Lecture 9

Recap

- Last lecture we finished describing the GKR protocol.
  - This interactive proof forces the prover to evaluate a layered arithmetic circuit gate-by-gate.
  - The verifier's runtime is $O(n + D \log S)$, where n is input size, D is circuit depth, and S is circuit size.
  - The prover can be implemented in time $O(S)$, though we did not cover details of how to achieve this.
- **Any** computation can be represented as an arithmetic circuit, but the circuit may be deep and/or large, and hence applying the GKR protocol to that circuit may not save the verifier time compared to just solving the problem with no prover.

Today's topics

- The Fiat-Shamir transformation: turning any public-coin interactive proof into a publicly-verifiable non-interactive argument.
- Front-ends: turning computer programs into circuits.
  - Key points to understand:
    1. **Any** algorithm running in time T can be turned into an arithmetic circuit of size not **too** much bigger than T (at most O(T^2 * polylog(T))).
       - But T^2 size is impractical, and the circuit may be deep.
    2. Fast **parallel** algorithms turn into small-depth circuits (parallel runtime $\approx$ circuit depth)
    3. Some algorithms running in time T naturally turn into small-depth circuits of size O(T) (e.g., naïve matrix multiplication)
    4. **Any** algorithm running time time T can be turned into an equivalent circuit **satisfiability** instance of size O(T * polylog(T)) and depth O(polylog(T)).

The Fiat-Shamir Transform

[FS86]

In a nutshell: Awesome technique for minimizing interaction in public-coin interactive protocols.

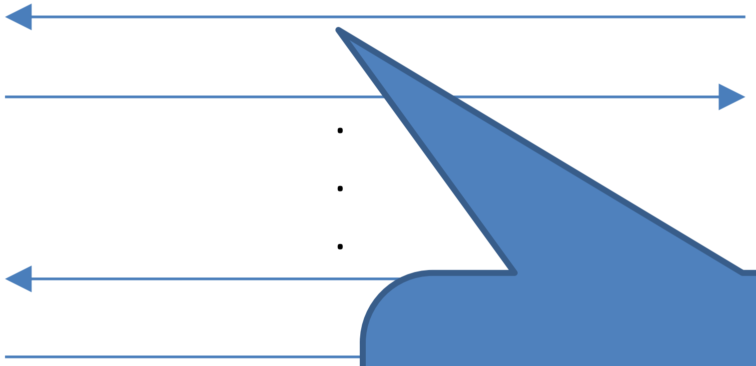
Fascinating both in theory and in practice.

Slide due to Ron Rothblum
(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

Interactive Argument [BCC88]

$x \in L?$

Prover P **Verifier V**



Public-coin if all V does is flip coins and send the result

- **Completeness**: P convinces

- **Computational Soundness** cheating prover can convince V to accept $x \notin L$ (except with negligible probability).

Slide due to Ron Rothblum
(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

Intuition

- Recall from the course reading that V's messages to P in an interactive proof are **predictable**, then the proof can be rendered non-interactive.
  - The non-interactive proof is just an "accepting transcript" of the interactive proof.
  - Intuitively, there is no reason for V to send a message to P if the prover can predict what the message will be.
    - P can just pretend the verifier sent the message, without V bothering to actually send it.
- Fiat-Shamir attempts to mimic this process **even when the verifier's messages are unpredictable.**
- First Idea: let the P choose V's challenges, which are supposed to just be random coins.
  - Problem: no way to force P to really choose the challenges uniformly at random, independent of the preceding messages in the protocol.

The Fiat-Shamir Transform

Public-Coin Interactive Argument $\xrightarrow{\text{generically}}$ Non-Interactive Argument

$\boldsymbol{P}$ $\quad$ $\boldsymbol{V}$

$\xrightarrow{\alpha_1}$

$\xleftarrow{\beta_1}$

...

$\xrightarrow{\alpha_{r-1}}$

$\xleftarrow{\beta_{r-1}}$

$\xrightarrow{\alpha_r}$

(Each β_i uniformly random)

$\boldsymbol{P_{FS}}$ $\quad$ $\boxed{\text{Hash Function } H}$ $\quad$ $\boldsymbol{V_{FS}}$

$\xrightarrow{\alpha_1}$

$\beta_1 = H(x, \alpha_1)$

Slide due to Ron Rothblum

(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

The Fiat-Shamir Transform

Public-Coin Interactive Argument $\xrightarrow{\text{generically}}$ Non-Interactive Argument

$$\boldsymbol{P} \quad \xrightarrow{\alpha_1} \quad \xleftarrow{\beta_1} \quad \cdots \quad \xrightarrow{\alpha_{r-1}} \quad \xleftarrow{\beta_{r-1}} \quad \xrightarrow{\alpha_r} \quad \boldsymbol{V}$$

(Each β_i uniformly random)

Hash Function H

$$\boldsymbol{P_{FS}} \quad \xrightarrow{\alpha_1\,,\alpha_2} \quad \boldsymbol{V_{FS}}$$

$$\beta_1 = H(x, \alpha_1)$$

Slide due to Ron Rothblum

(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

The Fiat-Shamir Transform

Public-Coin Interactive Argument $\xrightarrow{\text{generically}}$ Non-Interactive Argument

$\boldsymbol{P}$ $\qquad\qquad\qquad$ $\boldsymbol{V}$

$$\xrightarrow{\alpha_1}$$
$$\xleftarrow{\beta_1}$$
…
$$\xrightarrow{\alpha_{r-1}}$$
$$\xleftarrow{\beta_{r-1}}$$
$$\xrightarrow{\alpha_r}$$

(Each β_i uniformly random)

$\boldsymbol{P_{FS}}$ $\quad$ $\boxed{\text{Hash Function } H}$ $\quad$ $\boldsymbol{V_{FS}}$

$$\xrightarrow{\alpha_1, \dots, \alpha_r}$$

$$\beta_1 = H(x, \alpha_1)$$
$$\beta_2 = H(x, \alpha_1, \alpha_2)$$
…
$$\beta_i = H(x, \alpha_1, \dots, \alpha_i)$$

Slide due to Ron Rothblum
(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

The Fiat-Shamir Transform

Extremely influential methodology.

Powerful: We know that interaction buys a lot.

FS makes interaction free.

Practical: Very low overhead.

Slide due to Ron Rothblum
(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

Security of Fiat-Shamir

The Random Oracle Model [BR93]

The random oracle model simply means that all parties are given blackbox access to a fully random function $R: \{0,1\}^\lambda \to \{0,1\}^\lambda$.

Security should hold whp over the choice of R.

Q: How should we view protocols secure in ROM?

A: Protocols secure in the ROM are widely viewed as "secure in practice" by practitioners.

Slide due to Ron Rothblum (but answer to Q has been edited)
(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

FS in the ROM

Public-Coin Interactive Argument $\xrightarrow{\text{generically}}$ Non-Interactive Argument

$\boldsymbol{P}$ $\quad$ $\boldsymbol{V}$

$$\xrightarrow{\alpha_1}$$
$$\xleftarrow{\beta_1}$$
…
$$\xrightarrow{\alpha_{r-1}}$$
$$\xleftarrow{\beta_{r-1}}$$
$$\xrightarrow{\alpha_r}$$

(Each β_i uniformly random)

$\boldsymbol{P_{FS}}$ $\quad$ $\boxed{\text{Random Oracle } R}$ $\quad$ $\boldsymbol{V_{FS}}$

$$\xrightarrow{\alpha_1, \dots, \alpha_r}$$

$\beta_1 = R(x, \alpha_1)$

$\beta_2 = R(x, \alpha_1, \alpha_2)$

…

$\beta_i = R(x, \alpha_1, \dots, \alpha_i)$

Slide due to Ron Rothblum
(http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-7-_fiat_shamir_basic.pdf)

Recent Theoretical Results on Fiat-Shamir

- Recent theoretical results show that applying the Fiat-Shamir transformation to **all** of the interactive proofs we have seen in this course so far leads to a sound argument in the Random Oracle Model (previously soundness in the Random Oracle Model was known for constant-round protocols).

Recent Theoretical Results on Fiat-Shamir

- Recent theoretical results show that applying the Fiat-Shamir transformation to **all** of the interactive proofs we have seen in this course so far leads to a sound argument in the Random Oracle Model (previously soundness in the Random Oracle Model was known for constant-round protocols).

- Furthermore, recent results show that even when instantiating the Random Oracle with a concrete hash function satisfying a property called **correlation intractability**, the resulting argument is sound.

Recent Theoretical Results on Fiat-Shamir

- Recent theoretical results show that applying the Fiat-Shamir transformation to **all** of the interactive proofs we have seen in this course so far leads to a sound argument in the Random Oracle Model (previously soundness in the Random Oracle Model was known for constant-round protocols).

- Furthermore, recent results show that even when instantiating the Random Oracle with a concrete hash function satisfying a property called **correlation intractability**, the resulting argument is sound.
  - Still wide-open whether similar results are true when applying Fiat-Shamir to the public-coin **arguments** we will see later in this course.

Intuition for Security

- In any round i of the sum-check protocol, if P knew what V's next message r_i would be, P could cheat.
  - Let g_i be the polynomial P is supposed to send in round i, and s_i be the polynomial the prover actually sends.
  - Suppose P knows the value r_i that V will send in round i.
  - Then P can choose s_i so that $s_i \neq g_i$, yet $s_i(r_i) = g_i(r_i)$.
- In the Fiat-Shamir transformation, r_i is set to be:

  R(the transcript up to to round i, which includes s_i).
- So P cannot run the above attack unless it can find an (s_i, r_i) pair such that R(the transcript up to to round i, which includes s_i)= r_i $\mathbf{and}$ $s_i(r_i) = g_i(r_i)$.
- Correlation-intractability is defined to ensure finding such a pair is intractable.

Turning Computer Programs into Circuits

Example 1: Squaring the entries of a vector and then summing the results

Matrix multiplication: one n-dimensional vector vector a over $\boldsymbol{F}$, desired output is$\sum_{k=1}^{n} a_k^2$

Naïve algorithm (sequential):
Initialize c to be 0.
For i in $\{1, 2, \dots, n\}$ **do:**

$$c \leftarrow c + a_i^2$$

Naïve algorithm (parallel):
For i in $\{1, 2, \dots, n\}$ **in parallel do:**

$$T_i \leftarrow a_i^2$$

For i in $\{1, 2, \dots, n\}$ **in parallel do:**

$$\mathrm{c} \leftarrow \sum_{i=1}^{n} T_k$$

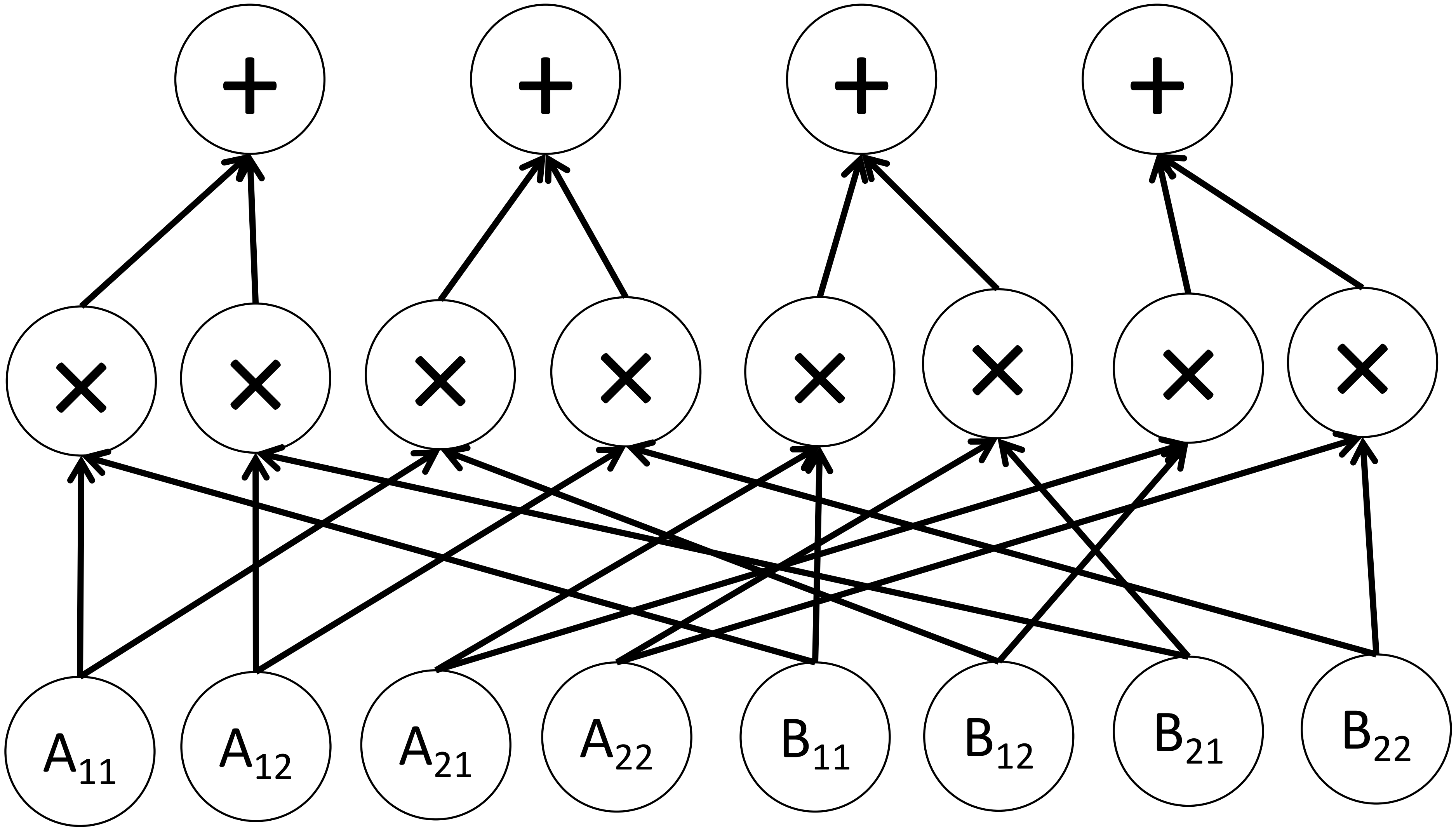
Corresponding Circuits

Example 2: Matrix Multiplication

Matrix multiplication: input is two $n \times n$ matrices A, B over $\boldsymbol{F}$, desired output is $A * B$

Naïve algorithm (sequential):
Initialize C to be an $n \times n$ matrix with all entries equal to 0.
For i in $\{1, 2, \dots, n\}$ **do:**
    For j in $\{1, 2, \dots, n\}$ **do:**
        For k in $\{1, 2, \dots, n\}$ **do:**
            $C_{i,j} \leftarrow C_{i,j} + A_{i,k} * B_{k,j}$

Naïve algorithm (parallel):
For (i, j, k) in $\{1, 2, \dots, n\}$ **in parallel do:**
    $T_{i,j,k} \leftarrow A_{i,k} * B_{k,j}$
For (i, j) in $\{1, 2, \dots, n\}$ **in parallel do:**
    $C_{i,j} \leftarrow \sum_{k=1}^{n} T_{i,j,k}$


+
+
+
+
×
×
×
×
×
×
×
×
A_{11}
A_{12}
A_{21}
A_{22}
B_{11}
B_{12}
B_{21}
B_{22}