Randomness and Computation
or, “Randomized Algorithms”

Mary Cryan

School of Informatics
University of Edinburgh
Tuesday’s lecture: Verification of polynomial identities

On Tuesday we considered the problem of taking two polynomials of degree $d$, $F(x)$ written as a product of “monomials” and $G(x)$ as an expansion of $x^i$ terms, and testing whether $F(x)$ is identical to $G(x)$.

The basic algorithm takes a single uniform random sample $x_1$ from the set $\{1, \ldots, 100d\}$ and calculates whether $F(x_1)$ and $G(x_1)$ are equal. This testing algorithm gives a correct answer with probability at least $\frac{1}{100}$ (“one-sided” error).

▶ The sample drawn to perform the test is just a single value chosen uniformly from $\{1, \ldots, 100d\}$… easy probability distribution to understand.

▶ To refine the algorithm, we can do $k$ trials and “power up” the error to just $\frac{1}{100^k}$.
Matrix multiplication verification

We are given three $n \times n$ matrices $A$, $B$, $C$, and we are asked to verify whether

$$AB \equiv C,$$

without carrying out the tiresome task of multiplying out $AB$.

Recall that the “high-school/Uni” algorithm for evaluating $AB$ would take $\Theta(n^3)$ time, and the algorithm with the best asymptotic bound is still $\Theta(n^{2.373})$ or so.

We will show how to verify (with high probability) in $\Theta(n^2)$ time.
Matrix multiplication verification

Assume that the values in the matrix are integers over some field like \( \mathbb{F}_2 \) (also known as \( GF(2) \)), or indeed any \( \mathbb{F}_p \) for prime \( p > 2 \), or even the standard field over \( \mathbb{Z} \).

The algorithm is parametrized by some natural number \( k > 1 \).

**Algorithm** \( \text{MMVERIFY}(n, A, B, C) \)

1. **for** \( j = 1, \ldots, k \) **do**
2. Generate \( \bar{x} \) uniformly at random from \( \{0, 1\}^n \)
3. Calculate \( \bar{y}_B = B \cdot \bar{x} \) in \( \Theta(n^2) \) time.
4. Calculate \( \bar{y}_{AB} = A \cdot \bar{y}_B \) in \( \Theta(n^2) \) time.
5. Calculate \( \bar{y}_C = C \cdot \bar{x} \) in \( \Theta(n^2) \) time.
6. **if** \( \bar{y}_{AB} \neq \bar{y}_C \)
7. **return** “no”
8. **return** “yes”
Analysing MMVerify

We will show on the Overhead that each of steps 3., 4., 5. can be done in $\Theta(n^2)$ for a specific vector $x$ of length $n$. Now for the analysis, we will show . . .

“One-sided error”

$AB \equiv C$: In this case, we know that $AB \cdot \bar{x} = C\bar{x}$ for every $\bar{x} \in \{0, 1\}^n$. Hence MMVERIFY is guaranteed to return the value “yes”.

$AB \not\equiv C$: We will now show that in this case, that when a vector $\bar{x}$ is drawn u.a.r. from $\{0, 1\}^n$, the probability that $AB \cdot \bar{x} = C \cdot \bar{x}$ is at most $1/2$.

After this analysis, we will calculate the effect of doing $k$ trials.
Consider the two $n \times n$ matrices $AB$ and $C$. They are non-identical, so there must be at least one cell $(i^*, j^*)$ such that the values $(AB)_{i^*j^*}$ and $C_{i^*j^*}$ are different.

Let $D = (AB - C)$. Then equivalently, we have $D_{i^*j^*} \neq 0$.

Consider row $i^*$ of $D$, and consider its product with a vector $\bar{x} \in \{0, 1\}^n$:

$$\sum_{j=1}^{n} D_{i^*j} \cdot x_j.$$ 

This gives the value for position $i^*$ in the length-$n$ vector computed by $D \cdot \bar{x}$.

We will show that this will be 0 with probability at most $1/2$. 

RC (2018/19) – Lecture 2 – slide 6
Analysing MMVerify: $AB \not\equiv C$

When drawing a random $\bar{x} \in \{0, 1\}^n$ uniformly at random (u.a.r.), each $\bar{x}$ has equal probability $(1/2^n)$.

This is equivalent to choosing the value $x_i \in \{0, 1\}$ independently with probability $1/2$, for each $i \in [n] = \{1, \ldots, n\}$.

Use this in the analysis (*principle of deferred decisions*).
Write $\sum_{j=1}^{n} D_{i^*j} \cdot x_j$ as

$$
\left( \sum_{j \in [n] \setminus \{j^*\}} D_{i^*j} \cdot x_j \right) + D_{i^*j^*} \cdot x_{j^*}
$$

Think about sampling $\bar{x}$ (*deferred decisions*) as $\{0, 1\}^{n-1}$ vector first, followed by the value for $x_{j^*}$ last.
Analysing MMVerify: $AB \not\equiv C$

After sampling the $\{0, 1\}^{n-1}$ vector for positions $\{x_j \mid j \in [n] \setminus j^*\}$, we now have a fixed value for

$$\sum_{j \in [n]\setminus\{j^*\}} D_{i^*j} \cdot x_j.$$

Then no matter which field we are in ($\mathbb{Z}$ with standard arithmetic, $\mathbb{F}_p$ for some prime $p > 2$, even $\mathbb{F}_2 \ldots$) there is at most one value which could be added to this to get 0 (maybe 0, maybe 1, maybe some other non-zero value).

Also, we know $D_{i^*j^*} \neq 0$. Sampling $x_{j^*}$ last, we will get $D_{i^*j^*} \cdot x_{j^*} = D_{i^*j^*}$ (which is non-zero) with prob. $1/2$, and $D_{i^*j^*} \cdot x_{j^*} = 0$ with prob. $1/2$. Hence

$$\Pr \left[ \sum_{j=1}^{n} D_{i^*j} \cdot x_j = 0 \right] \leq 1/2$$
All trials of MMVerify: \( AB \neq C \)

Previous slides present the analysis of what happens (\( AB \neq C \) case) on a single sample from \( \{0, 1\}^n \) (tested in lines 2.-7. of Algorithm MMVERIFY).

▶ The Algorithm is set up to return “no” (and terminate) on the first trial where it discovers a mismatch between \( AB \cdot \bar{x} \) and \( C \cdot \bar{x} \).

▶ It only returns “yes” if it passed through all iterations of the loop with all trials giving a match.

▶ “Every trial gives a match” is the bad event for analysing the \( AB \neq \) case.
All trials of MMVerify: $AB \not\equiv C$

Notice that the $k$ repeated trials fit into the paradigm of “sampling with replacement”.

Let $E_j$ be the event that the $j$-th sampled $\bar{x}$ satisfies $D \cdot \bar{x} = 0$ (ie $AB \cdot \bar{x} = C \cdot \bar{x}$).

$E_1, \ldots, E_k$ are all pairwise independent. Thus, applying Defn 1.3 from lecture 1 repeatedly,

$$\Pr[\cap_{j=1}^{k} E_j] = \prod_{j=1}^{k} \Pr[E_j].$$

We have already shown that $\Pr[E_j] \leq 1/2$.

Hence $\Pr[\cap_{j=1}^{k} E_j]$, the probability that the algorithm returns “yes” is at most $1/2^k$ (in the case of $AB \not\equiv C$).

(note: I don’t like/approve-of the stuff with Bayes in the book)
Continue reading Chapter 1 of “Probability and Computing”.