2, \dots, x_v) \in \{0,1\}^{v-1}} g(X_1, x_2, \dots, x_v)$. If g_1 is as claimed, then $H = g_1(0) + g_1(1)$.

Throughout, let $\deg_i(p)$ denote the degree of variable i in variable p. The polynomial $g_1(X_1)$ has degree $\deg_1(g)$. Hence g_1 can be specified with $\deg_1(g) + 1$ field elements, for example by sending the evaluation of g_1 at each point in the set $\{0, 1, \dots, \deg_1(g)\}$.

Then, in round $j > 1$, $\mathcal{V}$ chooses a value r_{j-1} uniformly at random from $\mathbb{F}$ and sends r_{j-1} to $\mathcal{P}$. We refer to this step by saying that variable $j-1$ gets *bound* to value r_{j-1}. In return, the prover sends a polynomial $g_j(X_j)$, and claims that

$$g_j(X_j) = \sum_{(x_{j+1}, \dots, x_v) \in \{0,1\}^{v-j}} g(r_1, \dots, r_{j-1}, X_j, x_{j+1}, \dots, x_v). \tag{1}$$

The verifier compares the two most recent polynomials by checking $g_{j-1}(r_{j-1}) = g_j(0) + g_j(1)$, and rejecting otherwise. The verifier also rejects if the degree of g_j is too high: each g_j should have degree $\deg_j(g)$, the degree of variable x_j in g.

In the final round, the prover has sent $g_v(X_v)$ which is claimed to be $g(r_1, \dots, r_{v-1}, X_v)$. $\mathcal{V}$ now checks that $g_v(r_v) = g(r_1, \dots, r_v)$ (recall that we assumed $\mathcal{V}$ has oracle access to g). If this test succeeds, and so do all previous tests, then the verifier is convinced that $H = g_1(0) + g_1(1)$.

The protocol is summarized below.

Description of Sum-Check Protocol.

- Fix an $H \in \mathbb{F}$.
- In the first round, $\mathcal{P}$ sends the univariate polynomial

$$g_1(X_1) := \sum_{(x_2,\dots,x_v)\in\{0,1\}^{v-1}} g(X_1, x_2, \dots, x_v).$$

  $\mathcal{V}$ checks that g_1 is a univariate polynomial of degree at most $\deg_1(g)$, and that $H = g_1(0) + g_1(1)$, rejecting if not.
- $\mathcal{V}$ chooses a random element $r_1 \in \mathbb{F}$, and sends r_1 to $\mathcal{P}$.
- In the jth round, for $1 < j < v$, $\mathcal{P}$ sends to $\mathcal{V}$ the univariate polynomial

$$g_j(X_j) = \sum_{(x_{j+1},\dots,x_v)\in\{0,1\}^{v-j}} g(r_1, \dots, r_{j-1}, X_j, x_{j+1}, \dots, x_v).$$

  $\mathcal{V}$ checks that g_j is a univariate polynomial of degree at most $\deg_j(g)$, and that $g_{j-1}(r_{j-1}) = g_j(0) + g_j(1)$, rejecting if not.
- $\mathcal{V}$ chooses a random element $r_j \in \mathbb{F}$, and sends r_j to $\mathcal{P}$.
- In Round v, $\mathcal{P}$ sends the univariate polynomial

$$g_v(X_v) = g(r_1, \dots, r_{v-1}, X_v)$$

  to $\mathcal{V}$. $\mathcal{V}$ checks that g_v is a univariate polynomial of degree at most $\deg_v(g)$, rejecting if not, and also checks that $g_{v-1}(r_{v-1}) = g_v(0) + g_v(1)$.
- $\mathcal{V}$ chooses a random element $r_v \in \mathbb{F}$ and evaluates $g(r_1, \dots, r_v)$ with a single oracle query to g. $\mathcal{V}$ checks that $g_v(r_v) = g(r_1, \dots, r_v)$, rejecting if not.
- If $\mathcal{V}$ has not yet rejected, $\mathcal{V}$ halts and accepts.

The following proposition formalizes the completeness and soundness properties of the sum-check protocol.

Proposition 1.1. *Let g be a v-variate polynomial of total degree at most d in each variable, defined over a finite field $\mathbb{F}$. For any $H \in \mathbb{F}$, let $\mathcal{L}$ be the language of of polynomials g (given as an oracle) such that*

$$H = \sum_{b_1\in\{0,1\}} \sum_{b_2\in\{0,1\}} \cdots \sum_{b_v\in\{0,1\}} g(b_1, \dots, b_v).$$

The sum-check protocol is an interactive proof system for L with completeness error $\delta_c = 0$ and soundness error $\delta_s \leq vd/|\mathbb{F}|$.

Proof. Completeness is evident: if the prover sends the prescribed polynomial $g_j(X_j)$ at all rounds j, then $\mathcal{V}$ will accept with probability 1.

The proof of soundness is by induction on v. In the case $v = 1$, $\mathcal{P}$'s only message specifies a degree d univariate polynomial $g_1(X_1)$. If $g_1(X_1) \neq g(X_1)$, then because any two distinct degree d univariate polynomials can agree at most d inputs, $g_1(r_1) \neq g(r_1)$ with probability at least $1 - d/|\mathbb{F}|$ over the choice of r_1, and hence $\mathcal{V}$'s final check will cause $\mathcal{V}$ to reject with probably at least $1 - d/|\mathbb{F}|$.

Assume by way of induction that for all $v-1$-variate polynomials, the sum-check protocol has soundness error at most $(v-1)d/|\mathbb{F}|$. Let $h_1(X_1) = \sum_{x_2,\dots,x_v\in\{0,1\}^{v-1}} g(X_1, x_2, \dots, x_v)$. Suppose $\mathcal{P}$ sends a polynomial

Communication	Rounds	$\mathcal{V}$ time	$\mathcal{P}$ time
$O\left(\sum_{i=1}^{v} \deg_i(g)\|\right)$ field elements	v	at most $O\left(\sum_{i=1}^{v} \deg_i(g)\right) + T$	at most $O\left(2^v \cdot T\right)$

Table 1: Costs of sum-check protocol when applied to a v-variate polynomial g over $\mathbb{F}$. $\deg_i(g)$ denotes the degree of variable i in g, and T denotes the cost of an oracle query to g.

$g_1(X_1) \neq h_1(X_1)$ in Round 1. Then because any two distinct degree d univariate polynomials can agree at most d inputs, $h_1(r_1) \neq g_1(r_1)$ with probability at least $1 - d/|\mathbb{F}|$. Conditioned on this event, $\mathcal{P}$ is left to prove the false claim in Round 2 that $g_1(r_1) = \sum_{(x_2,\ldots,x_v)\in\{0,1\}^{v-1}} g(r_1, x_2, \ldots, x_v)$. Since $g(r_1, x_2, \ldots, x_v)$ is a $v-1$-variate polynomial of total degree d, the inductive hypothesis implies that $\mathcal{V}$ will reject at some subsequent round of the protocol with probability at least $1 - d(v-1)/|\mathbb{F}|$. Therefore, $\mathcal{V}$ will reject with probability at least

$$1 - \Pr[h_1(r_1) \neq g_1(r_1)] - (1 - \Pr[\mathcal{V} \text{ rejects in some Round } j > 1 | h_1(r_1) \neq g_1(r_1)])$$
$$\geq 1 - \frac{d}{|\mathbb{F}|} - \frac{d(v-1)}{|\mathbb{F}|} = 1 - \frac{dv}{|\mathbb{F}|}.$$

□

Discussion of costs. There is one round in the sum-check protocol for each of the v variables of g. The total communication is $\sum_{i=1}^{v} \deg_i(g) + 1 = v + \sum_{i=1}^{v} \deg_i(g)$ field elements. In particular, if $\deg_i(g) = O(1)$ for all j, then the communication cost is $O(v)$ field elements.

The running time of the verifier over the entire execution of the protocol is proportional to the total communication, plus the cost of a single oracle query to g to compute $g(r_1, \ldots, r_v)$.

Determining the running time of the prover is less straightforward. Recall that $\mathcal{P}$ can specify g_j by sending for each $i \in \{0, \ldots, \deg_j(g)\}$ the value:

$$g_j(i) = \sum_{(x_{j+1},\ldots,x_v)\in\{0,1\}^{v-j}} g(r_1, \ldots, r_{j-1}, i, x_{j+1}, \ldots, x_v). \tag{2}$$

An important insight is that the number of terms defining the value $g_j(i)$ in Equation (2) falls geometrically with j: in the jth sum, there are only 2^{v-j} terms, each corresponding to a Boolean vector in $\{0,1\}^{v-j}$. Thus, the total number of terms that must be evaluated over the course of the protocol is $\sum_{j=1}^{v} \deg_j(g) 2^{v-j} = O(2^v)$ if $\deg_j(g) = O(1)$ for all j. Consequently, if $\mathcal{P}$ is given oracle access to g, then $\mathcal{P}$ will require just $O(2^v)$ time.

In all of the applications covered in this survey, $\mathcal{P}$ will not have oracle access to the truth table of g, and the key to many of the results in this survey is to show that $\mathcal{P}$ can nonetheless evaluate g at all of the necessary points in close to $O(2^v)$ total time.

The costs of the sum-check protocol are summarized in Table 1. Since $\mathcal{P}$ and $\mathcal{V}$ will not be given oracle access to g in applications, the table makes the number of oracle queries to g explicit.

Preview: Why multilinear extensions are useful. We will see several scenarios where it is useful to compute $H = \sum_{\mathbf{x}\in\{0,1\}^v} f(\mathbf{x})$ for some function $f\colon \{0,1\}^v \to \mathbb{F}$ derived from the verifier's input. We can compute H by applying the sum-check protocol to any low-degree extension g of f. When $g = \widetilde{f}$, or is derived from $\widetilde{f}$ in some way, then Lemma 1.6 from the previous lecture (which gave an explicit expression for $\widetilde{f}$ in terms of Lagrange basis polynomials) can often be exploited to ensure that enormous cancellations occur in the computation of the prover's messages, allowing fast computation. $\square$

Preview: Why using multilinear extensions is not always possible. Although the use of the MLE $\widetilde{f}$ typically ensures fast computation for the prover, $\widetilde{f}$ cannot be used in all applications. The reason is that the verifier has to be able to evaluate $\widetilde{f}$ at a random point $\mathbf{r} \in \mathbb{F}^v$ to perform the final check in the sum-check protocol, and in some settings, this computation would be too costly.

Lemma 1.8 from the previous lecture gives a way for $\mathcal{V}$ to evaluate $\tilde{f}(\mathbf{r})$ in time $\tilde{O}(2^v)$, given all evaluations of f at Boolean inputs. This might or might not be an acceptable runtime, depending on the relationship between v and the verifier's input size n. If $v = \log n + \text{poly}(\log\log n)$, then $\tilde{O}(2^v) = \tilde{O}(n)$, and the verifier runs in quasilinear time. But we will see some applications where $v = c \log n$ for some constant $c > 1$, and others where $v = n$ (cf. the #SAT protocol in the next lecture). In these settings, $\tilde{O}(2^v)$ runtime for the verifier is unacceptable, and we will be forced to use an extension g of f that has a succinct representation, enabling $\mathcal{V}$ to compute $g(\mathbf{r})$ in $o(2^v)$ time. Sometimes $\widetilde{f}$ itself has such a succinct representation, but other times we will be forced to use a higher-degree extension of f. $\square$

References

[LFKN92] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39:859–868, October 1992.