Structural Lower Bounds on Black-Box Constructions of Pseudorandom Functions

Amos Beimel	(Ben-Gurion University)
Tal Malkin	(Columbia University)
Noam Mazor	(Tel Aviv University)

The Main Question

The Main Question

What is the complexity of black-box PRF construction from PRGs?

The Main Question

What is the complexity of black-box PRF construction from PRGs?

Pseudorandom Generators (PRGs):

$G: \{0,1\}^n \rightarrow \{0,1\}^{2n}$ such that $G(\mathrm{U_n})$ is indistinguishable from $\mathrm{U_{2n}}$.

The Main Question

What is the complexity of black-box PRF construction from PRGs?

Pseudorandom Generators (PRGs):

$G: \{0,1\}^n \to \{0,1\}^{2n}$ such that $G(\mathrm{U_n})$ is indistinguishable from $\mathrm{U_{2n}}$.

Pseudorandom Functions (PRFs):

$$F = \left\{F_k: \{0,1\}^n \to \{0,1\}\right\}_{k \in \{0,1\}^\lambda}$$

The Main Question

What is the complexity of black-box PRF construction from PRGs?

Pseudorandom Generators (PRGs):

$G: \{0,1\}^n \to \{0,1\}^{2n}$ such that $G(\mathrm{U_n})$ is indistinguishable from $\mathrm{U_{2n}}$.

Pseudorandom Functions (PRFs):

$$F = \left\{F_k: \{0,1\}^n \to \{0,1\}\right\}_{k \in \{0,1\}^\lambda}$$

such that F_k is indistinguishable from a random function.

The Main Question

What is the complexity of black-box PRF construction from PRGs?

Pseudorandom Generators (PRGs):

$G: \{0,1\}^n \to \{0,1\}^{2n}$ such that $G(\mathrm{U_n})$ is indistinguishable from $\mathrm{U_{2n}}$.

Pseudorandom Functions (PRFs):

$$F = \left\{F_k: \{0,1\}^n \to \{0,1\}\right\}_{k \in \{0,1\}^\lambda}$$

such that F_k is indistinguishable from a random function.

$$\left| \Pr_{k \leftarrow \{0,1\}^\lambda}[A^{F_k} = 1] - \Pr_{\Pi \leftarrow \{\pi:\{0,1\}^n \to \{0,1\}\}}[A^{\Pi} = 1] \right| \leq negl(n)$$

The Main Question

What is the complexity of black-box PRF construction from PRGs?

The Main Question

What is the complexity of black-box PRF construction from PRGs?

- How many calls to the PRG are needed to evaluate F_k on an input x?

The Main Question

What is the complexity of black-box PRF construction from PRGs?

- How many calls to the PRG are needed to evaluate F_k on an input x?

- [Goldreich-Goldwasser-Micali' 84]: Construction with n adaptive calls

The Main Question

What is the complexity of black-box PRF construction from PRGs?

- How many calls to the PRG are needed to evaluate F_k on an input x?

- [Goldreich-Goldwasser-Micali' 84]: Construction with n adaptive calls
- [Levin' 87]: Levin's trick: Construction with $\omega(\log n)$ adaptive calls

The Main Question

What is the complexity of black-box PRF construction from PRGs?

- How many calls to the PRG are needed to evaluate F_k on an input x?

- [Goldreich-Goldwasser-Micali' 84]: Construction with n adaptive calls
- [Levin' 87]: Levin's trick: Construction with $\omega(\log n)$ adaptive calls

<u>Question</u>: Can we do better?

The Main Question

What is the complexity of black-box PRF construction from PRGs?

- How many calls to the PRG are needed to evaluate F_k on an input x?

- [Goldreich-Goldwasser-Micali' 84]: Construction with n adaptive calls
- [Levin' 87]: Levin's trick: Construction with $\omega(\log n)$ adaptive calls

Question: Can we do better?

- [Naor-Reingold '99, Banerjee-Peikert-Rosen '12]: Construction of PRF in $\mathbf{NC}^1$ based on DDH/lattices

Why Should We Care?

Why Should We Care?

- Fundamental question in Crypto

Why Should We Care?

- Fundamental question in Crypto
- Efficiency of many primitives

Why Should We Care?

- Fundamental question in Crypto
- Efficiency of many primitives
- Important beyond Crypto:

Why Should We Care?

- Fundamental question in Crypto
- Efficiency of many primitives
- Important beyond Crypto:
  - Natural Proofs [Razborov-Rudich]
  - Barriers on circuits lower bounds
  - Lower bounds on learning

Why Should We Care?

- Fundamental question in Crypto
- Efficiency of many primitives
- Important beyond Crypto:
  - Natural Proofs [Razborov-Rudich]
  - Barriers on circuits lower bounds
  - Lower bounds on learning

- PRFs from PRGs are much less understood than PRGs from OWFs

  [Gennero-Gertner-Katz-Trevisan, Holenstien]

Even Simpler Open Question

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k,x)),k,x)$$

Even Simpler Open Question

<u>Question</u>: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k, x)), k, x)$$

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k, x)), k, x)$$

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k,x)), k, x)$$

Thm [Miles-Viola]: There is no black-box one call bit-projection construction

$$F_k(x) = G\left(S_{k,x}\right)_i \text{ for } i = L(k,x) \in [2n]$$

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k,x)), k, x)$$

Thm [Miles-Viola]: There is no black-box one call bit-projection construction

$$F_k(x) = G\left(S_{k,x}\right)_i \text{ for } i = L(k,x) \in [2n]$$

$P(y,k,x) = y_{L(k,x)}$, where $L(k,x) \in [2n]$

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k,x)), k, x)$$

Thm [Miles-Viola]: There is no black-box one call bit-projection construction

$$F_k(x) = G\left(S_{k,x}\right)_i \text{ for } i = L(k,x) \in [2n]$$

$P(y,k,x) = y_{L(k,x)}$, where $L(k,x) \in [2n]$

Warm-up Thm [This work]: There is no black-box one call constructions with

$$P(y,k,x) = P\big(y, L(k,x)\big) \text{ for } |L(k,x)| = O(\log n)$$

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k,x)), k, x)$$

TMI Thm [This work]: Let L be a function with $O(\log n)$ output length, and P be a function with $n - \omega(\log n)$ output length. Let $O_{k,x}(s) = P(G(s), L(k,x))$.

Then there is no black-box constructions

$$F_k^G(x) = A^{O_{k,x}}(k,x)$$

For any oracle-aided algorithm A.

Even Simpler Open Question

Question: Can we rule out black-box PRF constructions that make one call to the PRG?

$$F_k(x) = P(G(S(k,x)),k,x)$$

TMI Thm [This work]: Let L be a function with $O(\log n)$ output length, and P be a function with $n - \omega(\log n)$ output length. Let $O_{k,x}(s) = P(G(s), L(k,x))$.

Then there is no black-box constructions

$$F_k^G(x) = A^{O_{k,x}}(k,x)$$

For any oracle-aided algorithm A.

To the best of our knowledge, one-call black-box PRF construction is still possible

Our results

Our results

We show that for a large class of constructions that naturally generalize GGM, GGM is essentially optimal.

Our results

We show that for a large class of constructions that naturally generalize GGM, GGM is essentially optimal.

- Some of the choices in the GGM constructions are optimal

Our results

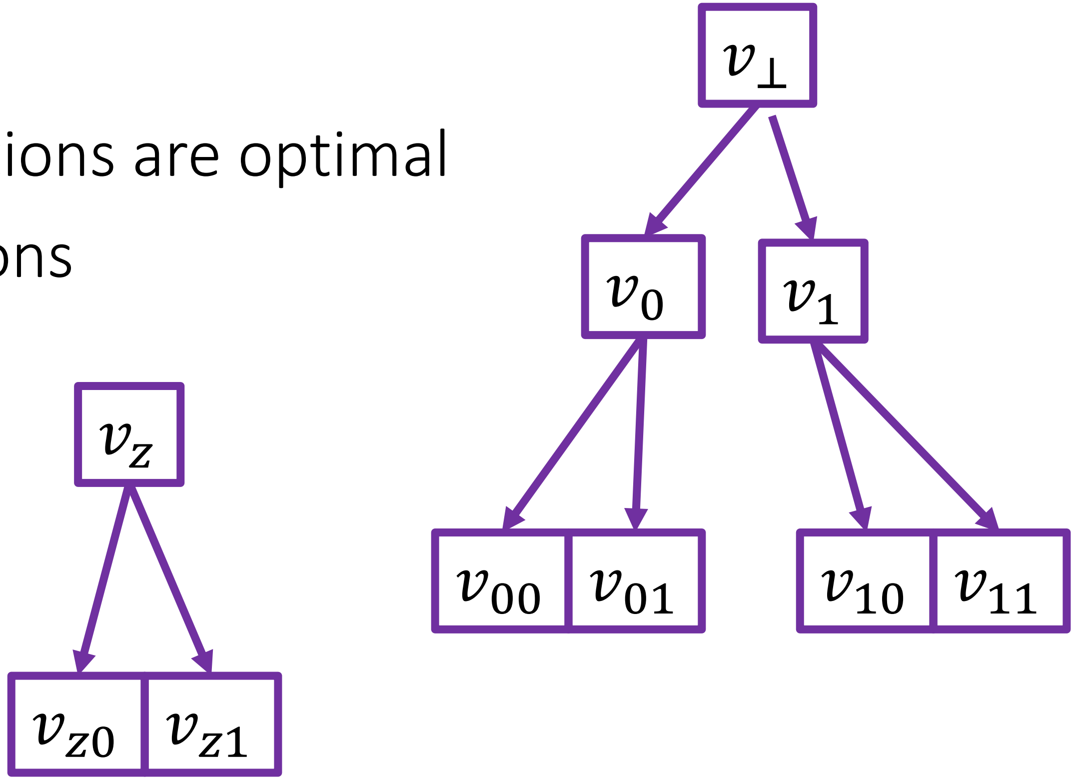
We show that for a large class of constructions that naturally generalize GGM, GGM is essentially optimal.

- Some of the choices in the GGM constructions are optimal
- “Natural generalization” = Tree constructions

Our results

We show that for a large class of constructions that naturally generalize GGM, GGM is essentially optimal.

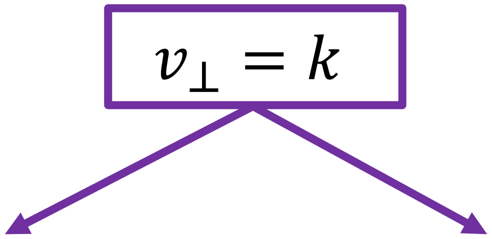
- Some of the choices in the GGM constructions are optimal
- “Natural generalization” = Tree constructions



The GGM Construction

Thm[GGM]: PRG ⇒ PRF

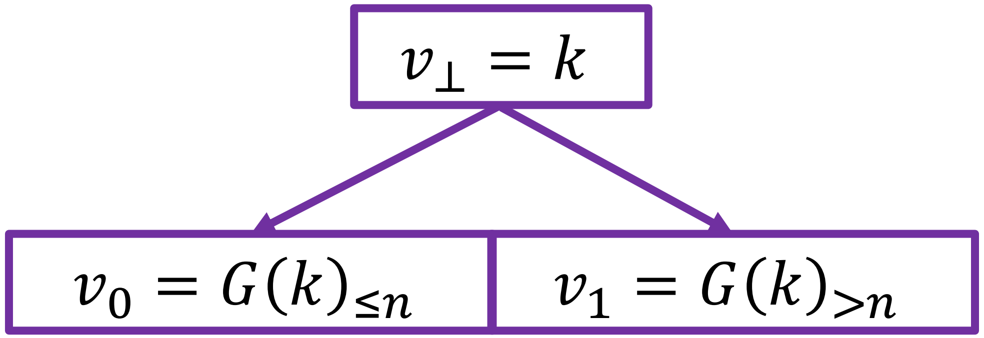
$G: \{0,1\}^n \to \{0,1\}^{2n}$



The GGM Construction

Thm[GGM]: PRG ⇒ PRF

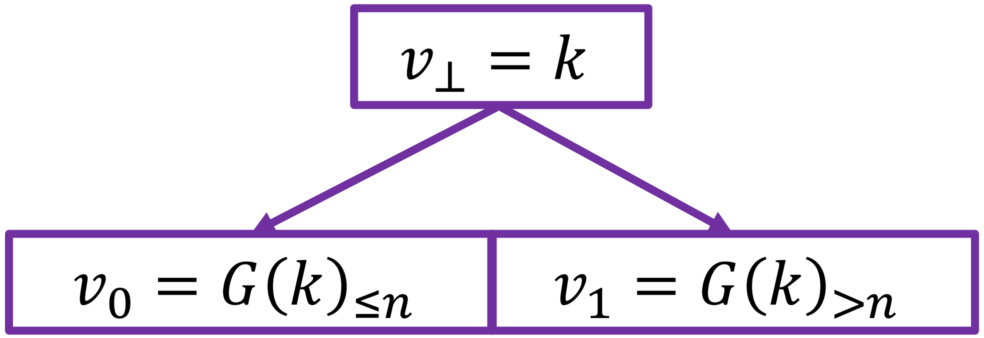
$G: \{0,1\}^n \to \{0,1\}^{2n}$

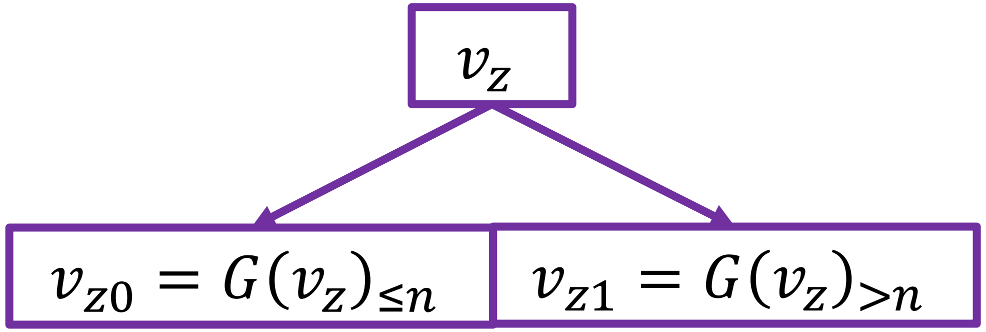


The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

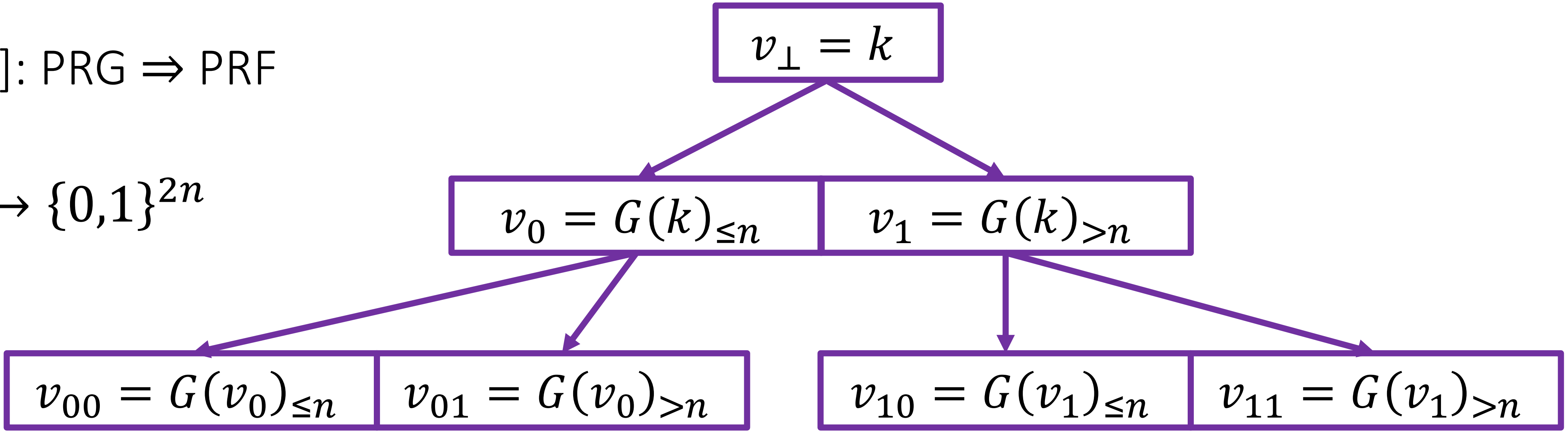


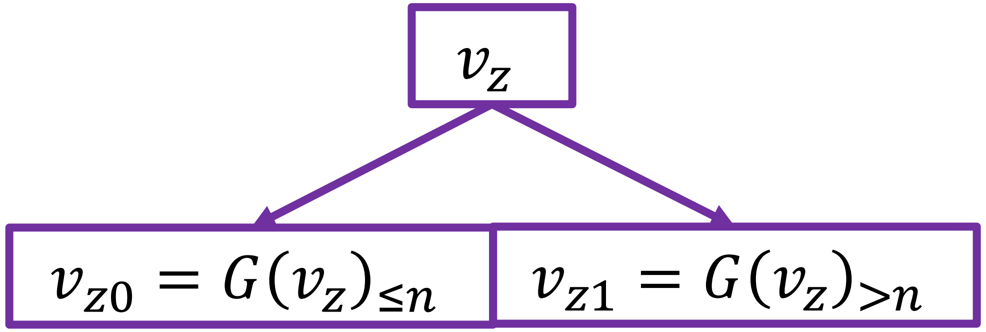


The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

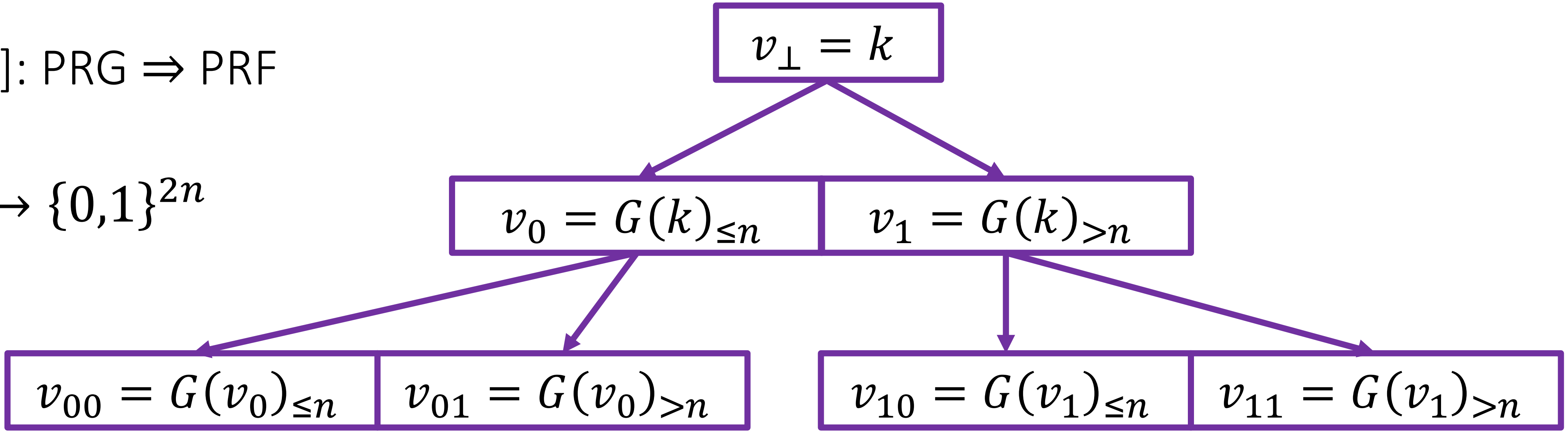




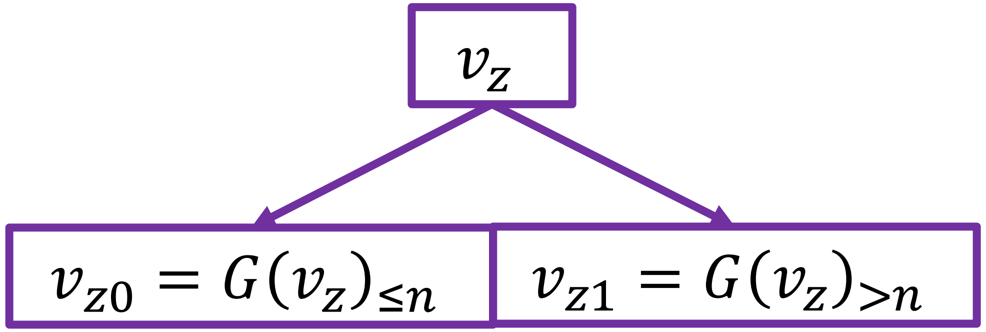
The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \rightarrow \{0,1\}^{2n}$



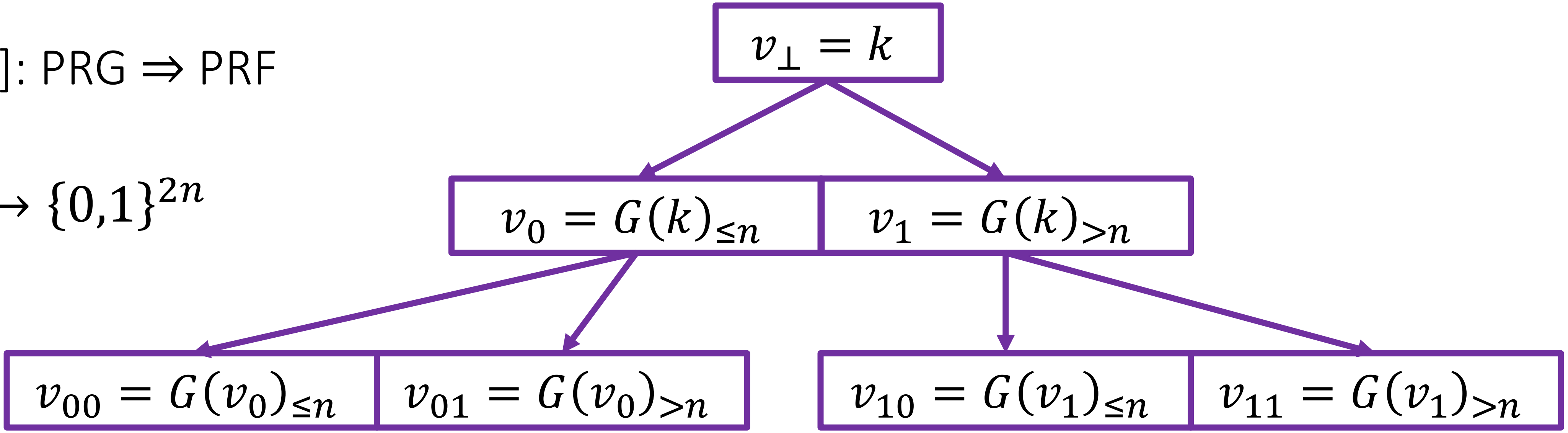
$F_k(x) = v_x$



The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

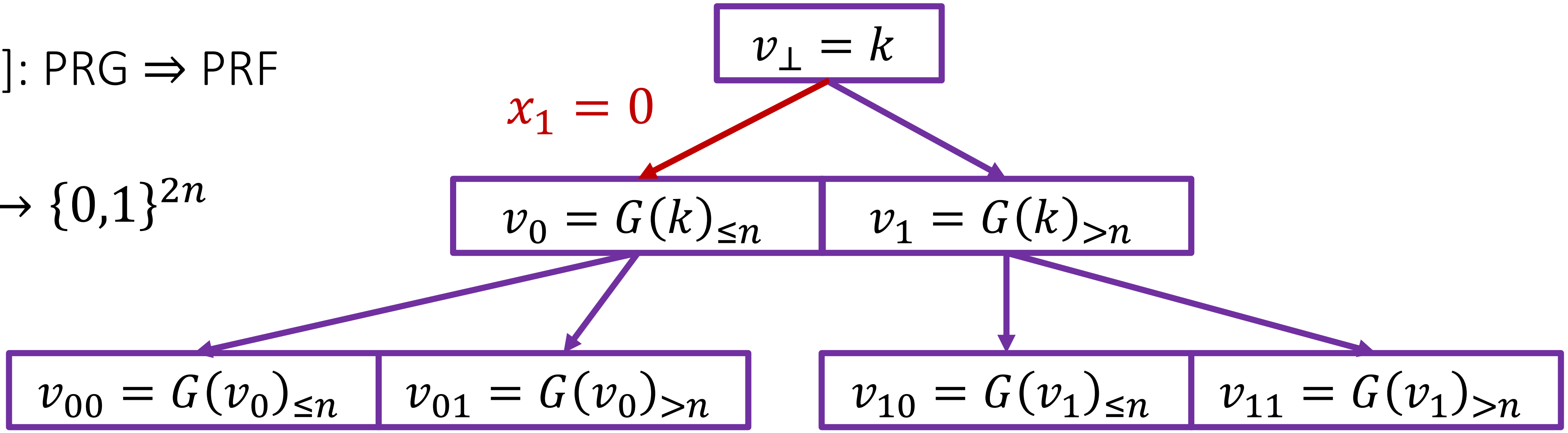


$F_k(01) = v_{01}$

The GGM Construction

<u>Thm</u>[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

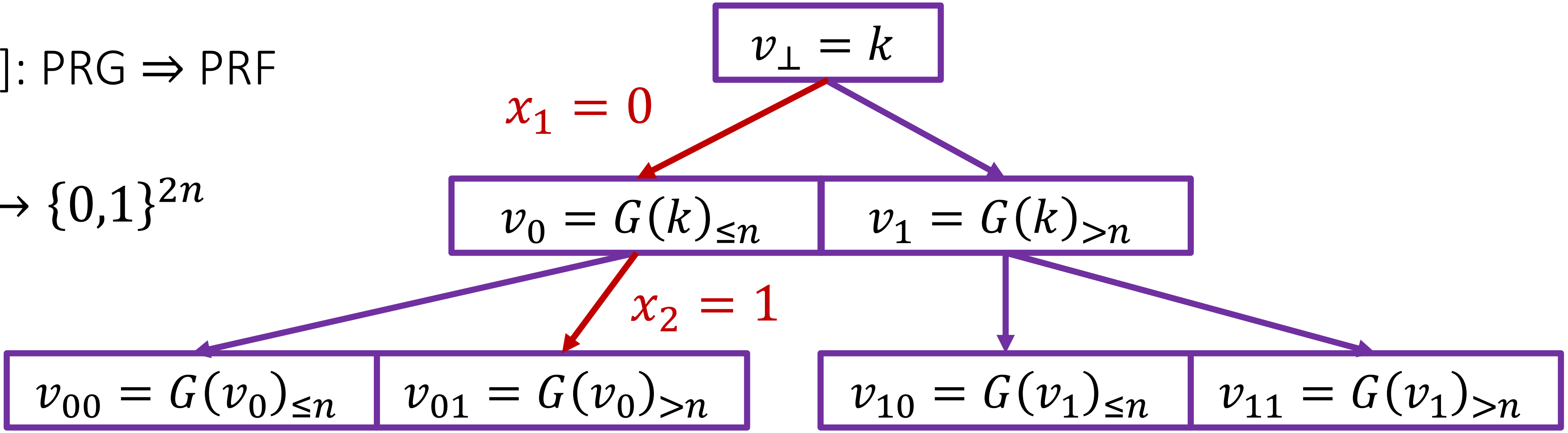


$F_k(01) = v_{01}$

The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

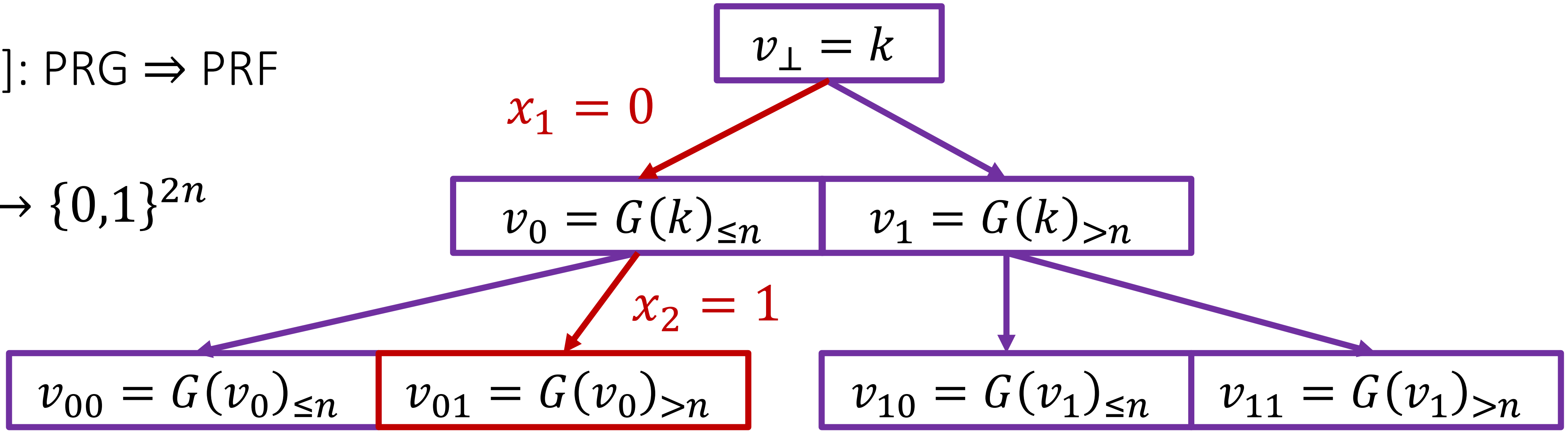


$F_k(01) = v_{01}$

The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

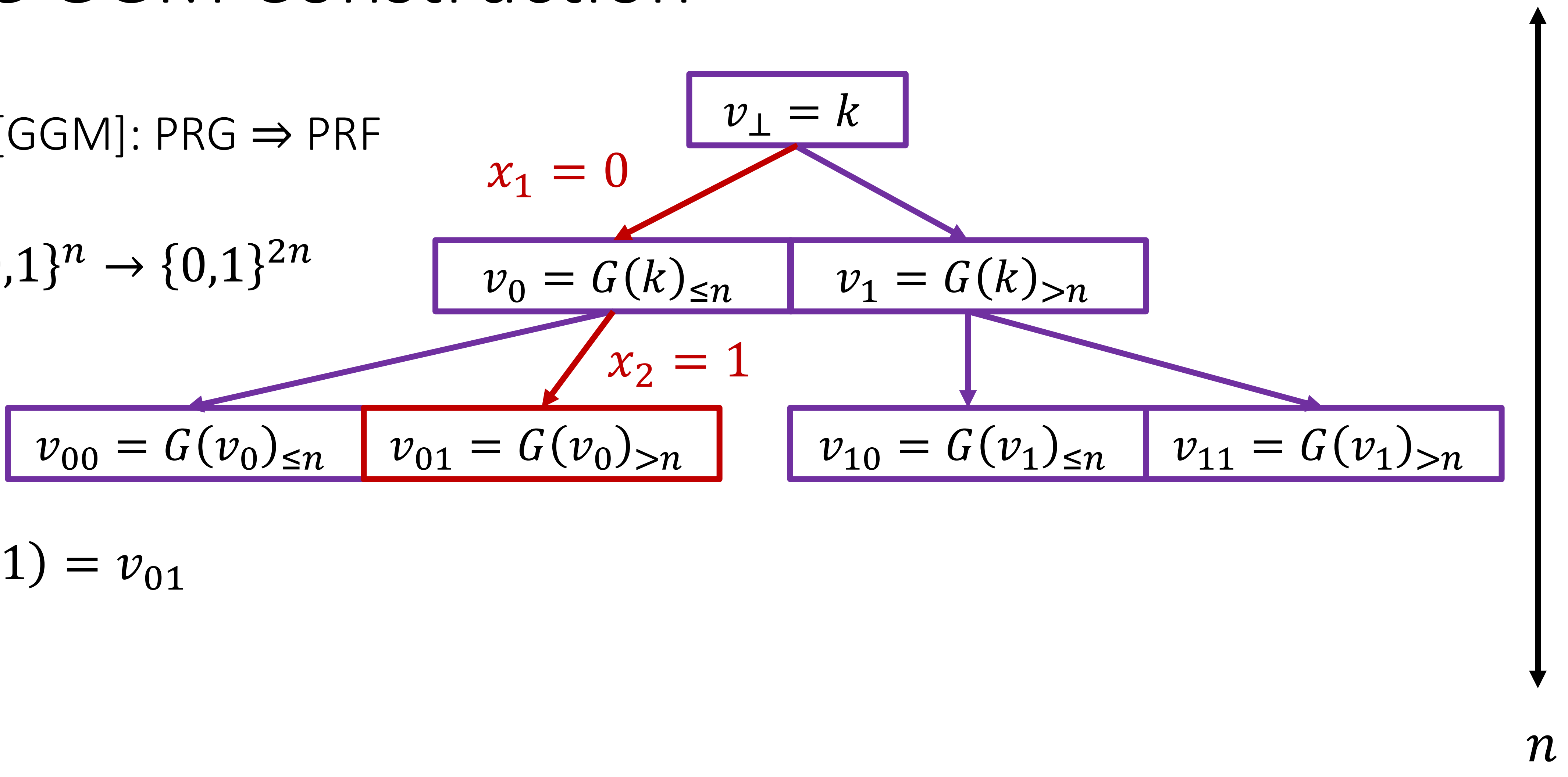


$F_k(01) = v_{01}$

The GGM Construction

Thm[GGM]: PRG ⇒ PRF

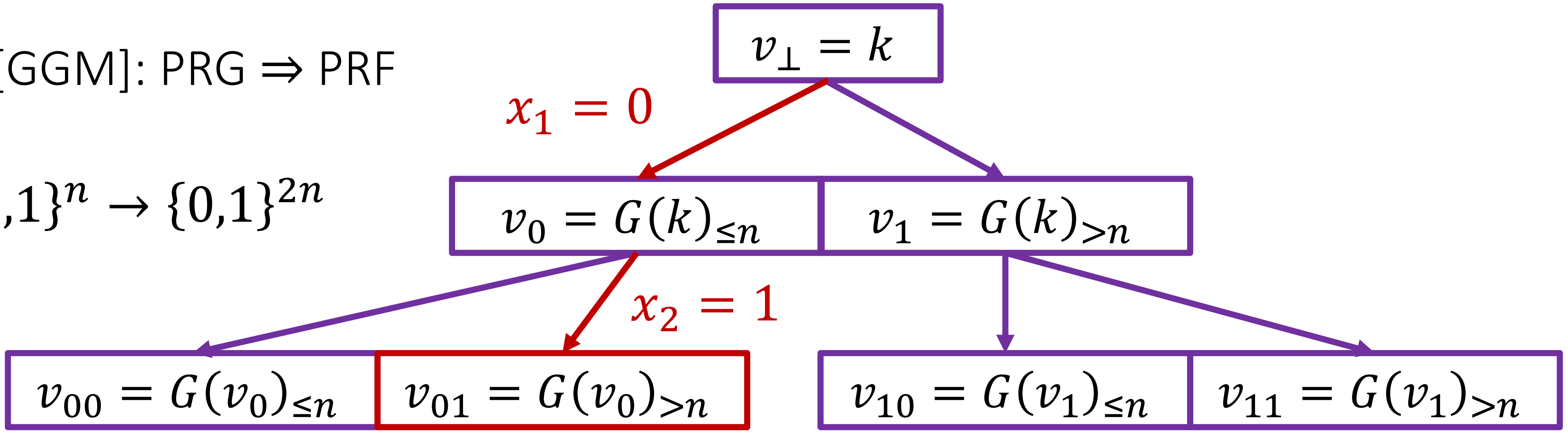
$G: \{0,1\}^n \to \{0,1\}^{2n}$

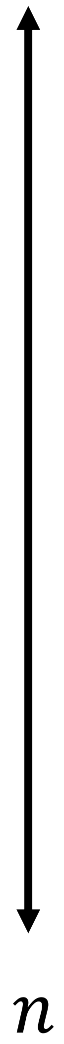


$F_k(01) = v_{01}$

The GGM Construction

Thm[GGM]: PRG ⇒ PRF





$G: \{0,1\}^n \to \{0,1\}^{2n}$

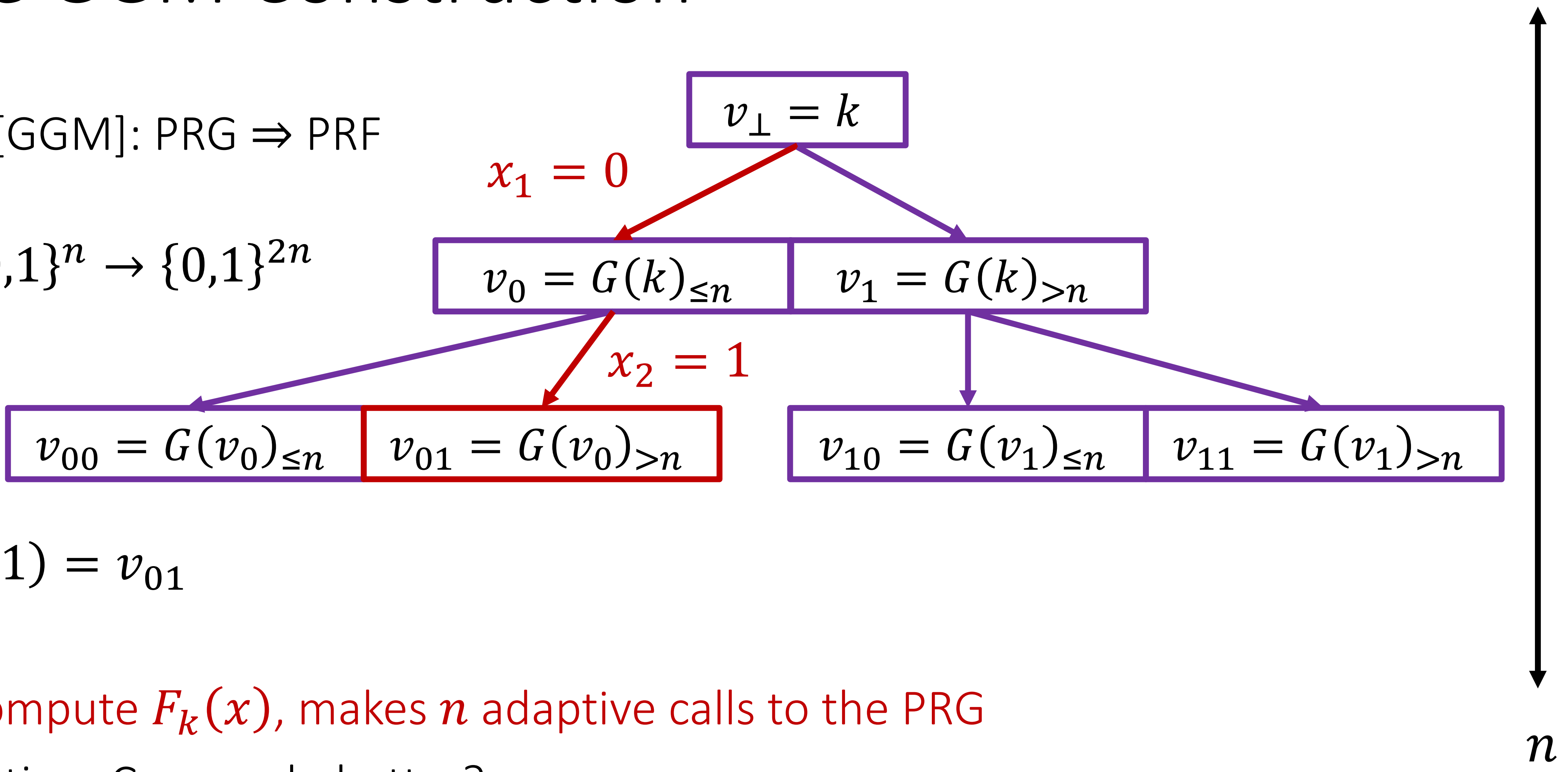
$F_k(01) = v_{01}$

To compute $F_k(x)$, makes n adaptive calls to the PRG

The GGM Construction

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$



$F_k(01) = v_{01}$

To compute $F_k(x)$, makes n adaptive calls to the PRG

Question: Can we do better?

Levin’s Trick (Domain Extention)

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \rightarrow \{0,1\}^{2n}$

Levin's Trick (Domain Extention)

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

Let $h: \{0,1\}^n \to \{0,1\}^{\omega(\log n)}$ is a two-universal hash function

Levin’s Trick (Domain Extention)

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$

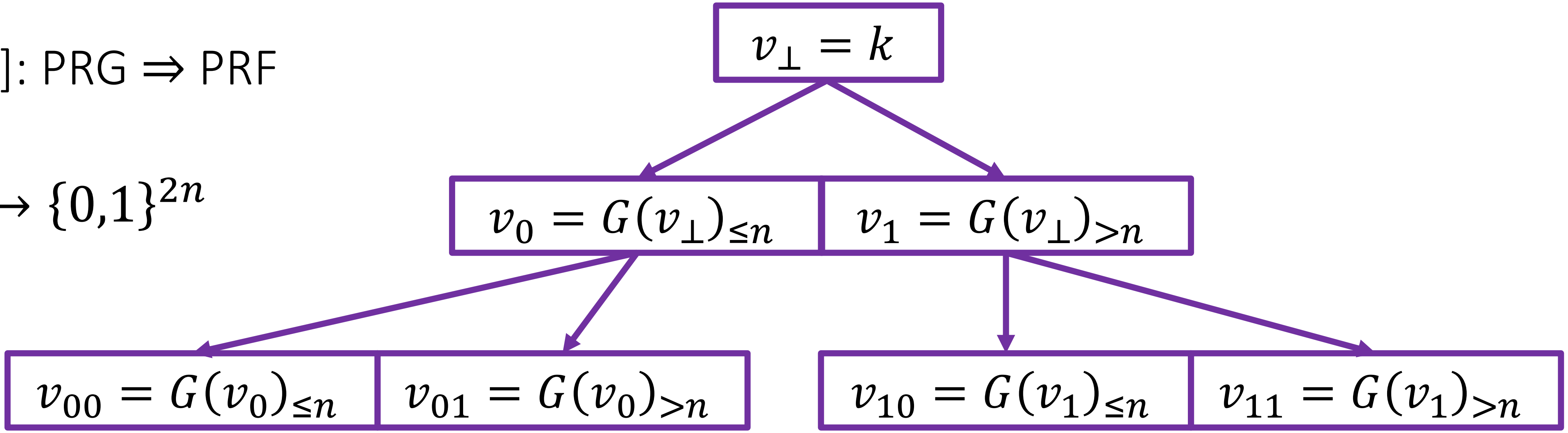
Let $h: \{0,1\}^n \to \{0,1\}^{\omega(\log n)}$ is a two-universal hash function

$$F_{h,k}(x) = F_k\big(h(x)\big) = \ v_{h(x)}$$

Levin's Trick (Domain Extention)

<u>Thm</u>[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$



Let $h: \{0,1\}^n \to \{0,1\}^{\omega(\log n)}$ is a two-universal hash function

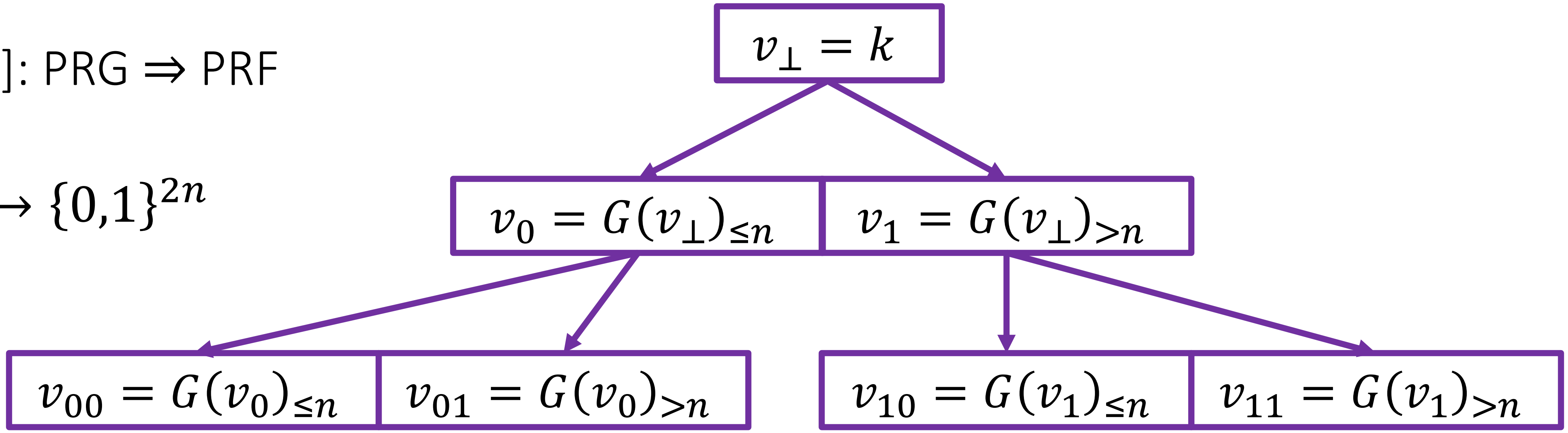
$$F_{h,k}(x) = F_k\big(h(x)\big) = v_{h(x)}$$

$\omega(\log n)$

Levin's Trick (Domain Extention)

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \to \{0,1\}^{2n}$



Let $h: \{0,1\}^n \to \{0,1\}^{\omega(\log n)}$ is a two-universal hash function

$$F_{h,k}(x) = F_k\big(h(x)\big) = v_{h(x)}$$

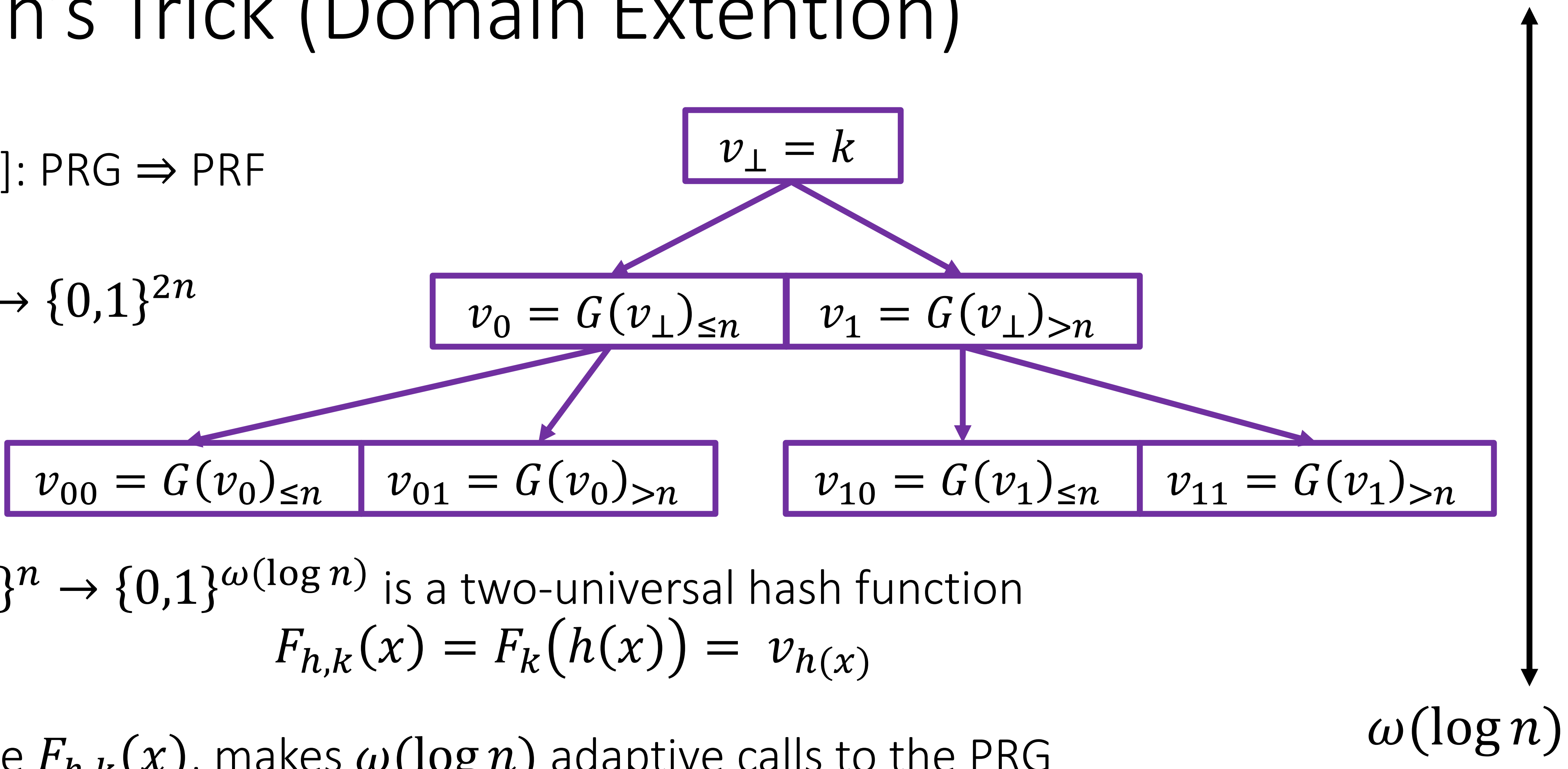
$\omega(\log n)$

To compute $F_{h,k}(x)$, makes $\omega(\log n)$ adaptive calls to the PRG

Levin's Trick (Domain Extention)

Thm[GGM]: PRG ⇒ PRF

$G: \{0,1\}^n \rightarrow \{0,1\}^{2n}$



Let $h: \{0,1\}^n \rightarrow \{0,1\}^{\omega(\log n)}$ is a two-universal hash function

$$F_{h,k}(x) = F_k\big(h(x)\big) = v_{h(x)}$$

To compute $F_{h,k}(x)$, makes $\omega(\log n)$ adaptive calls to the PRG

Question: Can we do better?

Tree constructions

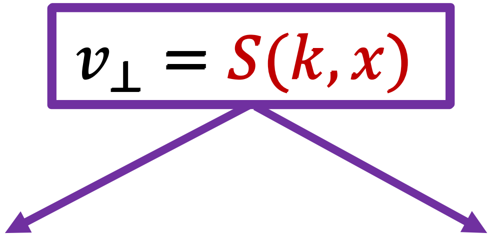
Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

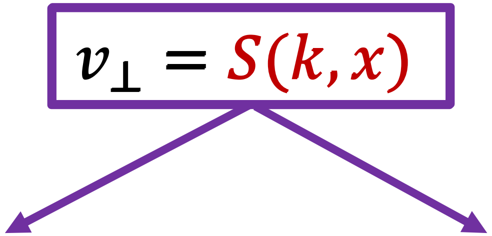
1. The root of the tree can be arbitrary function of k, x



Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

1. The root of the tree can be arbitrary function of k, x



In GGM, $S(k, x) = k$

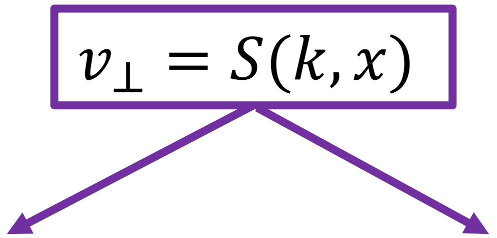
Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

2. The label of the children

can be arbitrary function

of $G(v_z)$



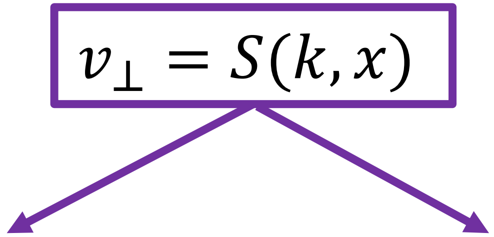
Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

2. The label of the children

can be arbitrary function

of $G(v_z)$



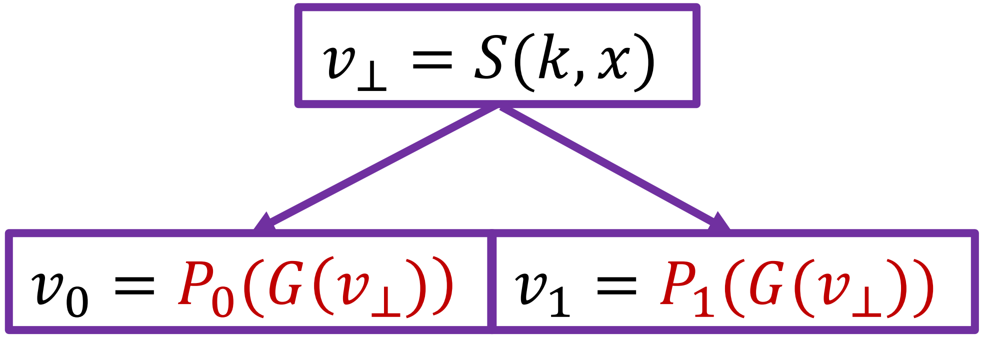
Tree constructions

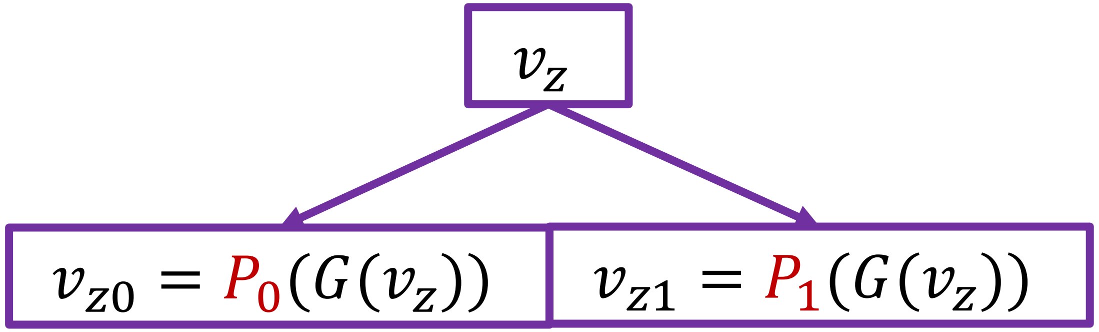
We define Tree-Constructions - a generalization of the GGM's construction in three ways:

2. The label of the children

can be arbitrary function

of $G(v_z)$





Tree constructions

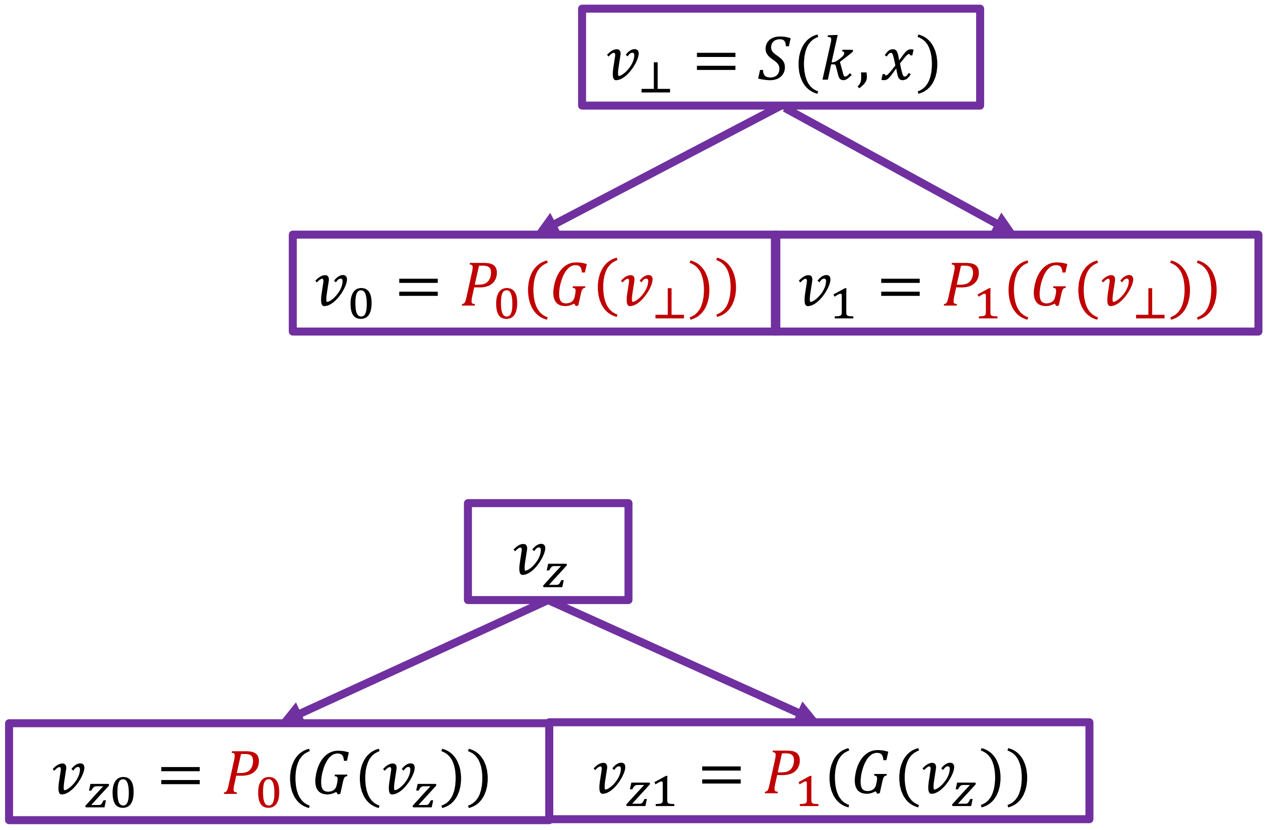
We define Tree-Constructions - a generalization of the GGM's construction in three ways:

2. The label of the children

can be arbitrary function

of $G(v_z)$

(can be different in each level)



Tree constructions

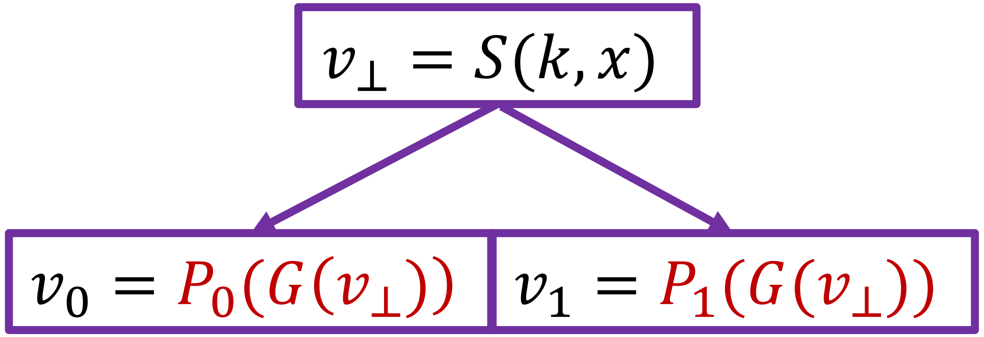
We define Tree-Constructions - a generalization of the GGM's construction in three ways:

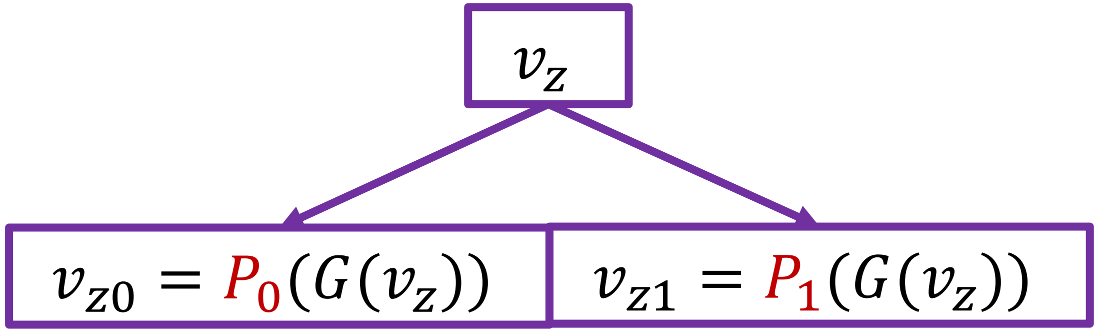
2. The label of the children

can be arbitrary function

of $G(v_z)$

(can be different in each level)





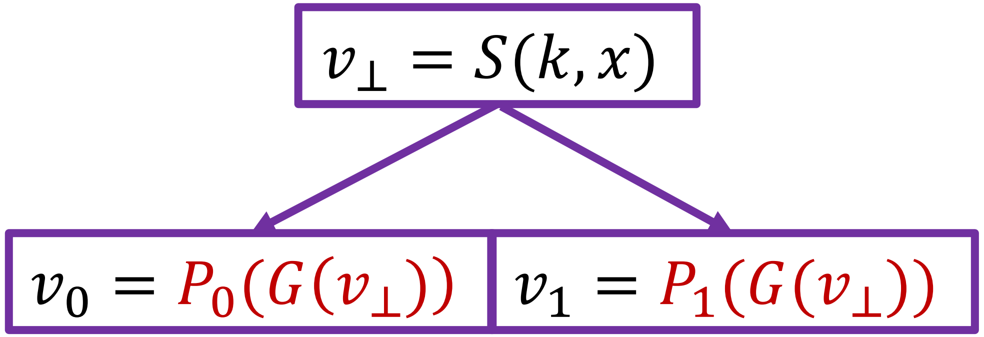
In GGM, $P_0(y) = y_{\leq n}$

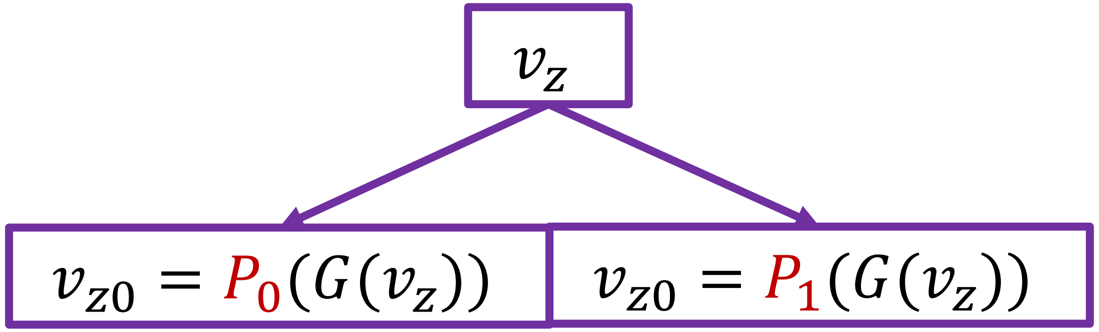
. $P_1(y) = y_{>n}$

Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

3. The choice of the path can be arbitrary function of k, x

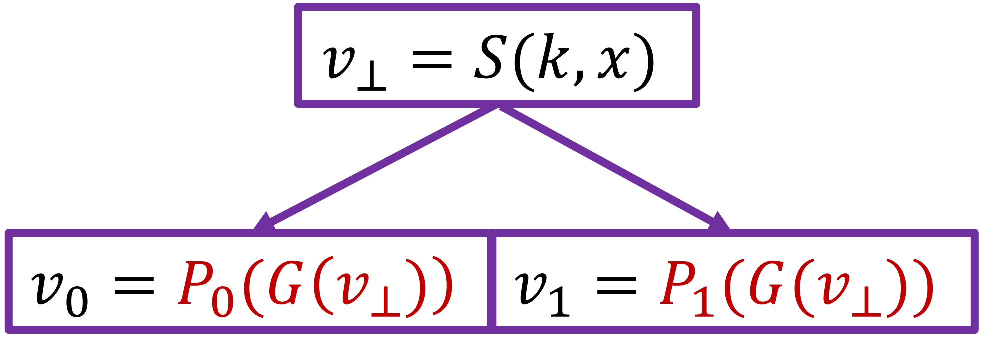




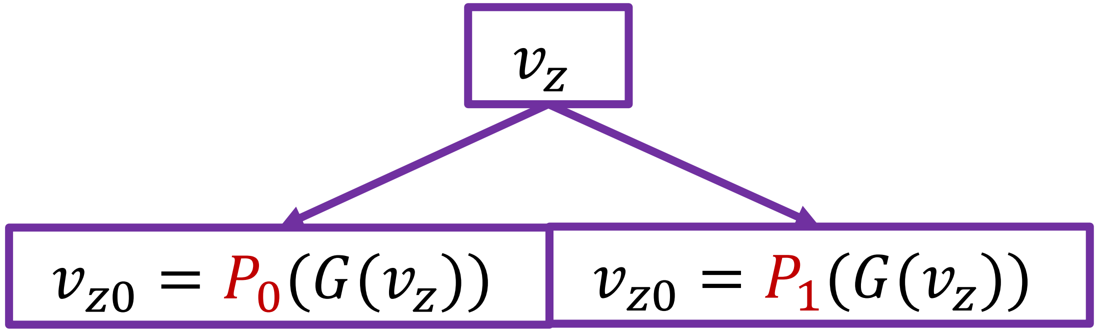
Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

3. The choice of the path can be arbitrary function of k, x



$$F_k(x) = v_{L(k,x)}$$



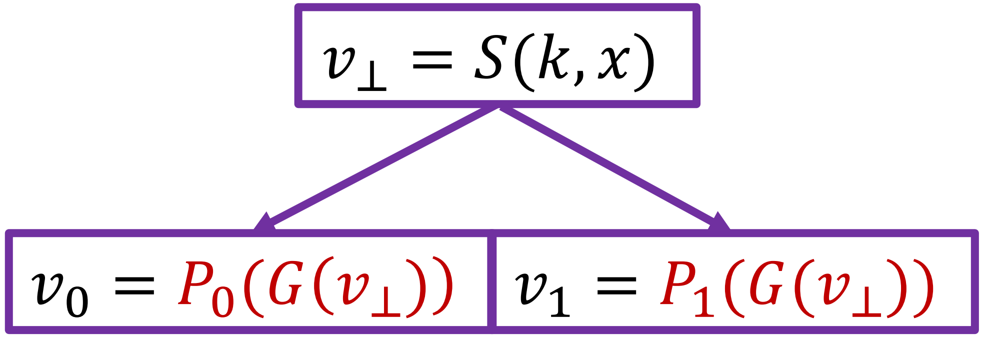
Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

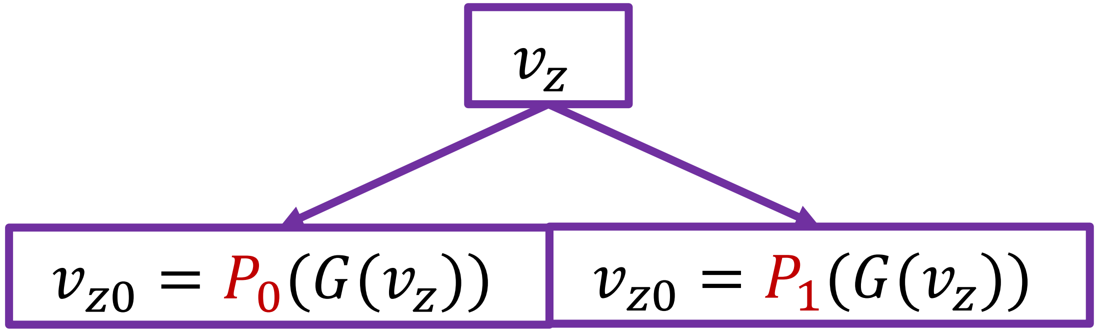
3. The choice of the path can be

arbitrary function of

k, x



$F_k(x) = v_{L(k,x)}$



In GGM, $L(k, x) = x$ [or $h(x)$]

Tree constructions

We define Tree-Constructions - a generalization of the GGM's construction in three ways:

1. The root of the tree can be an arbitrary function of k, x

2. The label of the children can be an arbitrary function of $G(v_z)$

3. The choice of the path can be an arbitrary function of k, x

Main Result: Lower Bounds on Tree Constructions

Main Result: Lower Bounds on Tree Constructions

Thm [This work]: There is no fully black-box tree constructions with depth of

$$\log n \; - \operatorname{loglog} n$$

Main Result: Lower Bounds on Tree Constructions

Thm [This work]: There is no fully black-box tree constructions with depth of

$$\log n \ - \text{loglog}\, n$$

- GGM is fully black-box tree constructions with depth $\omega(\log n)$

Main Result: Lower Bounds on Tree Constructions

Thm [This work]: There is no fully black-box tree constructions with depth of

$$\log n \ - \log\log n$$

- GGM is fully black-box tree constructions with depth $\omega(\log n)$
- For any constant-degree tree

Main Result: Lower Bounds on Tree Constructions

Thm [This work]: There is no fully black-box tree constructions with depth of

$$\log n \; - \operatorname{loglog} n$$

- GGM is fully black-box tree constructions with depth $\omega(\log n)$
- For any constant-degree tree
- For any stretch of the PRG

Proof Overview

Proof Overview

We use oracle methodology [Impagliazzo-Rudich, Gertner-Malkin-Reingold]

Proof Overview

We use oracle methodology [Impagliazzo-Rudich, Gertner-Malkin-Reingold]

For every low depth tree construction, we show an oracle with respect to which:

Proof Overview

We use oracle methodology [Impagliazzo-Rudich, Gertner-Malkin-Reingold]

For every low depth tree construction, we show an oracle with respect to which:

- There is a secure PRG G

Proof Overview

We use oracle methodology [Impagliazzo-Rudich, Gertner-Malkin-Reingold]

For every low depth tree construction, we show an oracle with respect to which:

- There is a secure PRG G
- There is an efficient algorithm $Break$ that breaks the PRF implementation using G

Proof Overview

Proof Overview

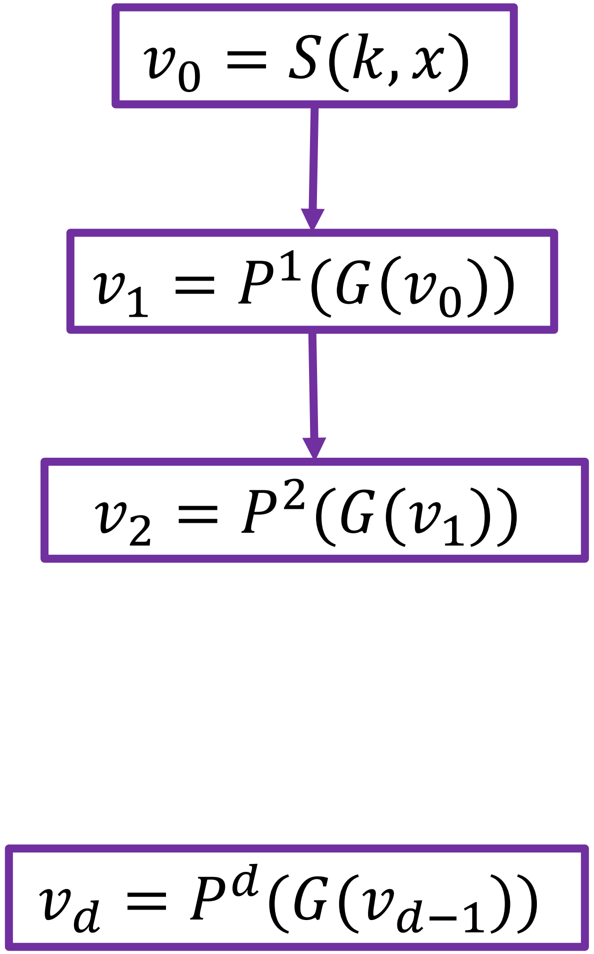
- Our main contribution is a technique to deal with adaptive calls

Proof Overview

- Our main contribution is a technique to deal with adaptive calls

- We use ideas from Miles-Viola to show it is enough

to exclude sequential constructions



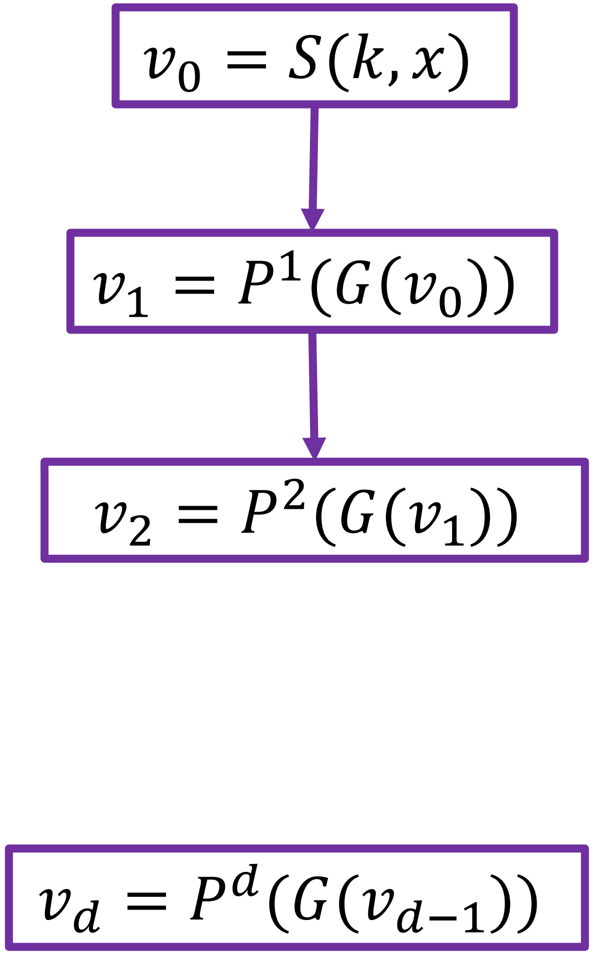
Proof Overview

- Our main contribution is a technique to deal with adaptive calls

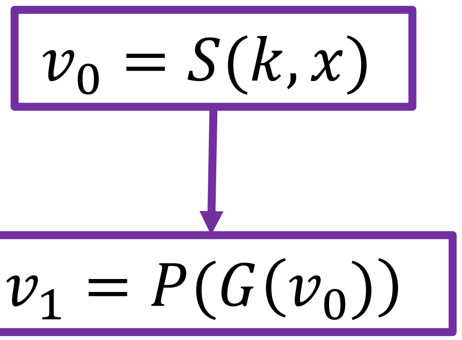
- We use ideas from Miles-Viola to show it is enough

to exclude sequential constructions

- We show that there are no such sequential constructions

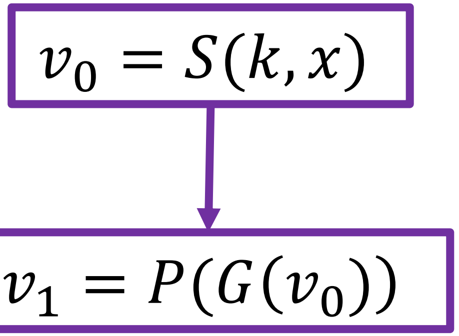


Depth-One Sequential Construction



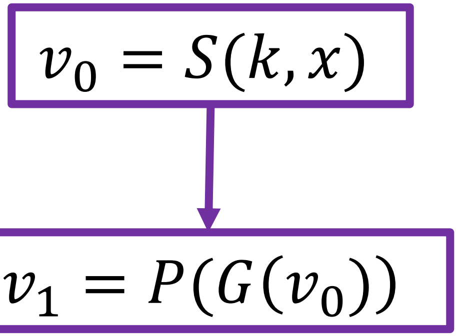
Depth-One Sequential Construction

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



Depth-One Sequential Construction

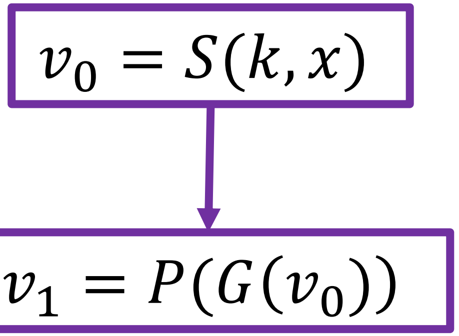
$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- We choose (a family of) PRGs G such that $F_k(x)$ can be computed from k, x without calling to G

Depth-One Sequential Construction

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$

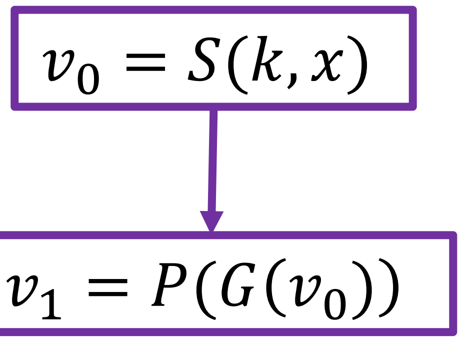


- We choose (a family of) PRGs G such that $F_k(x)$ can be computed from k, x without calling to G

$\Rightarrow$ Can break the security of F without breaking G

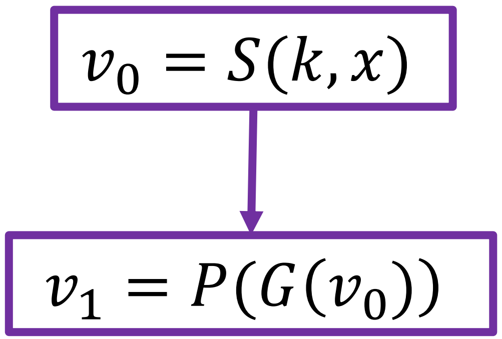
Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



Constructing the PRG

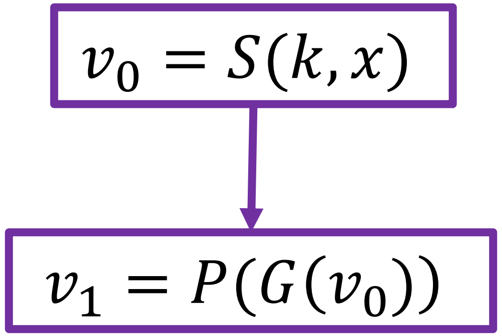
$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$

Constructing the PRG

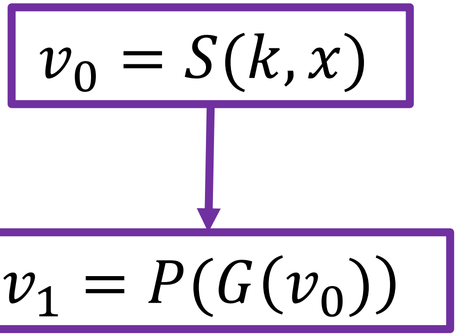
$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation

Constructing the PRG

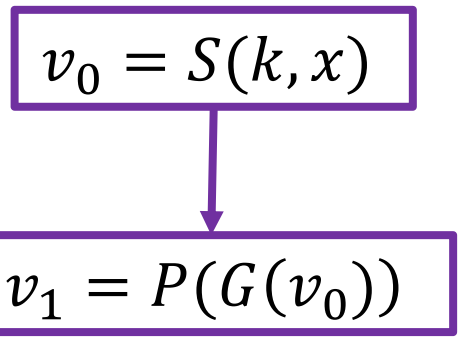
$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$

Constructing the PRG

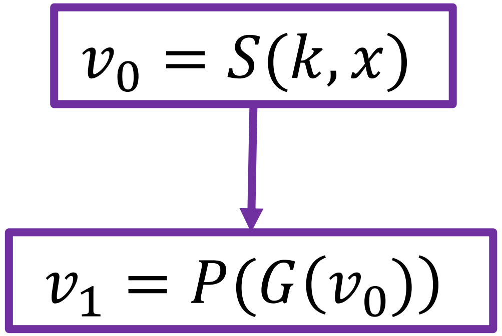
$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $[G(s) = \pi(G'(s))]$

Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$

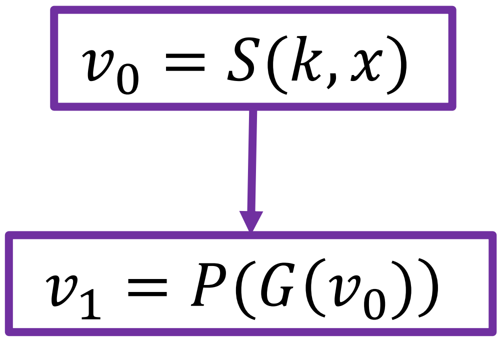


- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $[G(s) = \pi(G'(s))]$

$\Rightarrow$ G is a PRG

Constructing the PRG

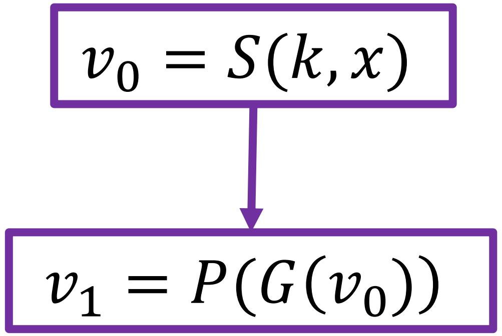
$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $\quad [G(s) = \pi(G'(s))]$

Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $[G(s) = \pi(G'(s))]$

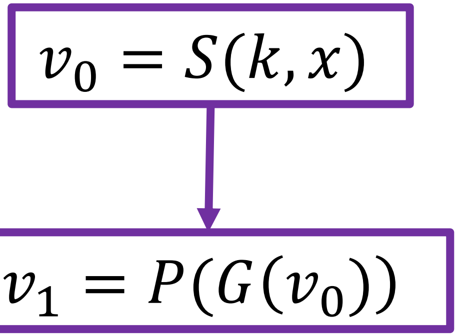
$$F_k(x) = P(G(S(k,x)))_1$$

Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$

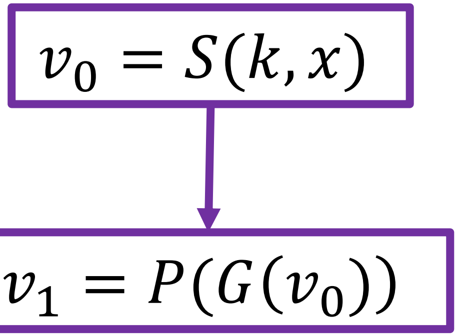
- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $\quad [G(s) = \pi(G'(s))]$

$$F_k(x) = P(G(S(k,x)))_1$$
$$= P(\pi(G'(S(k,x))))_1$$



Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$

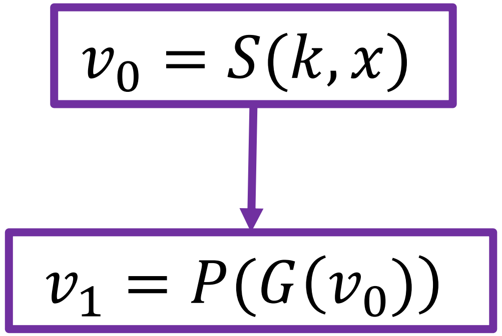


- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $\quad [G(s) = \pi(G'(s))]$

$$\begin{aligned} F_k(x) &= P(G(S(k,x)))_1 \\ &= P(\pi(G'(S(k,x))))_1 \\ &= G'(S(k,x))_1 \end{aligned}$$

Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$



- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $\quad [G(s) = \pi(G'(s))]$

$$\begin{aligned} F_k(x) &= P(G(S(k,x)))_1 \\ &= P(\pi(G'(S(k,x))))_1 \\ &= G'(S(k,x))_1 \\ &= S(k,x)_1 \end{aligned}$$

Constructing the PRG

$$F_k(x) = P\big(G(v_0)\big)_1 = P\Big(G\big(S(k,x)\big)\Big)_1$$

$v_0 = S(k,x)$

↓

$v_1 = P(G(v_0))$

- Let G' be a PRG such that $G'(s)_1 = s_1$
- Assume that P is a permutation
  - Let $\pi = P^{-1}$
  - Let $G = \pi \circ G'$ $[G(s) = \pi(G'(s))]$

$$\begin{aligned} F_k(x) &= P(G(S(k,x)))_1 \\ &= P(\pi(G'(S(k,x))))_1 \\ &= G'(S(k,x))_1 \\ &= S(k,x)_1 \end{aligned}$$

Key point:
- $P\big(\pi(y)\big)_1 = y_1$

where $y = G'(s)$

General Post-Processing

Key point:

- $P\big(\pi(y)\big)_1 = y_1$

where $y = G'(s)$

General Post-Processing

Key point:
- $P\big(\pi(y)\big)_1 = y_1$

where $y = G'(s)$

More generally, we need to find π such that for every i,

$$P(\pi(y))_{\leq i} \text{ can be computed from the first} \approx i \text{ bits of } y$$

General Post-Processing

Key point:
- $P\big(\pi(y)\big)_1 = y_1$

where $y = G'(s)$

More generally, we need to find π such that for every i,

$P(\pi(y))_{\leq i}$ can be computed from the first $\approx i$ bits of y

- For a permutation, $P(\pi(y))_{\leq i} = y_{\leq i}$

General Post-Processing

Key point:

- $P\big(\pi(y)\big)_1 = y_1$

where $y = G'(s)$

More generally, we need to find π such that for every i,

$P(\pi(y))_{\leq i}$ can be computed from the first $\approx i$ bits of y

- For a permutation, $P(\pi(y))_{\leq i} = y_{\leq i}$

When P is a not a permutation?

The Pseudo-Inverse Lemma

Lemma: For any function $P: \{0,1\}^n \rightarrow \{0,1\}^n$ there exists a function $\pi: \{0,1\}^n \rightarrow \{0,1\}^n$ such that:

The Pseudo-Inverse Lemma

Lemma: For any function $P: \{0,1\}^n \to \{0,1\}^n$ there exists a function $\pi: \{0,1\}^n \to \{0,1\}^n$ such that:

1. π is almost a permutation ($\pi(U_n) \approx U_n$)

The Pseudo-Inverse Lemma

Lemma: For any function $P: \{0,1\}^n \to \{0,1\}^n$ there exists a function $\pi: \{0,1\}^n \to \{0,1\}^n$ such that:

1. π is almost a permutation ($\pi(U_n) \approx U_n$)
2. For every $i \in [n]$, $P\big(\pi(y)\big)_{\leq i}$ can be computed from $y_{\leq i + \log^2 n}$

Summary

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n \ - \operatorname{loglog} n$$

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n \ - \operatorname{loglog} n$$

- GGM is of depth $\omega(\log n)$

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n \; - \operatorname{loglog} n$$

- GGM is of depth $\omega(\log n)$

+ More lower bounds

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n \ - \operatorname{loglog} n$$

- GGM is of depth $\omega(\log n)$

+ More lower bounds

Open questions:

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n \ - \operatorname{loglog} n$$

- GGM is of depth $\omega(\log n)$

+ More lower bounds

Open questions:

- What is the complexity of black-box PRF constructions?

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n \ - \operatorname{loglog} n$$

- GGM is of depth $\omega(\log n)$

+ More lower bounds

Open questions:

- What is the complexity of black-box PRF constructions?
- Can we construct one-call PRF?

Summary

Thm [This work]: There is no black-box tree constructions with depth of

$$\log n - \log\log n$$

- GGM is of depth $\omega(\log n)$

+ More lower bounds

Open questions:

- What is the complexity of black-box PRF constructions?
- Can we construct one-call PRF?

Thanks!