Candidate Witness Encryption from Lattice Assumptions

Rotem Tsabary

Reichman University (IDC), Israel

Google Research, Israel

Witness Encryption [Garg, Gentry, Sahai, Waters, STOC13]

Encrypt a message s.t. decryption requires solving a problem in NP.

Encrypt with respect to a predicate $f: \{0,1\}^n \rightarrow \{0,1\}$.

Decrypt with any witness w where $f(w) = 1$.

Security: decryption is hard if $f(w) = 0$ for all w.

Witness Encryption [Garg, Gentry, Sahai, Waters, STOC13]

Existing Candidates

- From MMaps [GLW14]
- Lattice-based candidates [WZ17,CVW18]
- From iO

Motivation

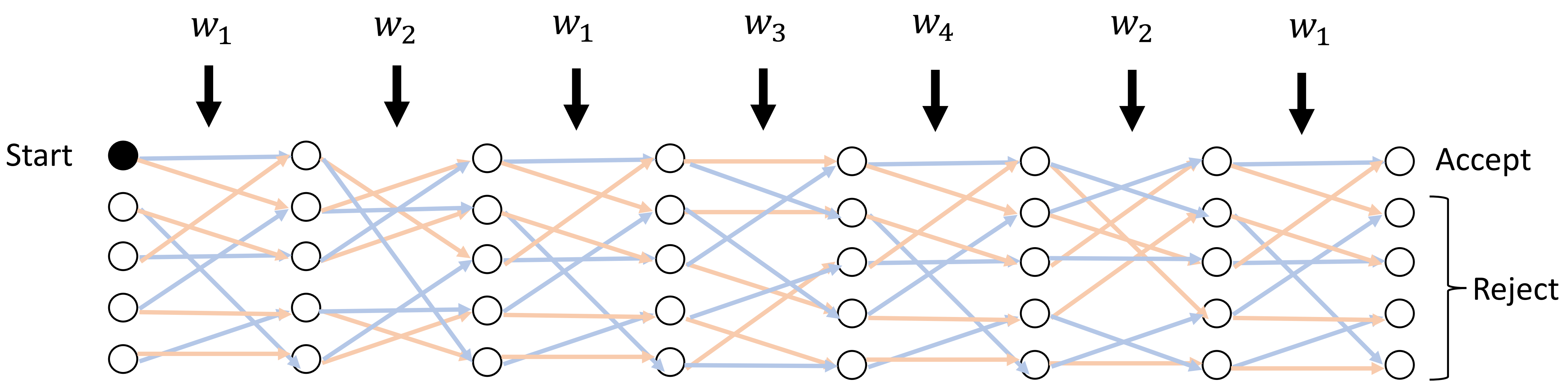
A 'simple' lattice-based candidate with better insight on its security.

Our Contribution

[Wee22] Evasive LWE

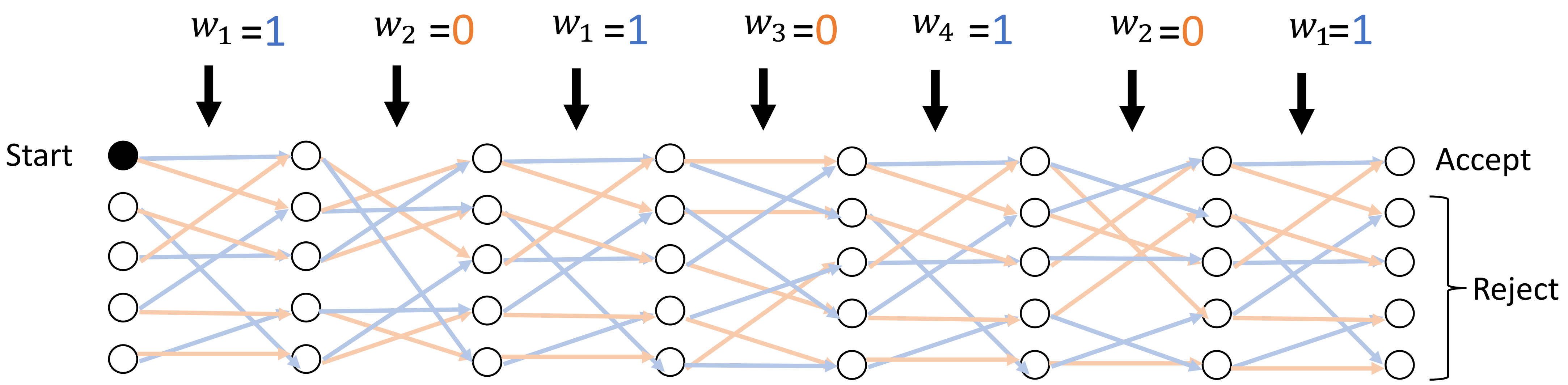
A candidate with provable security from a new* lattice assumption.

Branching Programs



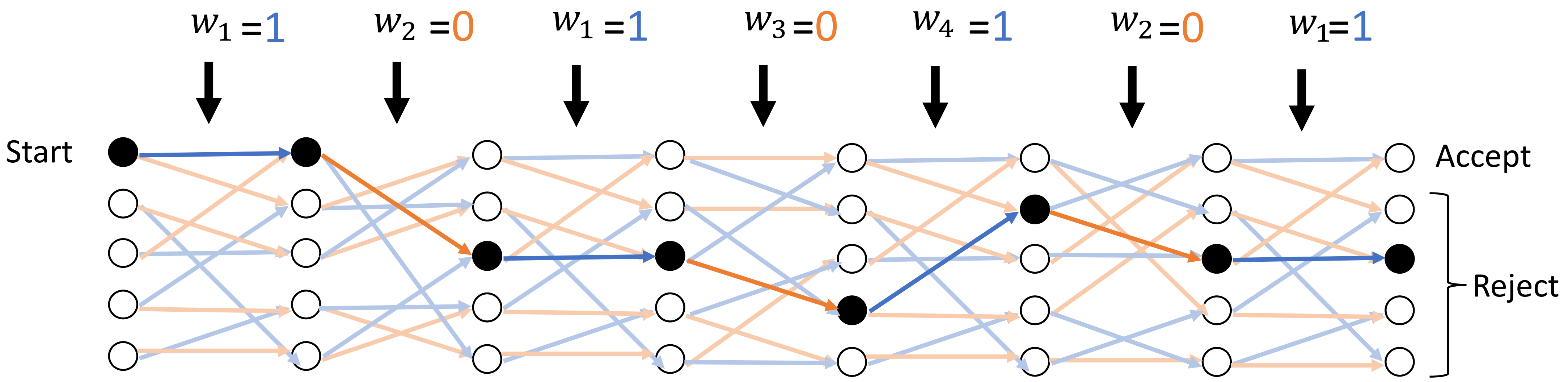
Branching Programs

$$w = 1001$$



Branching Programs

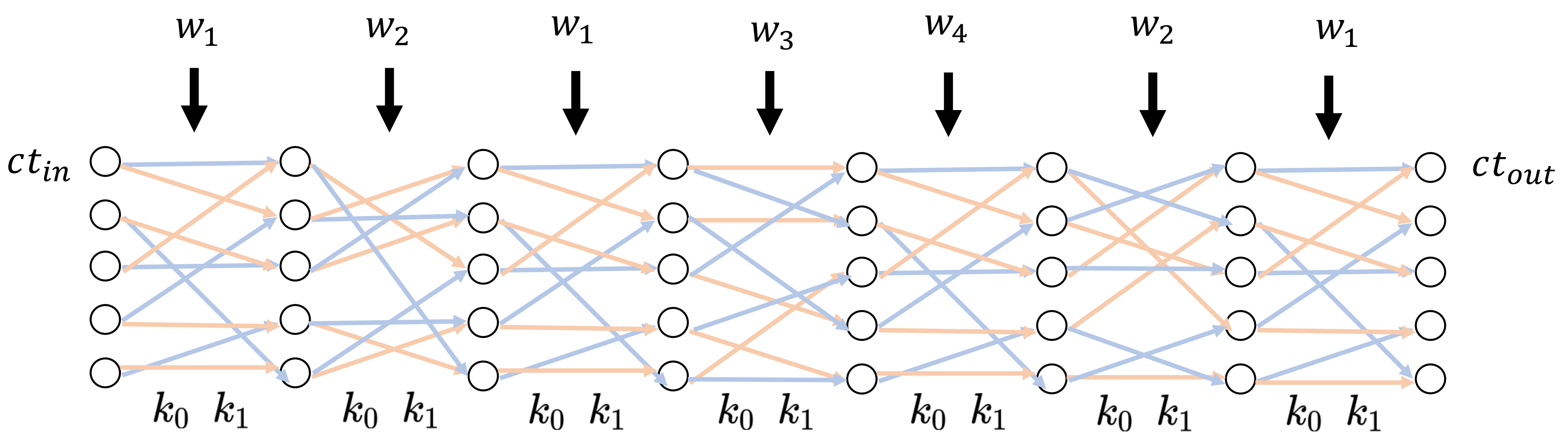
$w = 1001$



[Barrington 89]: Every $f \in NC^1$ can be computed by a poly-sized BP.

Witness Encryption from Branching Programs

A natural approach: ct_{in}, ct_{out}, key-pair for each level

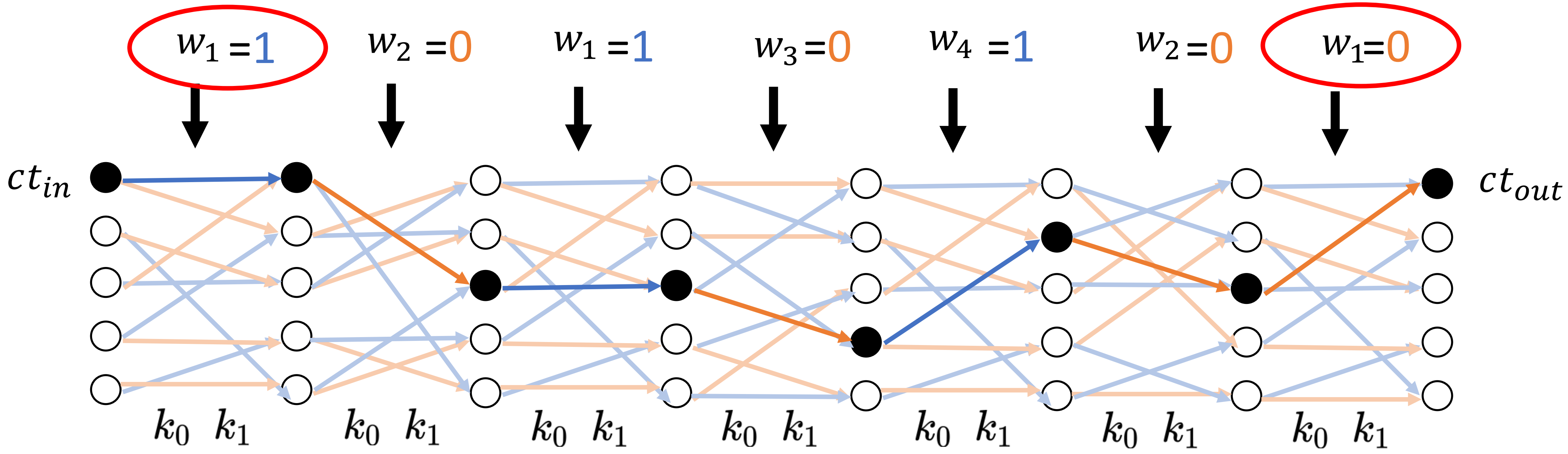


Security relies on the premise that $f(w) = 0$ for all w.

Problem: Inconsistent inputs might result in the accepting state.

Witness Encryption from Branching Programs

A natural approach: ct_{in}, ct_{out}, key-pair for each level

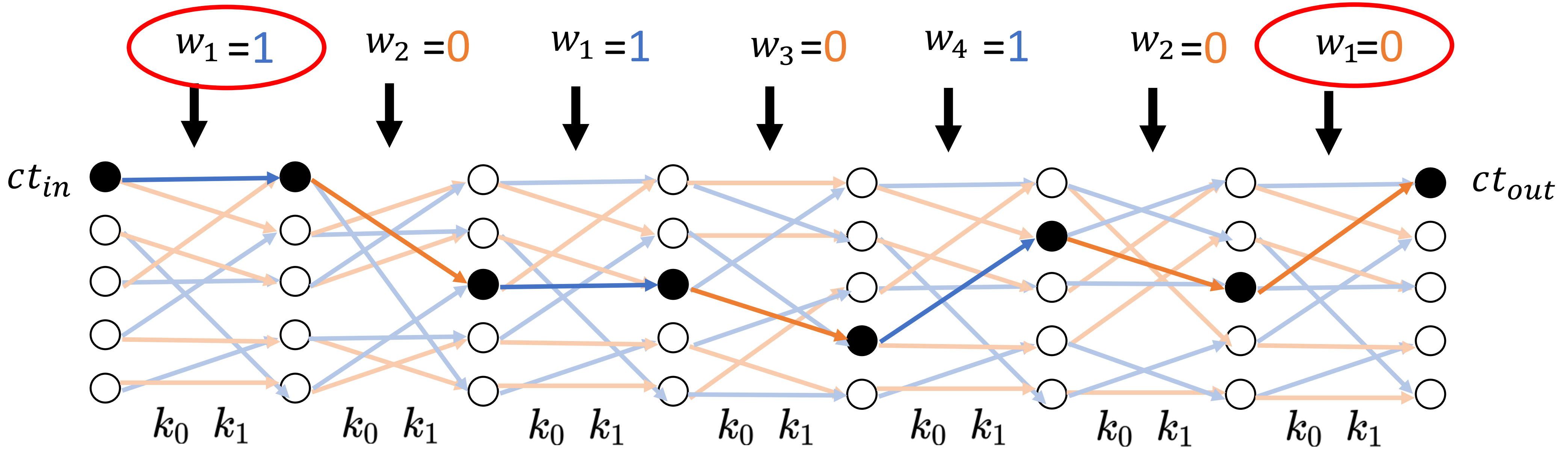


Security relies on the premise that $f(w) = 0$ for all w.

Problem: Inconsistent inputs might result in the accepting state.

Witness Encryption from Branching Programs

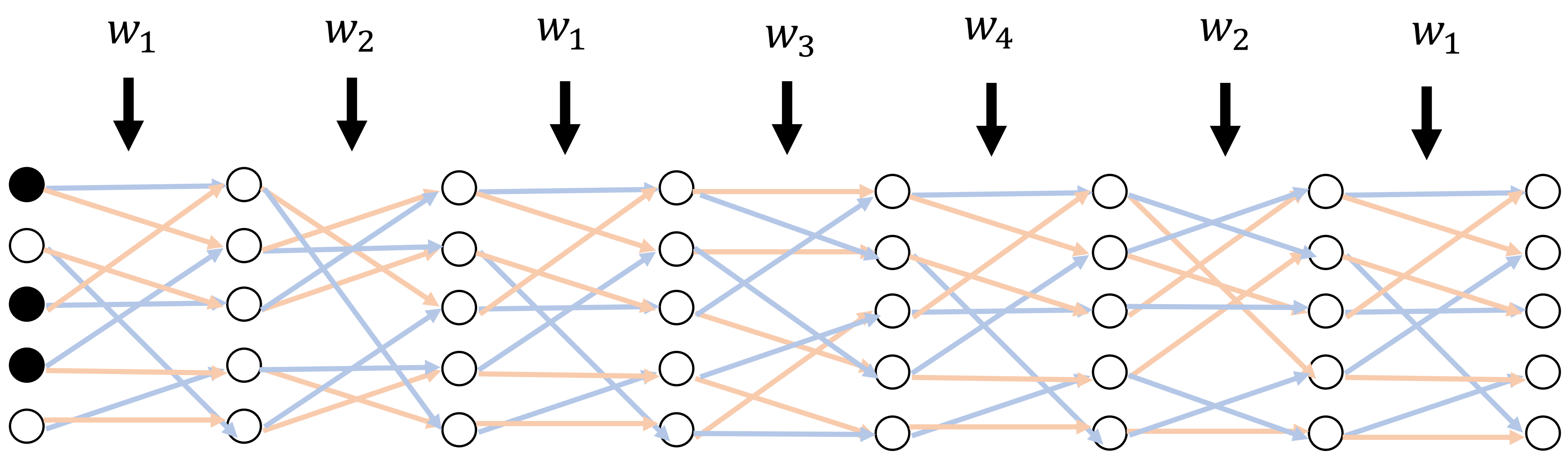
A natural approach: ct_{in}, ct_{out}, key-pair for each level



Observation: WE for read-once BP is trivial.

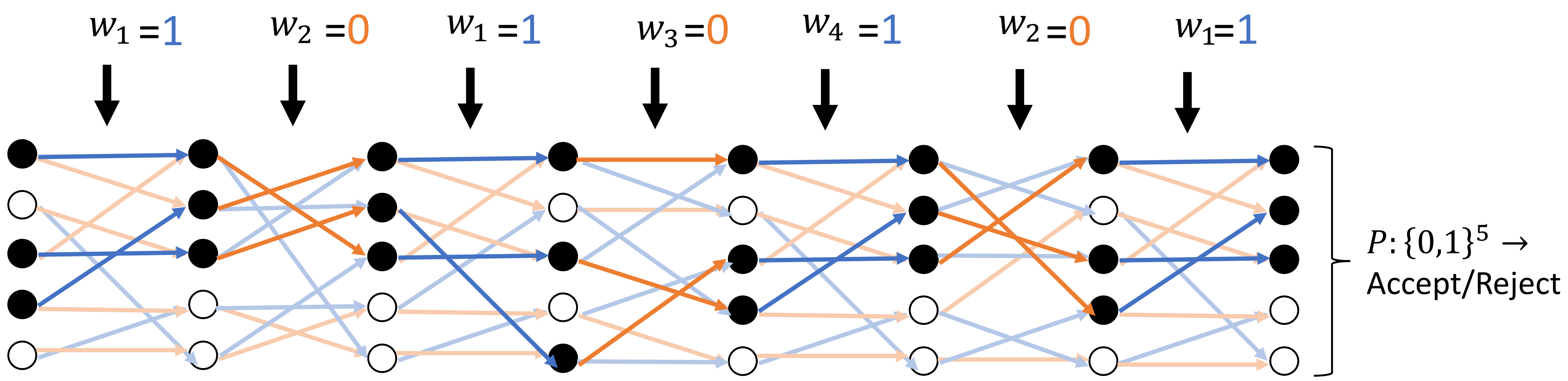
BP with Consistency Check [CHVW19]

Multi-State Branching Program



BP with Consistency Check [CHVW19]

Multi-State Branching Program

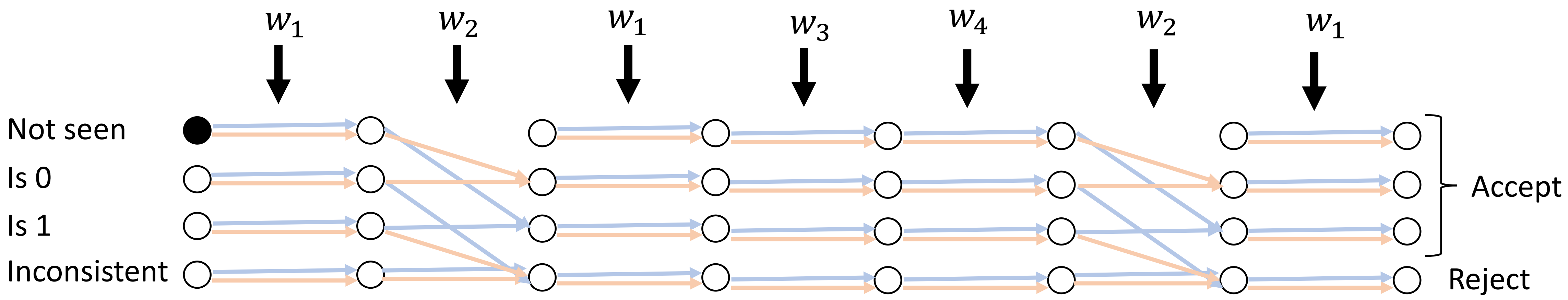


BP with Consistency Check [CHVW19]

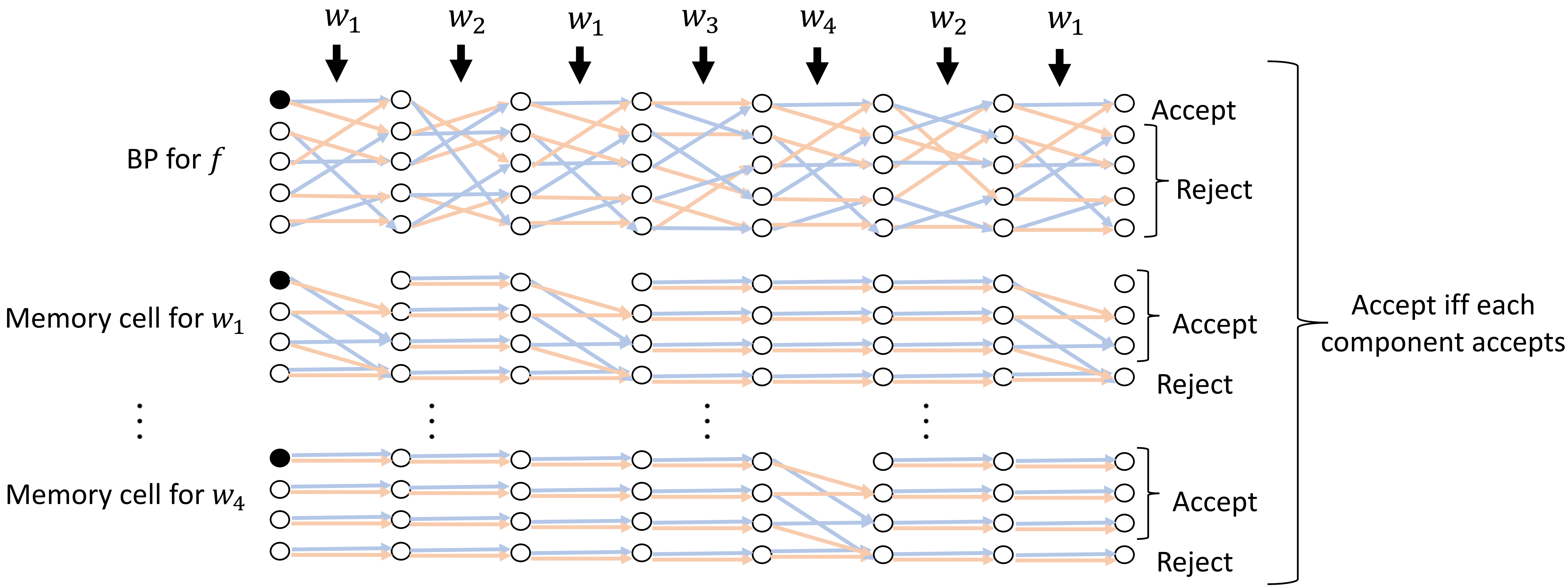
Consistency check:

Add a 'memory cell' for each input bit that verifies its consistency.

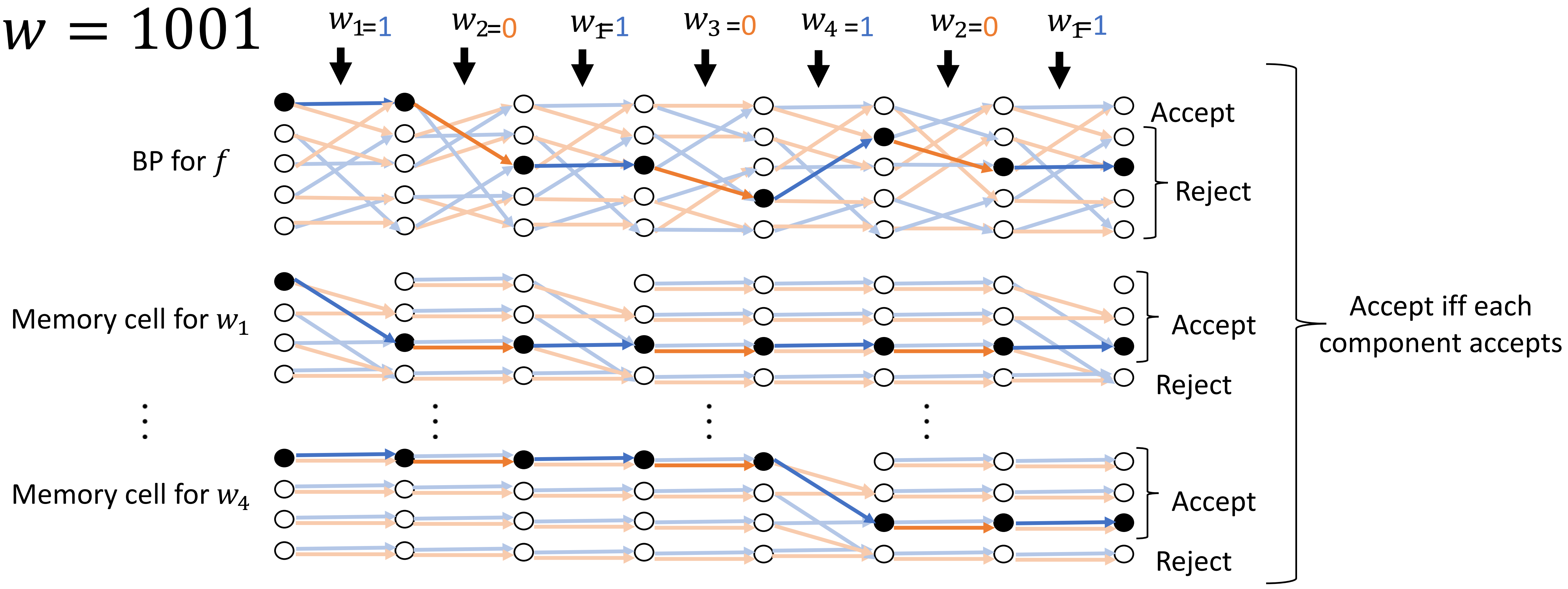
Memory cell for w_2:



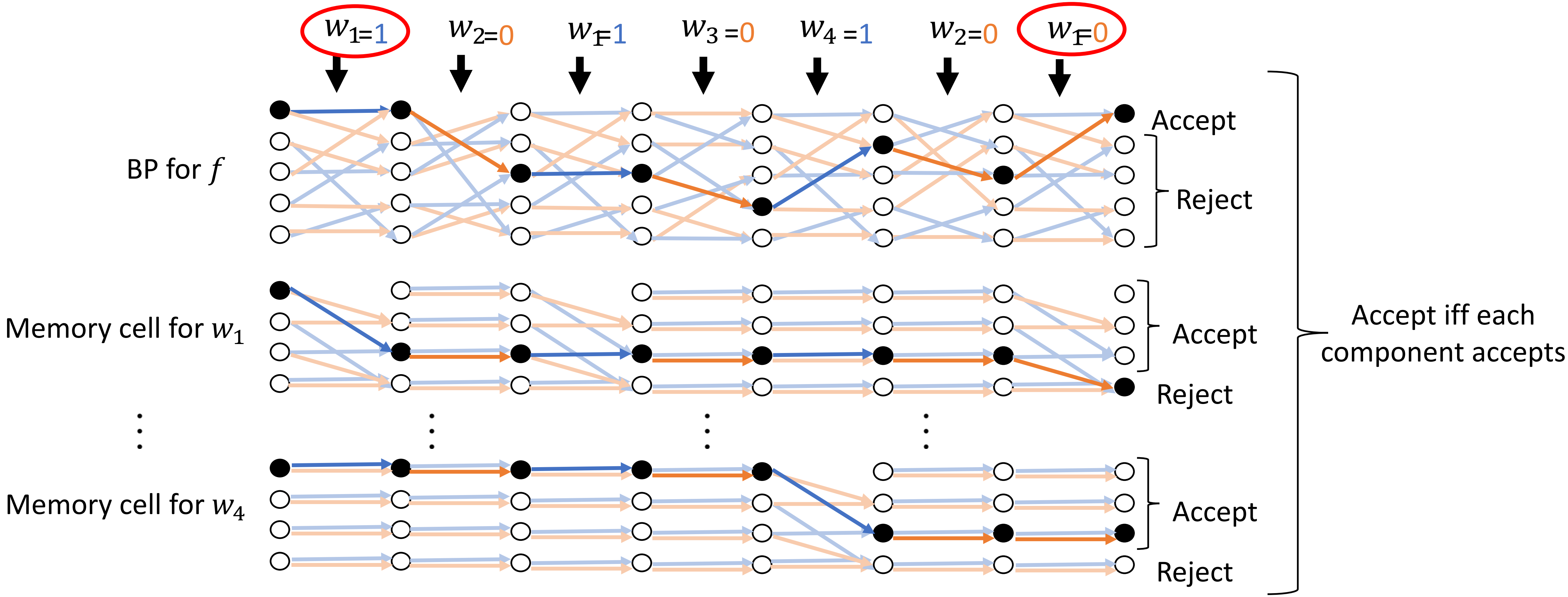
BP with Consistency Check [CHVW19]



BP with Consistency Check [CHVW19]

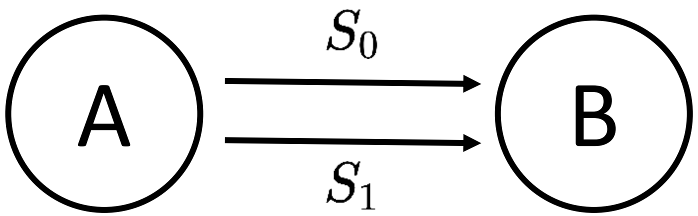


BP with Consistency Check [CHVW19]



Graph-Induced Lattice Encodings (GGH15)

[Gentry, Gorbunov, Halevi, TCC15]



$$Encode(A^{td}, S_0, B) \rightarrow K_0$$

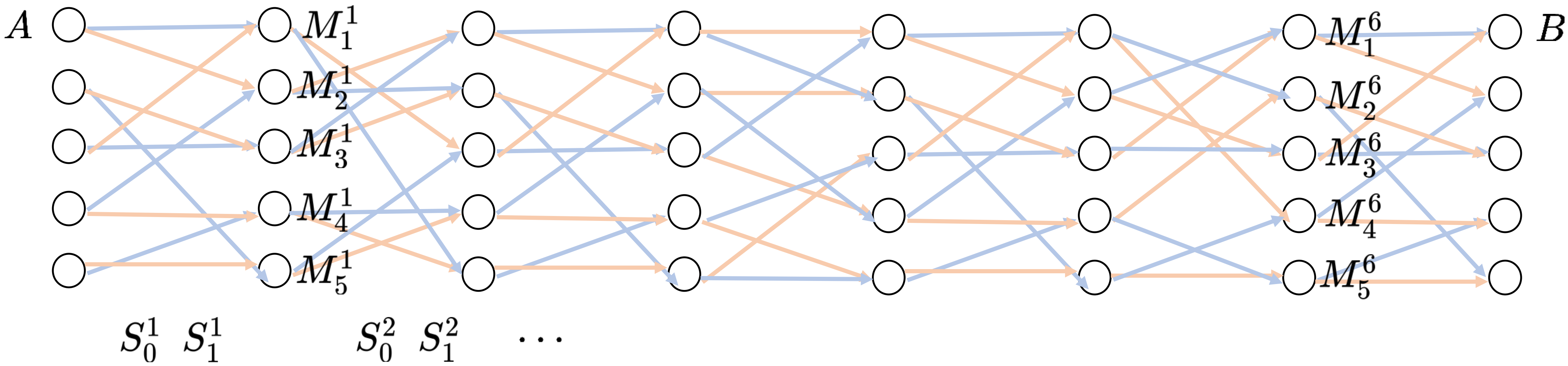
$$Encode(A^{td}, S_1, B) \rightarrow K_1$$

$$(sA + e)K_0 = sS_0B + e'$$

$$(sA + e)K_1 = sS_1B + e'$$

GGH15 Encodings for Branching Programs

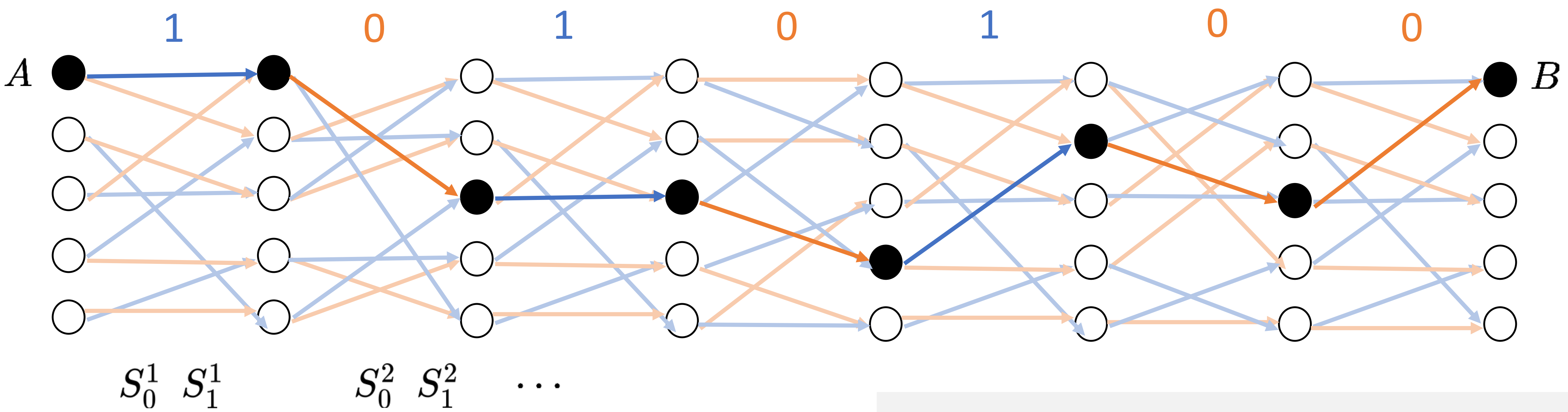
[GGH15,CC17,WZ17,GKW17a,CVW18,CHVW19]



BP Encoding: $sA + e, \ \{K_0^i, K_1^i\}_i$

GGH15 Encodings for Branching Programs

[GGH15,CC17,WZ17,GKW17a,CVW18,CHVW19]



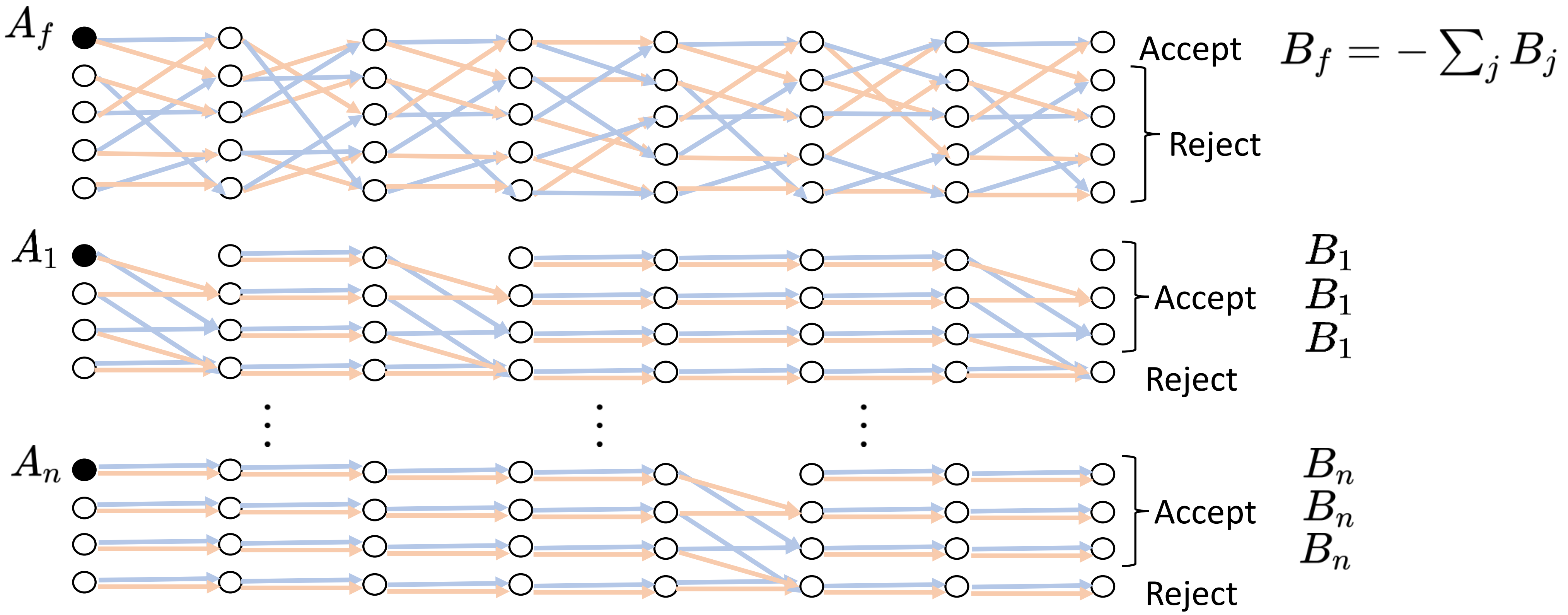
Evaluation on x:

$$(sA + e)\prod_i K^i_{x_i} \approx s\prod_i S^i_{x_i} B$$

[BPR12,CC17]:

$$PRF(x) := s\prod_i S^i_{x_i} B$$

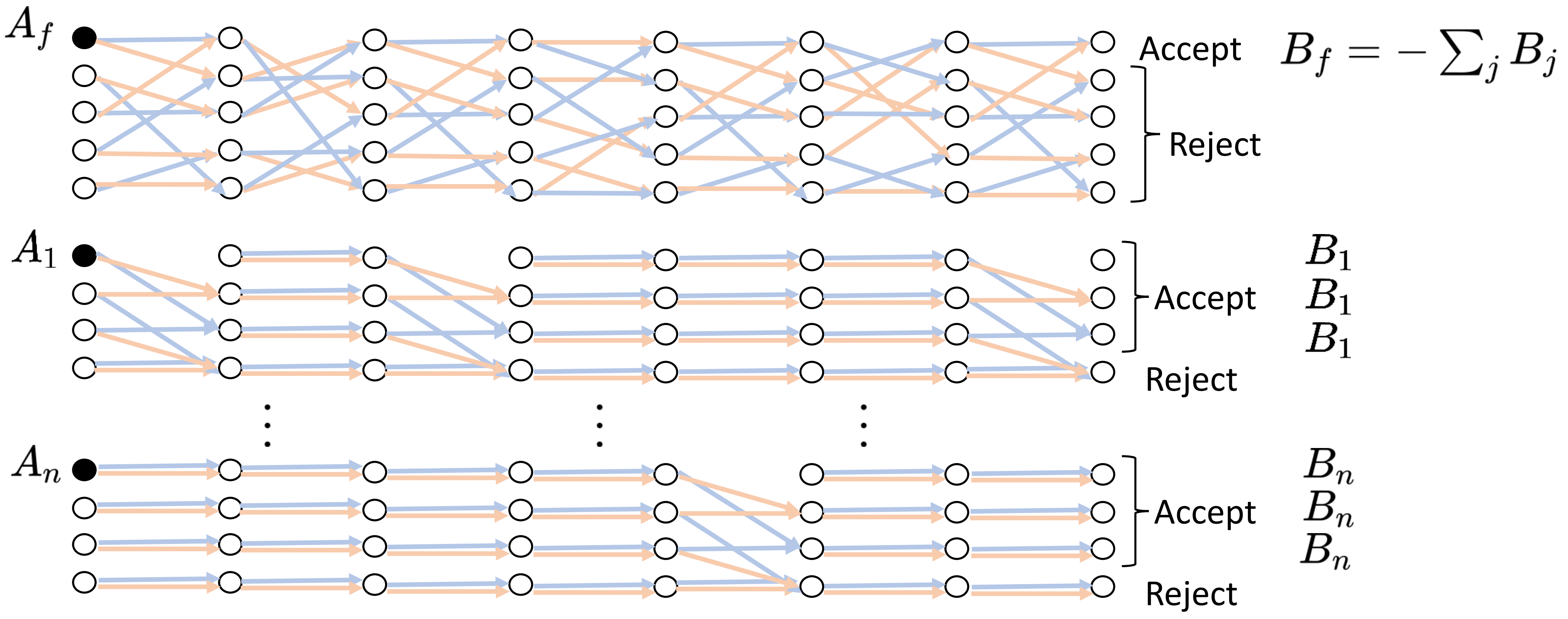
Encoding the Consistency-Checking BP



BP Encoding: $s[A_f|A_1|\ldots|A_n] + e, \ \{K_0^i, K_1^i\}_i$

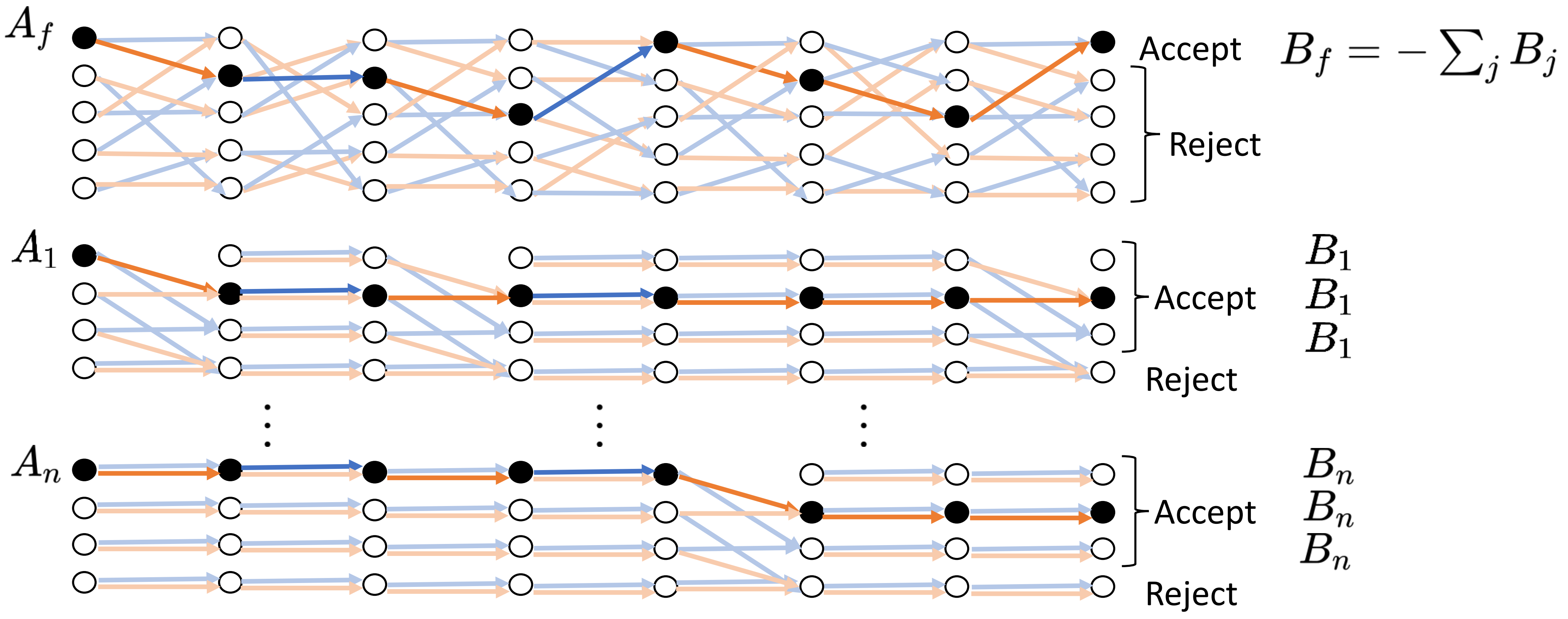
Can compute $s\prod_i S_{x_i}^i B$ for any transcript x that leads to a matrix B.

Decryption



Let x be a transcript of a witness w s.t. $f(w) = 1$.

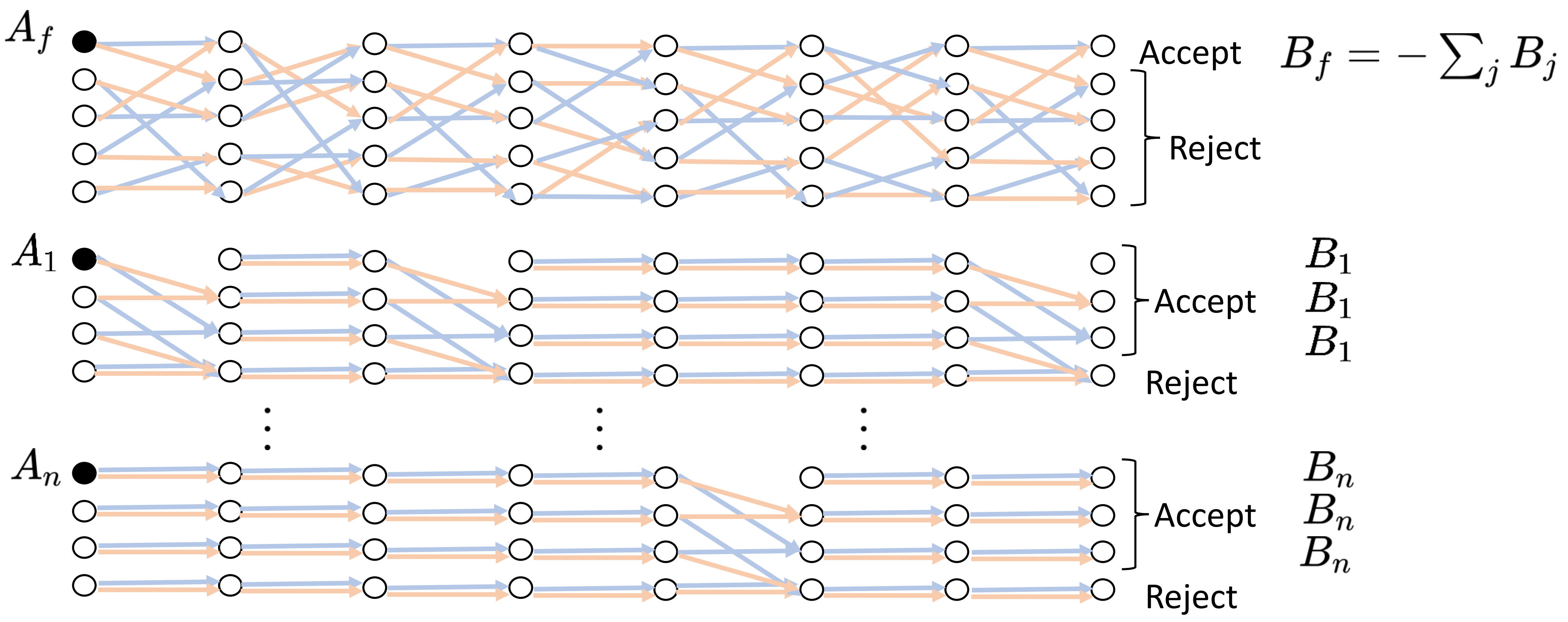
Decryption



Let x be a transcript of a witness w s.t. $f(w) = 1$.

Compute $s \prod_i S_{x_i}^i \boldsymbol{B_f} + \sum_j s \prod_i S_{x_i}^i \boldsymbol{B_j}$

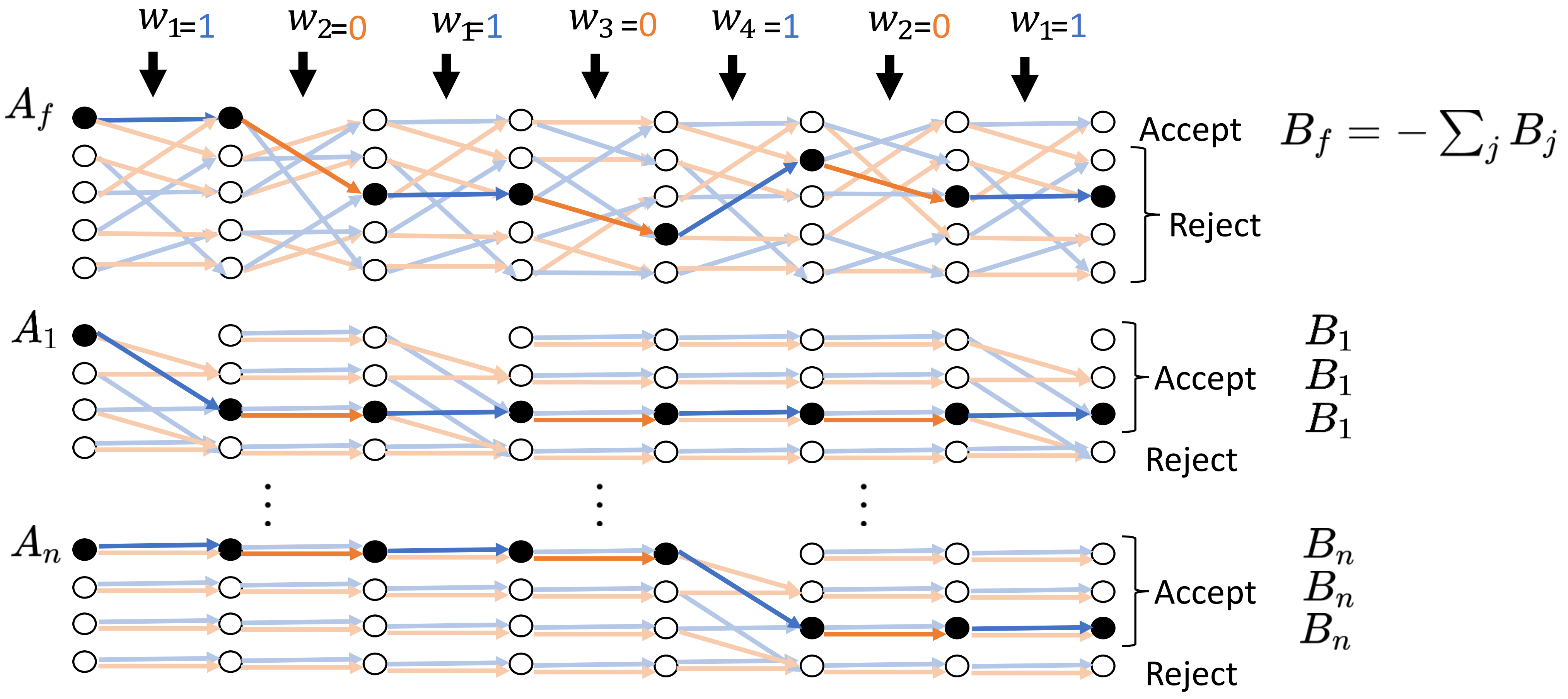
Security - Intuition



Let x be a consistent transcript w.r.t. some w.

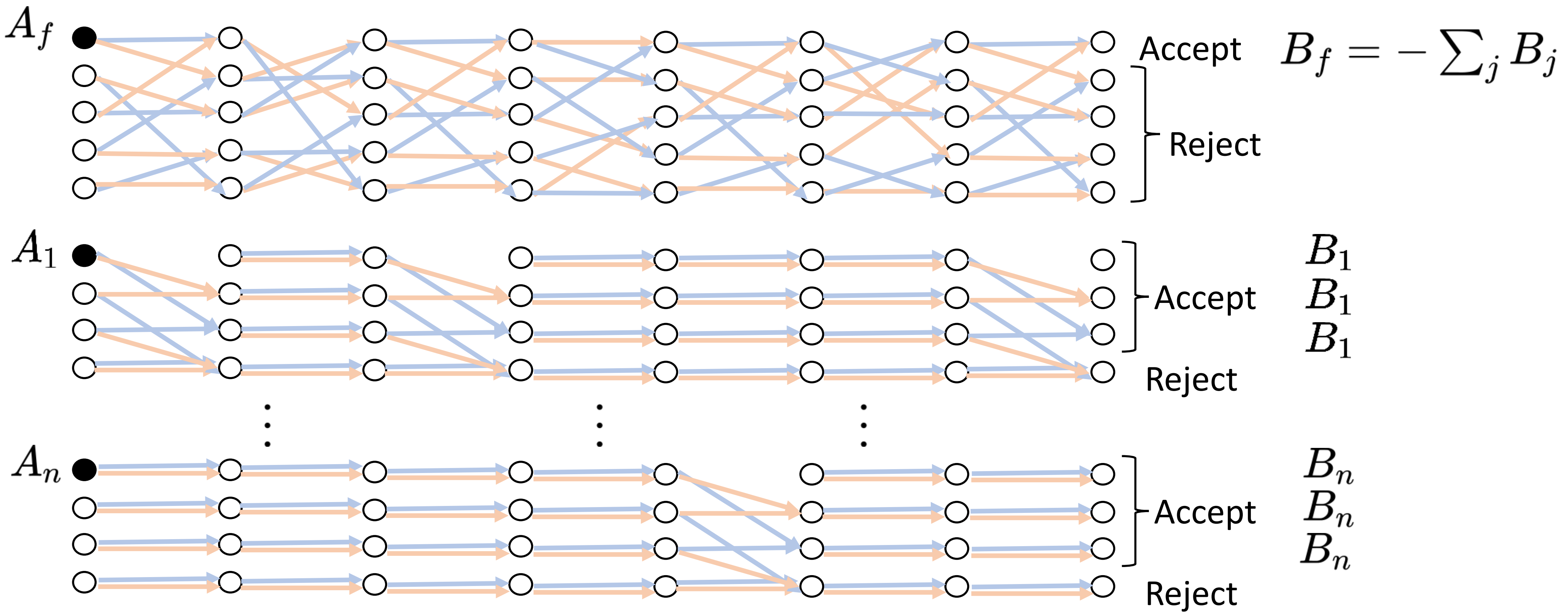
Recall: $f(w) = 0$ for all w.

Security - Intuition



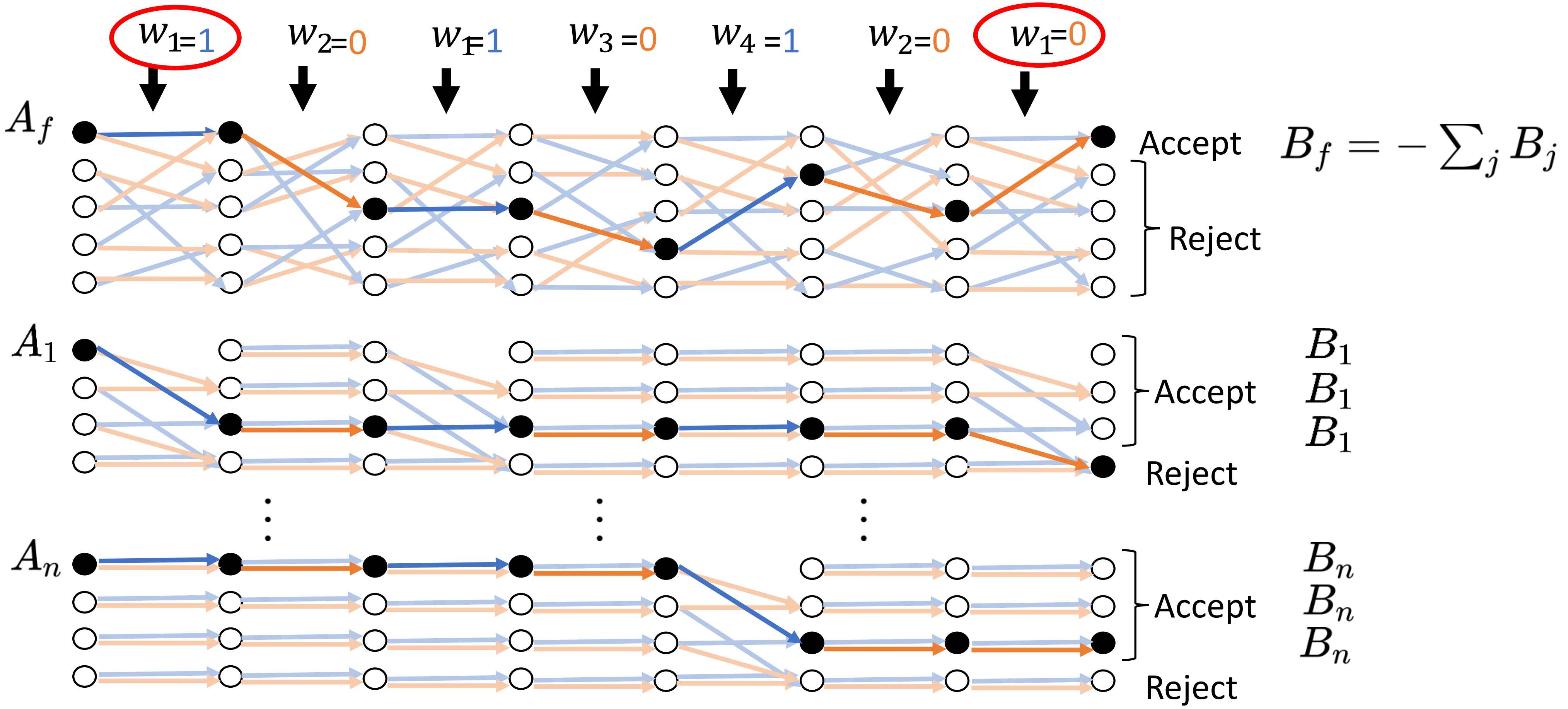
Let x be a consistent transcript w.r.t. some w.
Recall: $f(w) = 0$ for all w. $\Longrightarrow$ Cannot compute $s \prod_i S^i_{x_i} \boldsymbol{B_f}$

Security - Intuition



Let x be a transcript inconsistent at index j.

Security - Intuition



Let x be a transcript inconsistent at index j. $\Longrightarrow$ Cannot compute $s\prod_i S^i_{x_i}\,\boldsymbol{B_j}$

Security Analysis – GGH15 Example

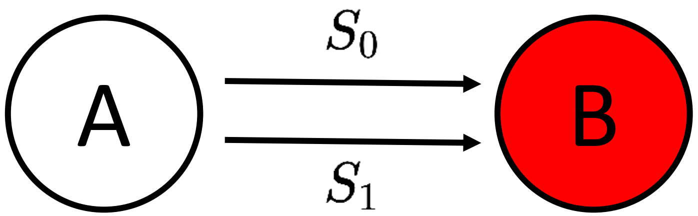
$$\begin{tikzcd} A \arrow[r, shift left, "S_0"] \arrow[r, shift right, "S_1"'] & B \end{tikzcd}$$

$$Encode\left(A^{td}, S_0, B\right) \to K_0$$

$$Encode\left(A^{td}, S_1, B\right) \to K_1$$

$$sA + e$$

Security Analysis – GGH15 Example



$Encode(A^{td}, S_0, B) \rightarrow K_0$

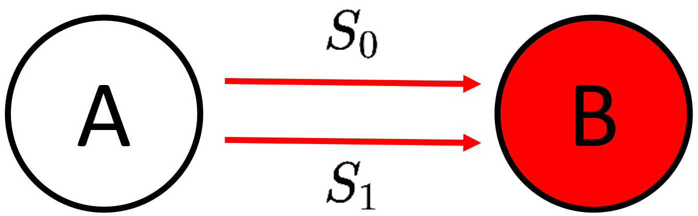
$Encode(A^{td}, S_1, B) \rightarrow K_1$

$sA + e$

Standard Analysis Steps:

1. LWE w.r.t. B is hard.

Security Analysis – GGH15 Example



$Encode(A^{td}, S_0, B) \to K_0$

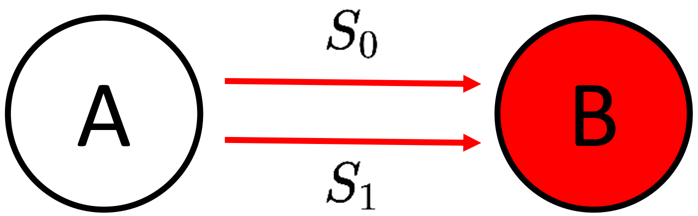
$Encode(A^{td}, S_1, B) \to K_1$

$sA + e$

Standard Analysis Steps:

1. LWE w.r.t. B is hard.
2. Simulate K_0, K_1 without a trapdoor.

Security Analysis – GGH15 Example



$Encode(A^{td}, S_0, B) \to K_0$

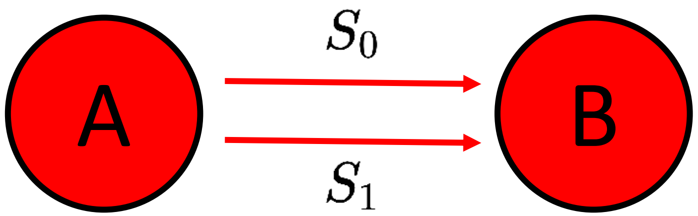
$Encode(A^{td}, S_1, B) \to K_1$

$sA + e$

Standard Analysis Steps:

1. LWE w.r.t. B is hard.
2. Simulate K_0, K_1 without a trapdoor.
3. Generate A without a trapdoor.

Security Analysis – GGH15 Example



$$Encode(A^{td}, S_0, B) \rightarrow K_0$$

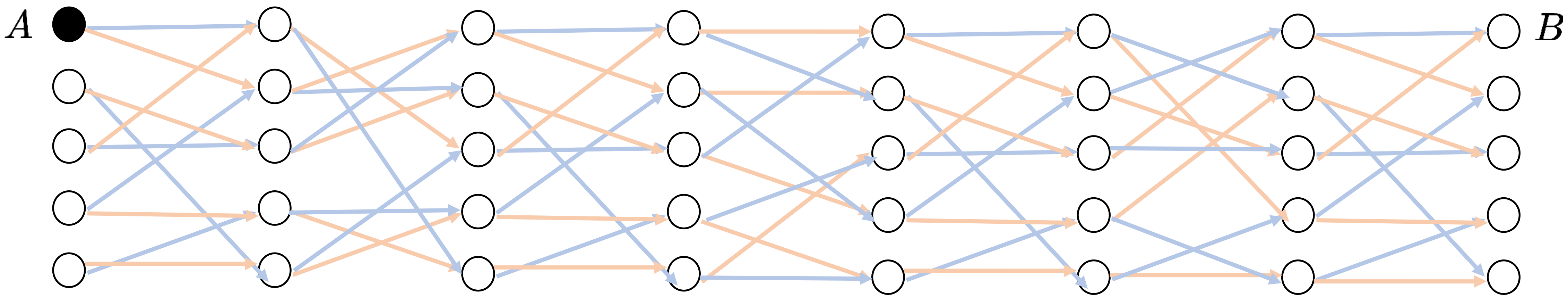
$$Encode(A^{td}, S_1, B) \rightarrow K_1$$

$$sA + e$$

Standard Analysis Steps:

1. LWE w.r.t. B is hard.
2. Simulate K_0, K_1 without a trapdoor.
3. Generate A without a trapdoor.
4. LWE w.r.t. A is hard.

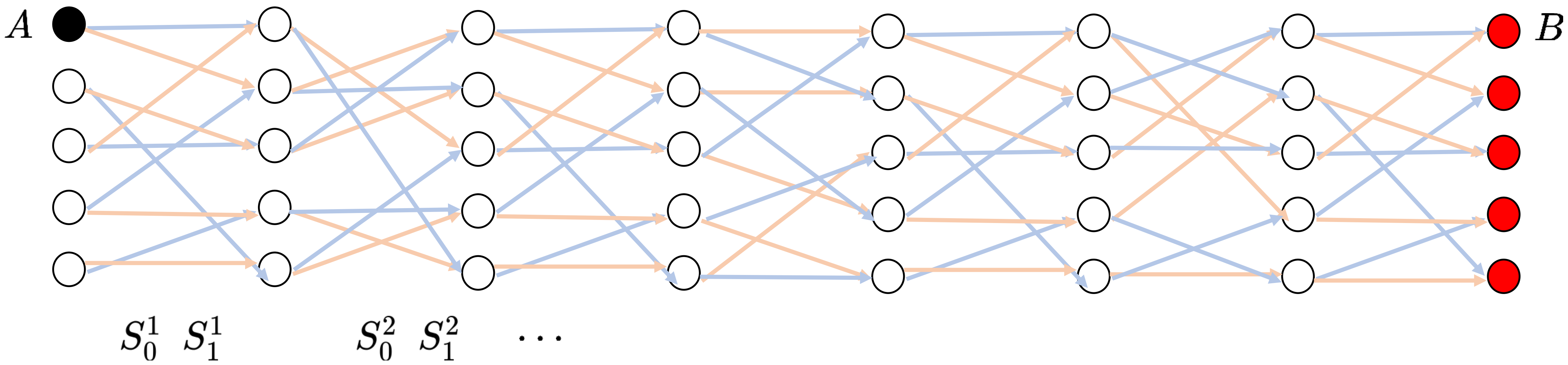
Security Analysis – BP Encoding



$S_0^1 \;\; S_1^1 \qquad S_0^2 \;\; S_1^2 \quad \cdots$

$$\boldsymbol{s}A + \boldsymbol{e}, \quad \{K_0^i, K_1^i\}_i$$

Security Analysis – BP Encoding

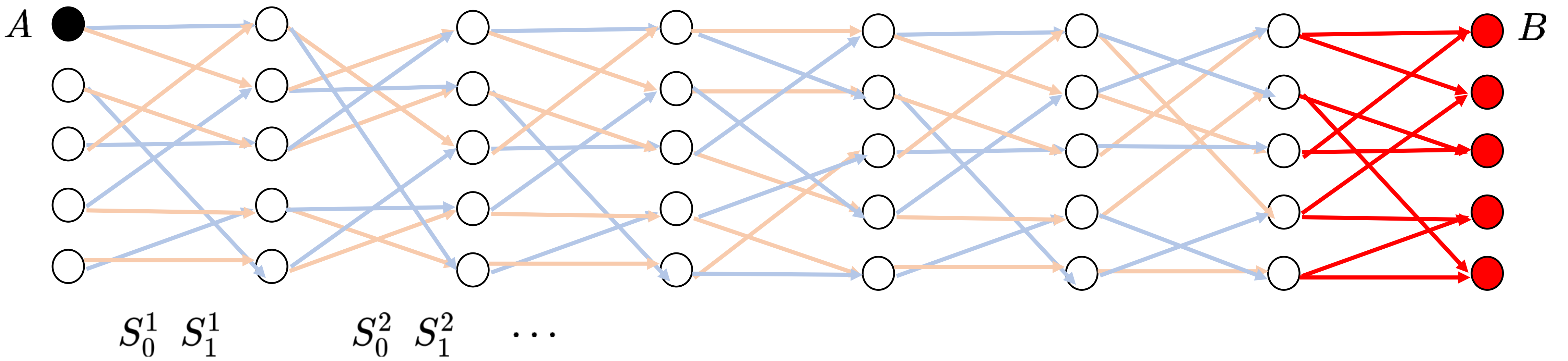


$$sA + e, \quad \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.

Security Analysis – BP Encoding

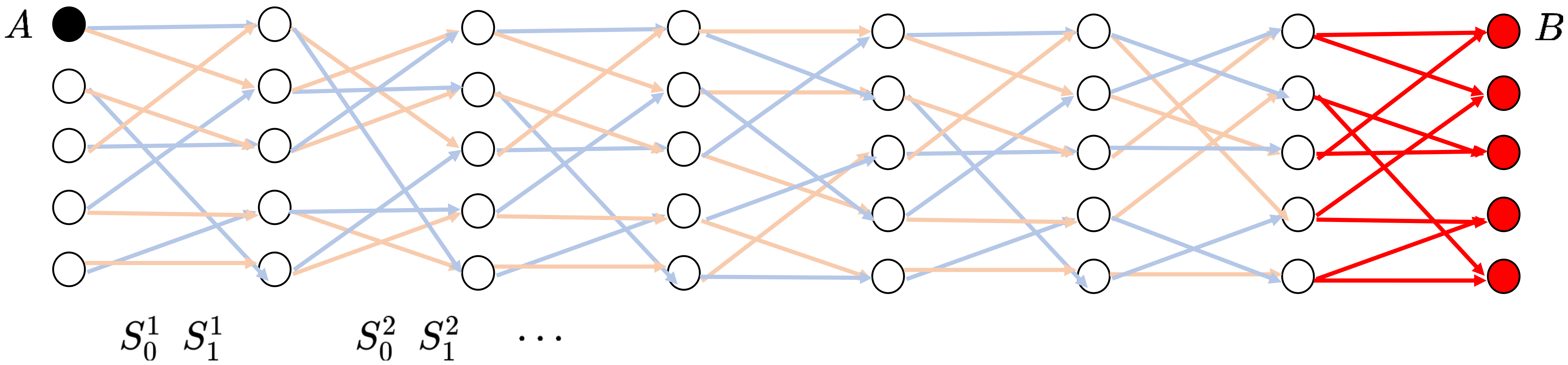


$$sA + e, \quad \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.

Security Analysis – BP Encoding

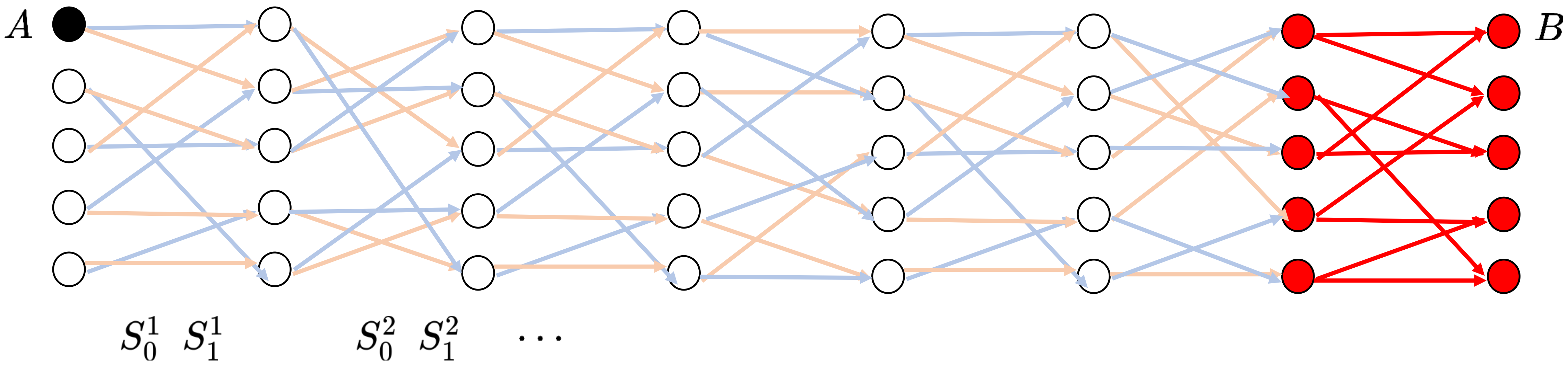


$$sA + e, \quad \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.
3. Generate (i-1)'th level without a trapdoor.

Security Analysis – BP Encoding

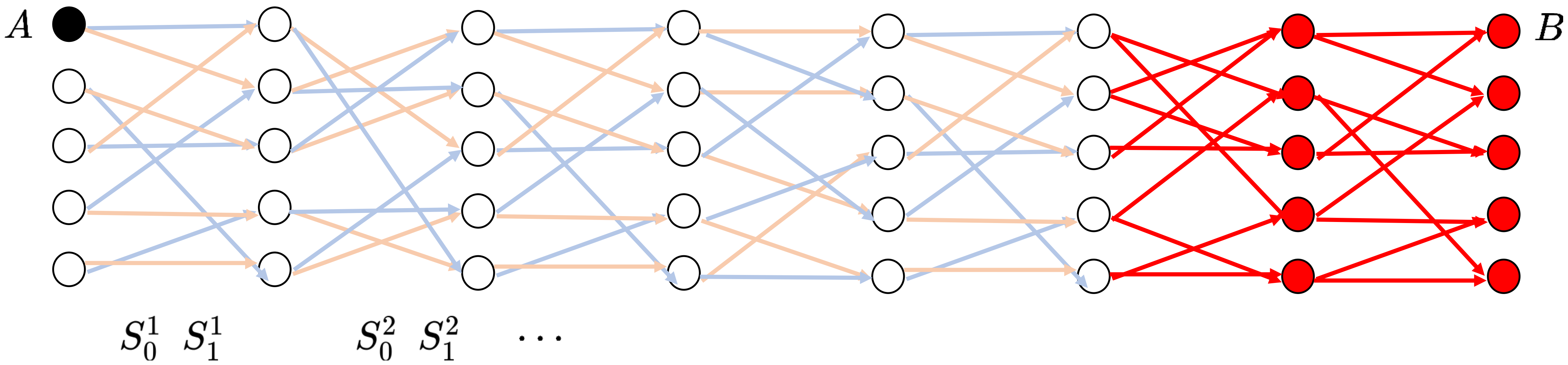


$$sA + e, \quad \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.
3. Generate (i-1)'th level without a trapdoor.
4. LWE w.r.t. (i-1)'th level is hard.

Security Analysis – BP Encoding

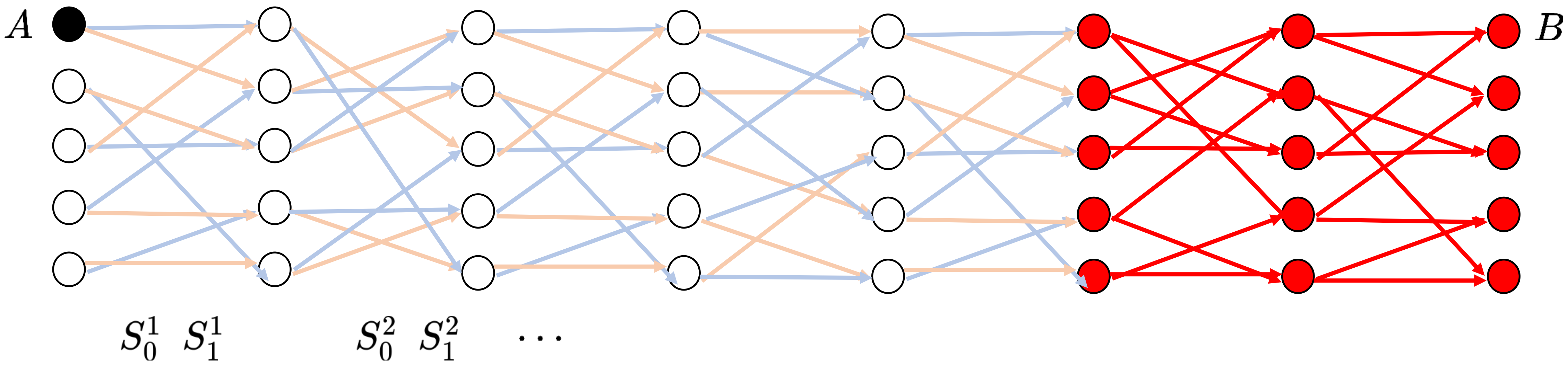


$$sA + e, \quad \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.
3. Generate (i-1)'th level without a trapdoor.
4. LWE w.r.t. (i-1)'th level is hard.

Security Analysis – BP Encoding

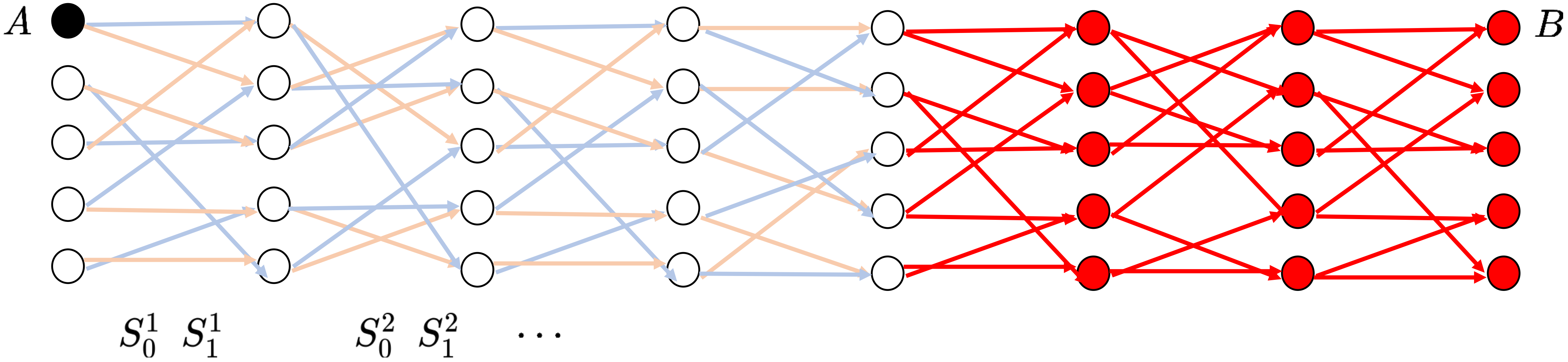


$$sA + e, \;\; \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.
3. Generate (i-1)'th level without a trapdoor.
4. LWE w.r.t. (i-1)'th level is hard.

Security Analysis – BP Encoding

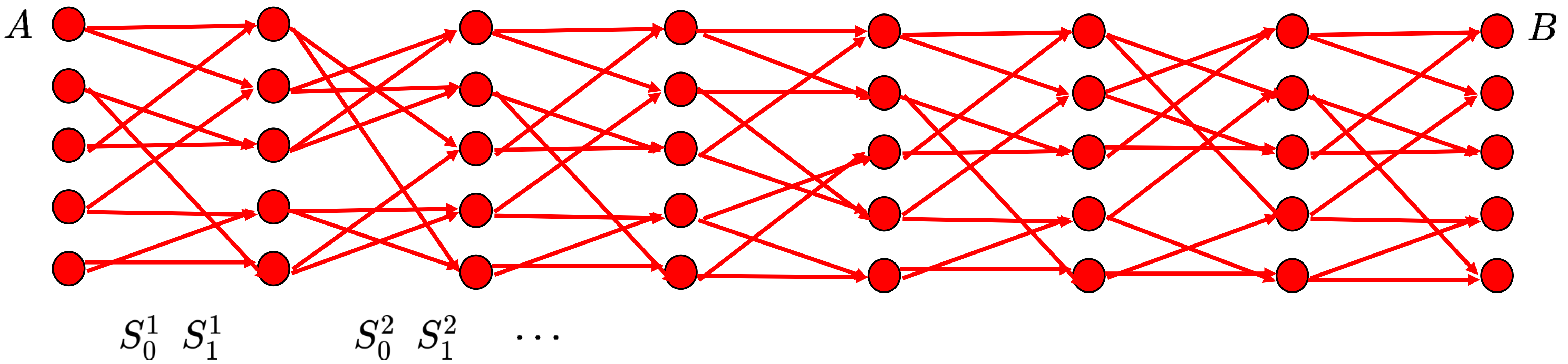


$$sA + e, \quad \{K_0^i, K_1^i\}_i$$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.
3. Generate (i-1)'th level without a trapdoor.
4. LWE w.r.t. (i-1)'th level is hard.

Security Analysis – BP Encoding

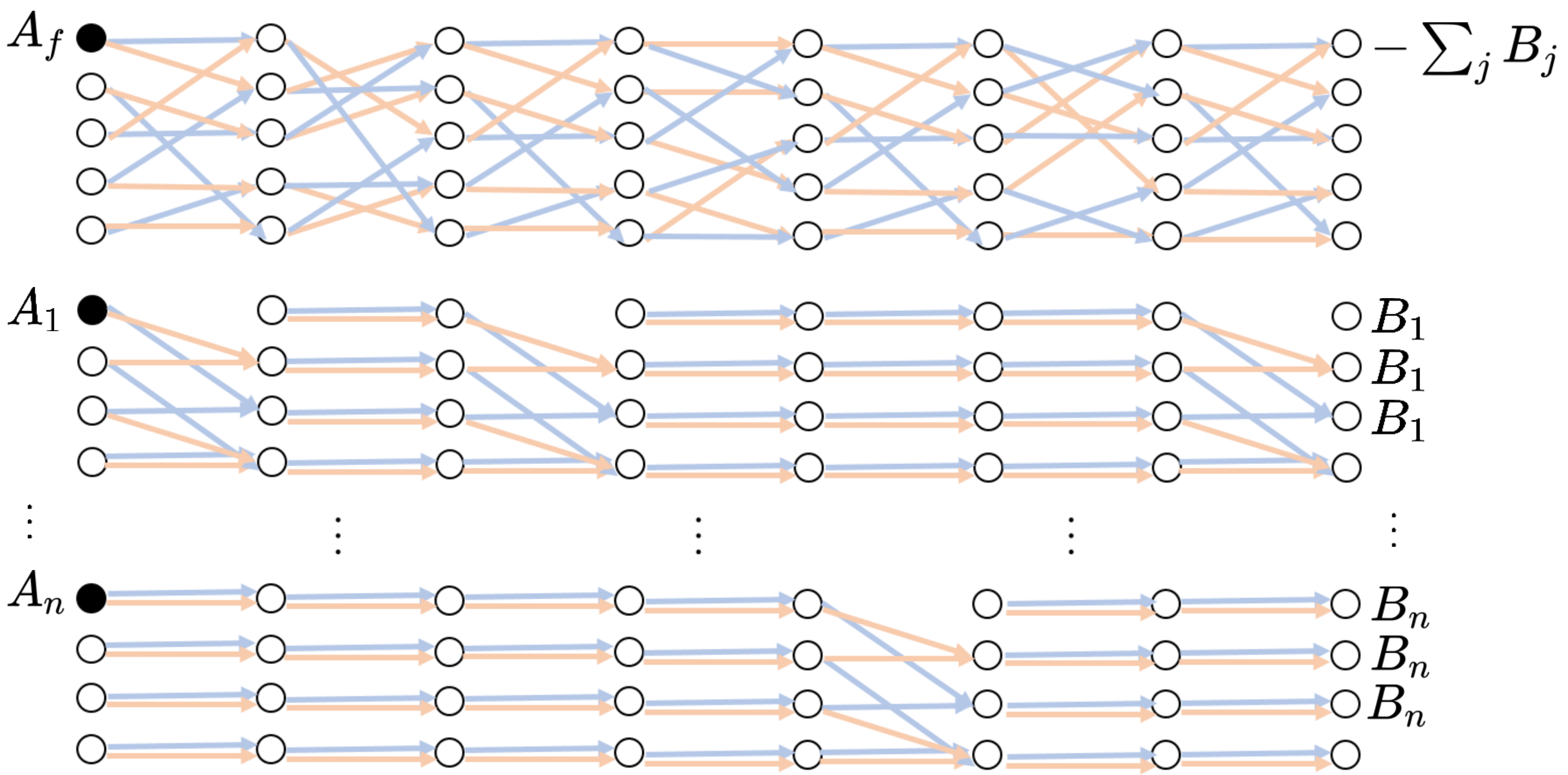


$sA + e, \quad \{K_0^i, K_1^i\}_i$

Standard Analysis Steps:

1. LWE w.r.t. i'th level is hard.
2. Simulate K_0^i, K_1^i without a trapdoor.
3. Generate (i-1)'th level without a trapdoor.
4. LWE w.r.t. (i-1)'th level is hard.

Security Analysis



LWE with respect to the last level is not hard since the matrices are correlated. However, correlated matrices **cannot** be accessed with the same LWE secret.

Security Analysis

Define a designated LWE experiment:

$s, \{S_0^i, S_1^i\}, \{B_j\}$

$$\xleftarrow{\quad x \quad}$$

$$\longrightarrow$$

$$\left\{ s \prod_i S^i_{x_i} B_j \;:\; x \text{ consistent at } j \right\}_{j \in [n]}$$

$$s \prod_i S^i_{x_i} B_f \quad \text{if} \quad BP_f(x) = 1$$

Within natural barriers that were discussed in [GSW13, GLW14].

We show the hardness of this experiment via $2^{poly(n)}$ reductions to standard LWE.

Security Analysis

WE security game $\longrightarrow$ designated LWE experiment

A new assumption [Wee22,Tsa22]**:**

Let $A, B \in \mathbb{Z}_q^{n \times m}$ and $K \leftarrow A^{td}(B)$

LWE w.r.t. $[A]$ given $aux = K$	is as hard as	LWE w.r.t. $[A \vert B]$

Security Analysis - Summary