Communication against restricted adversaries: between Shannon and Hamming

University of Melbourne

Anand D. Sarwate

Rutgers University / ITSOC DL Program

13 August 2025

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.
- What is the maximum rate (capacity) $\frac{1}{n}\log_2(M)$ such that Bob can decode reliably?

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.
- What is the maximum rate (capacity) $\frac{1}{n} \log_2(M)$ such that Bob can decode reliably?

This problem has been studied to death! What more is there to understand?

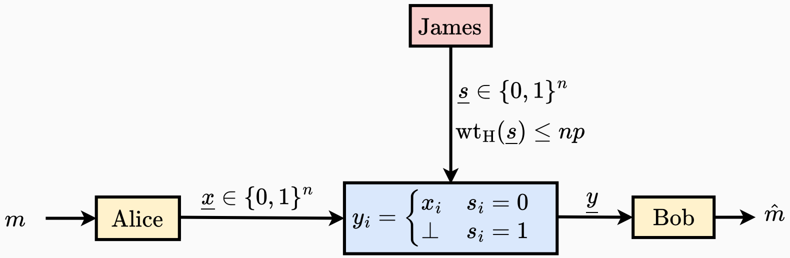
Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.
- What is the maximum rate (capacity) $\frac{1}{n}\log_2(M)$ such that Bob can decode reliably?

This problem has been studied to death! What more is there to understand?

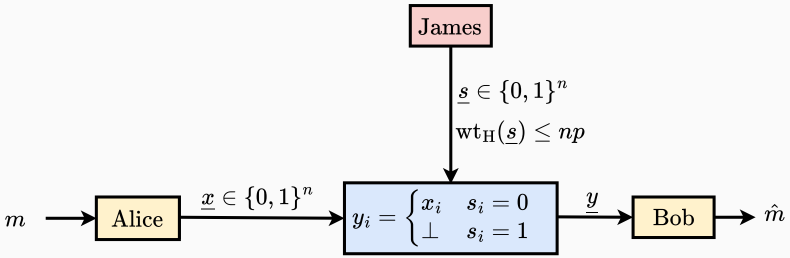
Let's zoom in on binary channels with erasures.

Binary input channels with erasures



- Alice encodes a message $m \in \{1, 2, \ldots, 2^{nR}\}$ into a codeword $\underline{x} = \{0, 1\}^n$.
- The channel *erases* bits: s_i indicates whether $y_i = x_i$ or is erased. Only np erasures can happen during the block.
- Assume $\underline{s}$ is chosen by an **adversary** James.

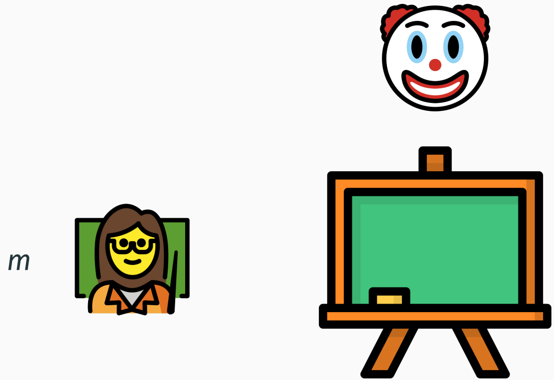
Binary input channels with erasures



- Alice encodes a message $m \in \{1, 2, \ldots, 2^{nR}\}$ into a codeword $\underline{x} = \{0, 1\}^n$.
- The channel *erases* bits: s_i indicates whether $y_i = x_i$ or is erased. Only np erasures can happen during the block.
- Assume $\underline{s}$ is chosen by an **adversary** James.

How does James choose $\underline{s}$?

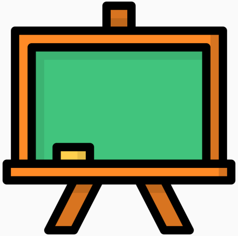
Shannon theory: James is oblivious



Shannon theory studies the binary erasure channel (BEC): $\underline{s} \overset{\text{i.i.d.}}{\sim}$ Bernoulli(p) so with high probability $\approx pn$ positions are erased.

Shannon theory: James is oblivious

001001100



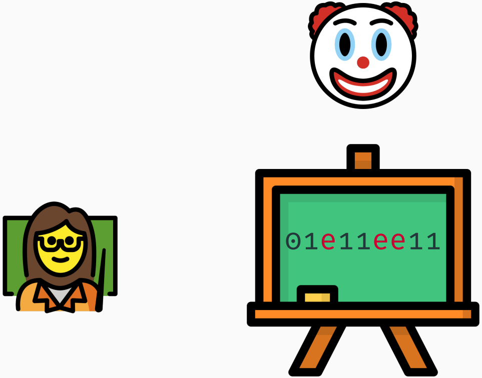
m

010111011

Shannon theory studies the binary erasure channel (BEC): $\underline{s} \overset{\text{i.i.d.}}{\sim} \mathbf{Bernoulli}(p)$ so with high probability $\approx pn$ positions are erased.

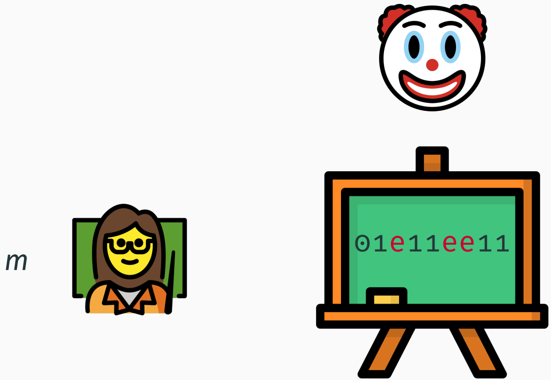
Shannon theory: James is **oblivious**

m



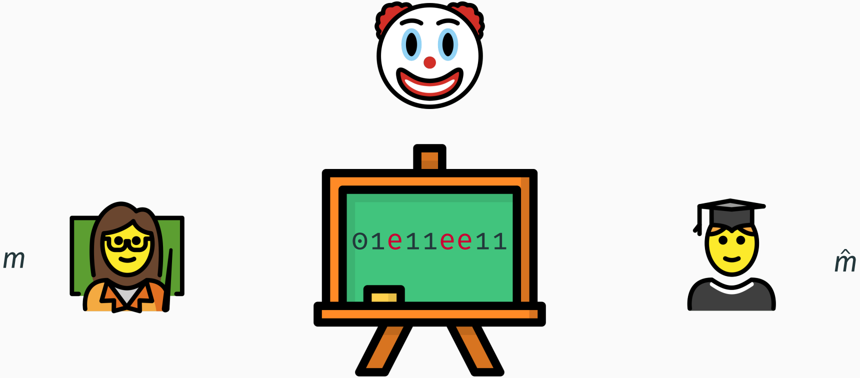
Shannon theory studies the binary erasure channel (BEC): $\underline{s} \overset{\text{i.i.d.}}{\sim}$ Bernoulli(p) so with high probability $\approx pn$ positions are erased.

Shannon theory: James is oblivious



Shannon theory studies the binary erasure channel (BEC): $\underline{s} \overset{\text{i.i.d.}}{\sim} \textbf{Bernoulli}(p)$ so with high probability $\approx pn$ positions are erased.

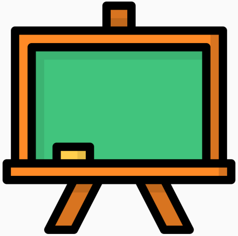
Shannon theory: James is **oblivious**



Shannon theory studies the binary erasure channel (BEC): $\underline{s} \overset{\text{i.i.d.}}{\sim}$ **Bernoulli(p)** so with high probability $\approx pn$ positions are erased.

Coding theory (“Hamming”): James is omniscient

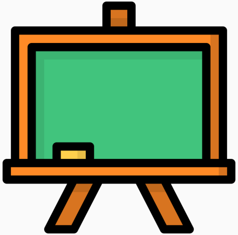
m



In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

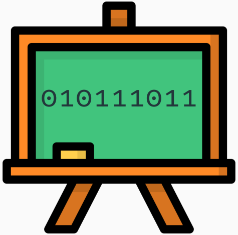
Coding theory ("Hamming"): James is omniscient

m 



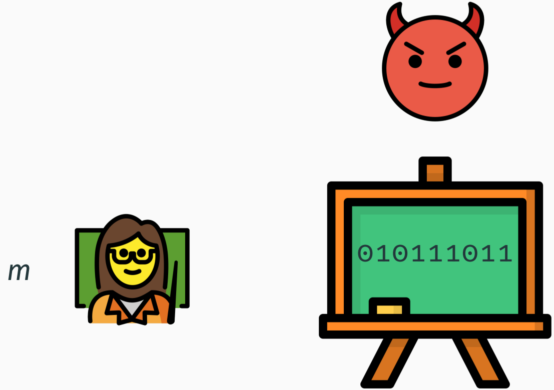
In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

Coding theory ("Hamming"): James is omniscient

m  

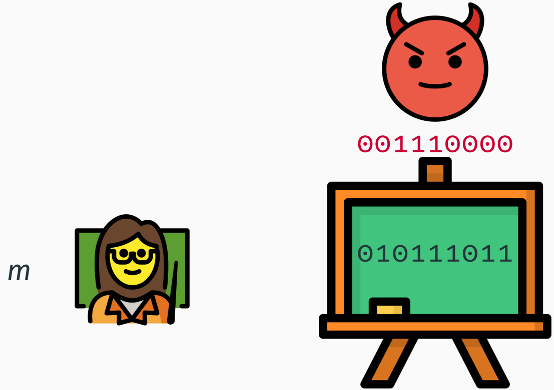
In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

Coding theory (“Hamming”): James is omniscient



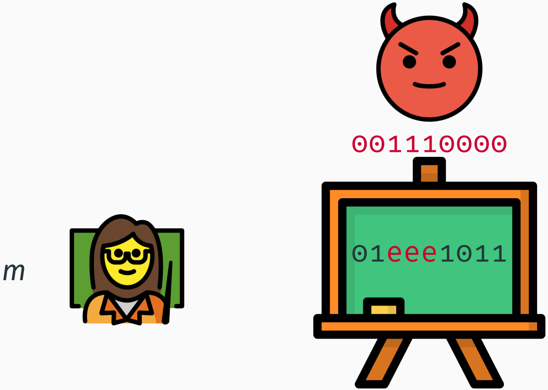
In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

Coding theory ("Hamming"): James is omniscient



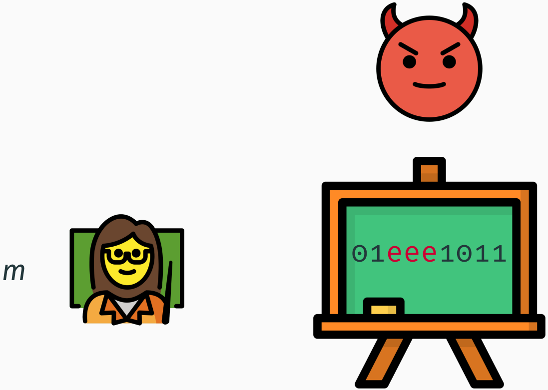
In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

Coding theory (“Hamming”): James is omniscient



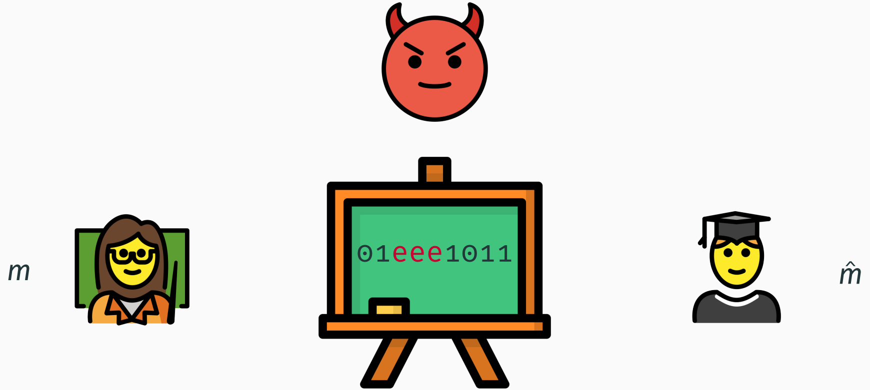
In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

Coding theory (“Hamming”): James is omniscient



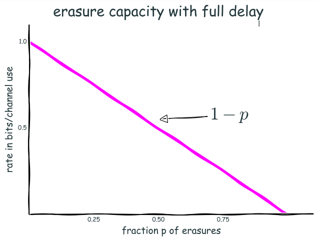
In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

Coding theory ("Hamming"): James is omniscient



In the coding theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

The erasure channel: adversarial vs. random

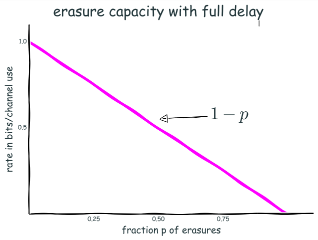


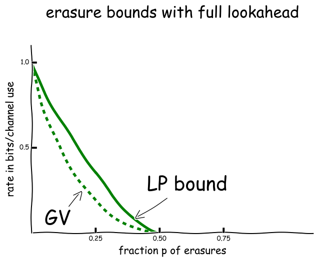
With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

There are many different ways to achieve this rate.

The erasure channel: adversarial vs. random





With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

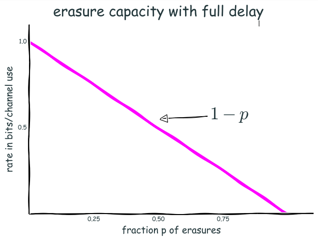
There are many different ways to achieve this rate.

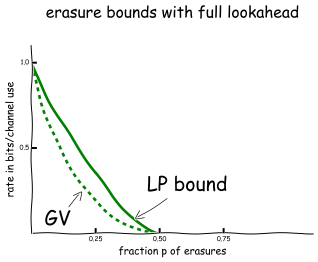
With the (Hamming-like) omniscient *worst-case* model, the capacity upper bounded:

$$C < 1 - 2p.$$

Lower bound: Gilbert-Varshamov (random) codes.

The erasure channel: adversarial vs. random





With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

There are many different ways to achieve this rate.

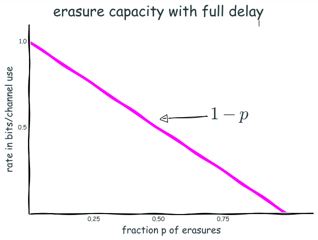
With the (Hamming-like) omniscient *worst-case* model, the capacity upper bounded:

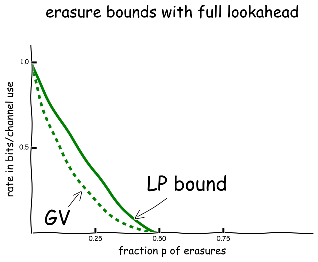
$$C < 1 - 2p.$$

Lower bound: Gilbert-Varshamov (random) codes.

That's a big gap...

The erasure channel: adversarial vs. random





With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

There are many different ways to achieve this rate.

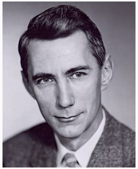
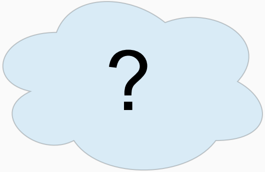
With the (Hamming-like) omniscient *worst-case* model, the capacity upper bounded:

$$C < 1 - 2p.$$

Lower bound: Gilbert-Varshamov (random) codes.

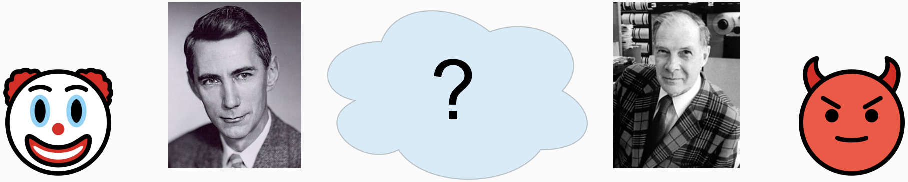
That's a big gap... where does it come from?

Why more is there to explore here?

We want to explore this gap through modeling:

Why more is there to explore here?



We want to explore this gap through modeling:

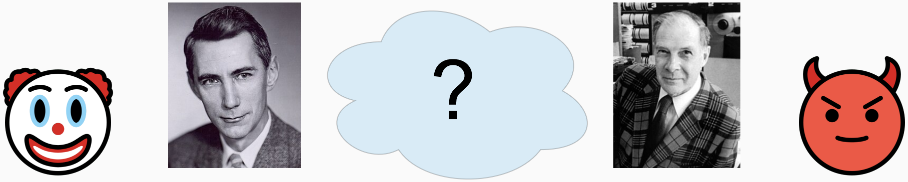
1. Use **arbitrarily varying channels (AVCs)** to develop a **unified framework** for both the Shannon and Hamming models.

Why more is there to explore here?

We want to explore this gap through modeling:

1. Use **arbitrarily varying channels (AVCs)** to develop a **unified framework** for both the Shannon and Hamming models.
2. Explore **intermediate models** to see **what causes the gap**.

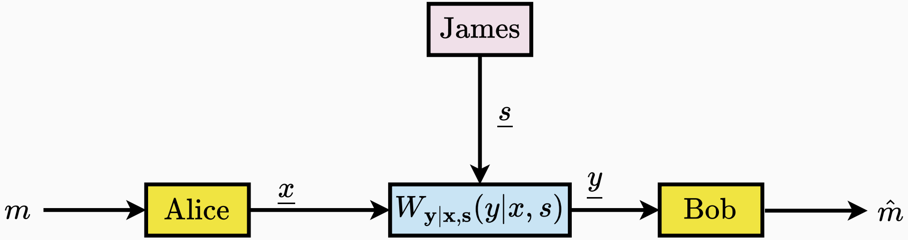
Why more is there to explore here?



We want to explore this gap through modeling:

1. Use **arbitrarily varying channels (AVCs)** to develop a **unified framework** for both the Shannon and Hamming models.
2. Explore **intermediate models** to see **what causes the gap**.
3. Discover **coding strategies** to see what **resources are needed** to communicate reliably.

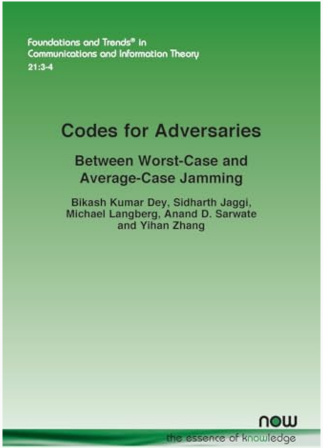
AVCs model channel "noise" as a state variable



In an **adversarial channel model**, **Alice** wants to communicate with **Bob** over a channel whose time-varying state is controlled by an adversarial **jammer** James.

- Alice and James may be **constrained** in how they communicate.
- Capacity depends on **what James knows** about m and $\underline{x}$.

Shameless self-promotion



This talk is based on a recent (December 2024) monograph: check it out!

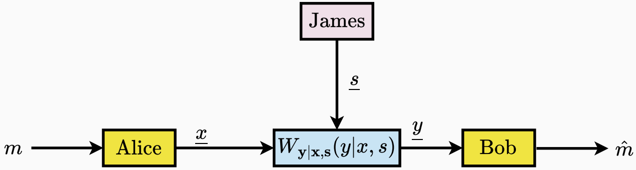
- ✔ Unified treatment of random noise (Shannon-theoretic) and worst-case noise (coding-theoretic).
- ✔ Intermediate models for jammers who can eavesdrop: online and myopic.
- ✔ Examples, open problems, and more!

What’s coming up next

1. Arbitrarily varying channels (AVCs)
2. Some key ingredients
3. Causal adversarial models
4. Myopic adversarial models
5. Computationally efficient codes for causal adversaries
6. Looking forward

Arbitrarily varying channels (AVCs)

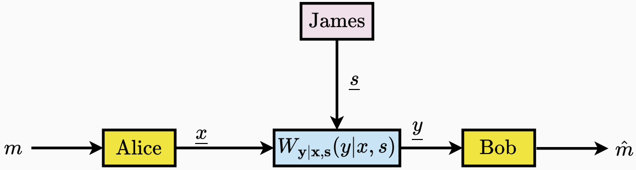
The basic channel model



Let $\mathcal{X}$, $\mathcal{S}$, and $\mathcal{Y}$ be discrete alphabets. An AVC is a discrete channel $W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$ such that

$$W_{\underline{\mathbf{y}}|\underline{\mathbf{x}},\underline{\mathbf{s}}}(\underline{y}|\underline{x},\underline{s}) = \prod_{i=1}^{n} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y_i|x_i,s_i)$$

The basic channel model

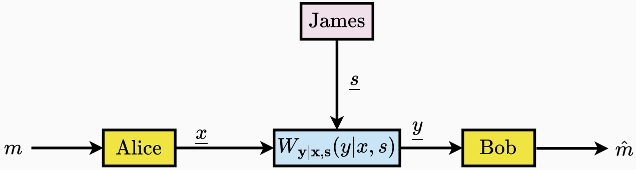


Let $\mathcal{X}$, $\mathcal{S}$, and $\mathcal{Y}$ be discrete alphabets. An AVC is a discrete channel $W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$ such that

$$W_{\underline{\mathbf{y}}|\underline{\mathbf{x}},\underline{\mathbf{s}}}(\underline{y}|\underline{x},\underline{s}) = \prod_{i=1}^{n} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y_i|x_i,s_i)$$

The **state** $\underline{s} \in \mathcal{S}^n$ is controlled by an adversarial **jammer** (James).

The basic channel model



Let $\mathcal{X}$, $\mathcal{S}$, and $\mathcal{Y}$ be discrete alphabets. An AVC is a discrete channel $W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$ such that

$$W_{\underline{\mathbf{y}}|\underline{\mathbf{x}},\underline{\mathbf{s}}}(\underline{y}|\underline{x},\underline{s}) = \prod_{i=1}^{n} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y_i|x_i,s_i)$$

The **state** $\underline{s} \in \mathcal{S}^n$ is controlled by an adversarial **jammer** (James).
Examples: For binary channels $\underline{s}$ could be an error or erasure pattern.

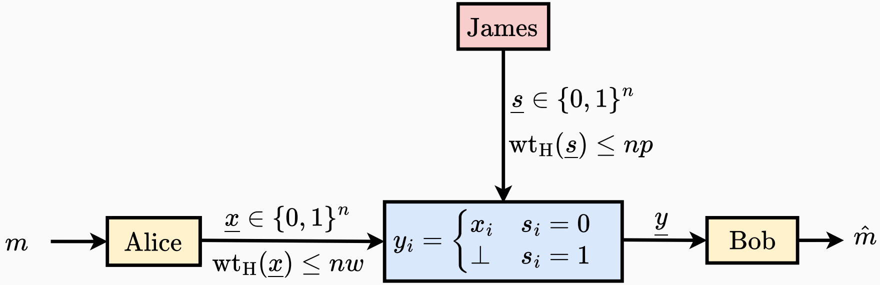
Input and cost constraints for AVCs

We impose that the types $T_{\underline{x}}$ and $T_{\underline{s}}$ of the codeword $\underline{x}$ and the state $\underline{s}$ lie be in convex subsets of the probability simplices $\Delta(\mathcal{X})$ and $\Delta(\mathcal{S})$:

$$T_{\underline{x}} \in \Gamma \subseteq \Delta(\mathcal{X})$$

$$T_{\underline{s}} \in \Lambda \subseteq \Delta(\mathcal{S})$$

Example: For binary channels $\underline{x}$ and $\underline{s}$ have bounded Hamming weight.

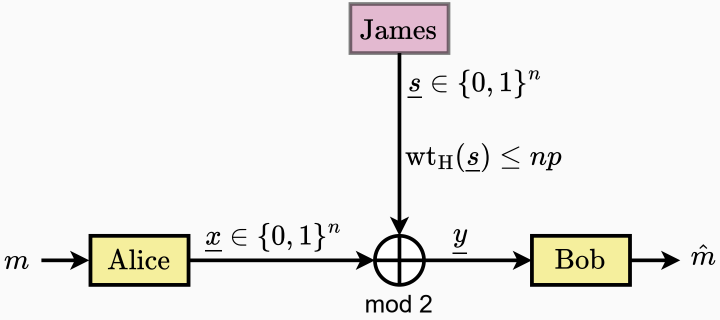


Input and cost constraints for AVCs

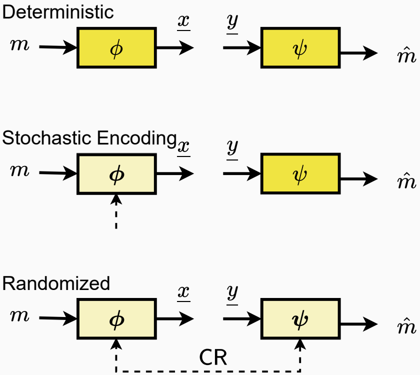
We impose that the types $T_{\underline{x}}$ and $T_{\underline{s}}$ of the codeword $\underline{x}$ and the state $\underline{s}$ lie be in convex subsets of the probability simplices $\Delta(\mathcal{X})$ and $\Delta(\mathcal{S})$:

$$T_{\underline{x}} \in \Gamma \subseteq \Delta(\mathcal{X})$$
$$T_{\underline{s}} \in \Lambda \subseteq \Delta(\mathcal{S})$$

Example: For binary channels $\underline{x}$ and $\underline{s}$ have bounded Hamming weight.



Defining codes and input constraints



An (n, M, Γ) code is

$$\begin{aligned} \phi &: [M] \to \mathcal{X}^n && \text{(encoder)} \\ \psi &: \mathcal{Y}^n \to [M] && \text{(decoder)} \end{aligned}$$

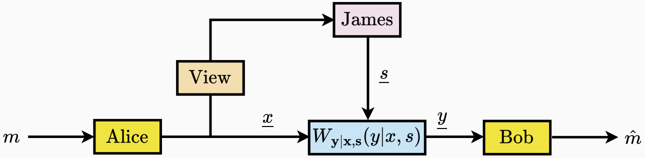
such that

$$T_{\phi(m)} \in \Gamma$$

The rate is $R = \frac{1}{n} \log_2(M)$.

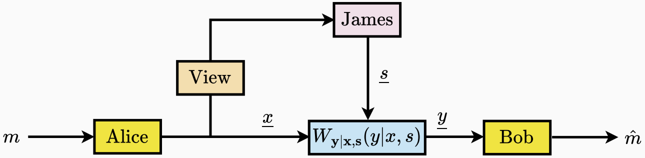
A **randomized code** lets Alice and Bob choose their code in secret. If Alice and Bob do not share common randomness, Alice can still use **stochastic encoding**.

What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

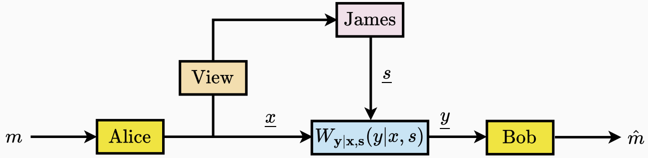
What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.

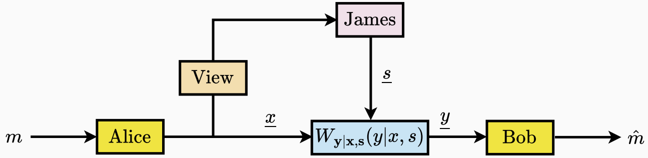
What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).

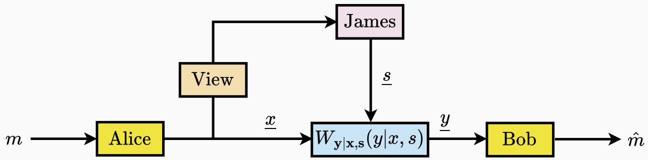
What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

What James knows: Shannon, Hamming, and in between

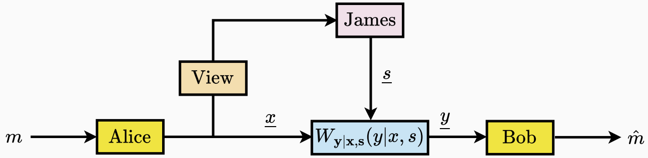


James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

These constrain the set of **strategies** James can use.

What James knows: Shannon, Hamming, and in between



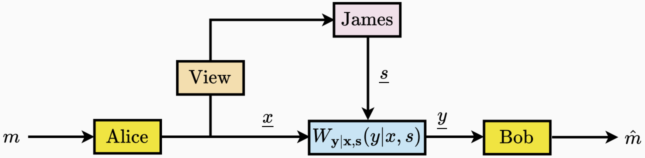
James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

These constrain the set of **strategies** James can use.

- **Oblivious** (Shannon): the message only.

What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

These constrain the set of **strategies** James can use.

- **Oblivious** (Shannon): the message only.
- **Omniscient** (Hamming): the message and the codeword.

Maximal error and capacity

The **error** for a particular message m is

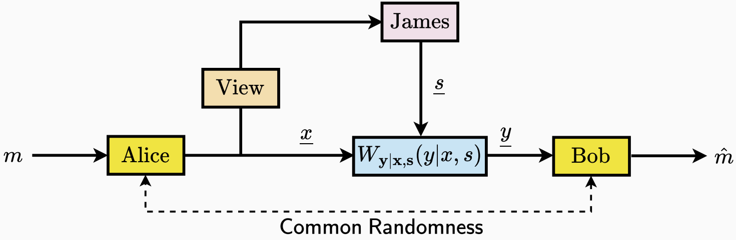
$$P_{\mathrm{err}}(m, \phi, \psi) = \max_{\text{jamming strategies}} \sum_{\mathbf{x} \in \mathcal{X}^n} \mathbb{P}\left(\psi(\mathbf{y}) \neq m \mid \mathbf{x}\right) \mathbb{P}_{\phi}\left(\phi(m) = \mathbf{x}\right)$$

A rate R is **achievable** if for any $\epsilon > 0$ there exists an infinite sequence of rate R codes ($n \to \infty$) such that $P_{\mathrm{err}}(m, \phi, \psi) < \epsilon$ for all m.

The capacities C_{obl} and C_{omni} for oblivious and omniscient cases satisfy:

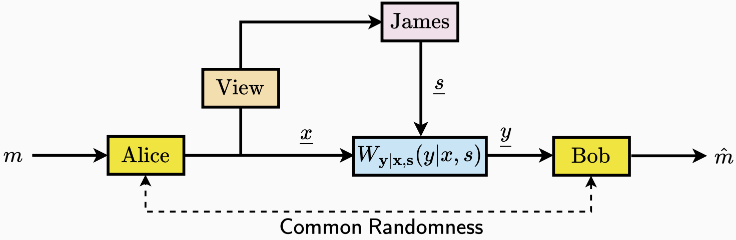
$$(\textit{Hamming}) \qquad C_{\mathrm{omni}} \qquad \leq \qquad C_{\mathrm{obl}} \qquad (\textit{Shannon})$$

Common randomness makes the problem easier



Blackwell et al. (1960) proposed the AVC model and studied **randomized codes**, where Alice and Bob share common randomness. James just minimizes the mutual information over equivalent DMCs:

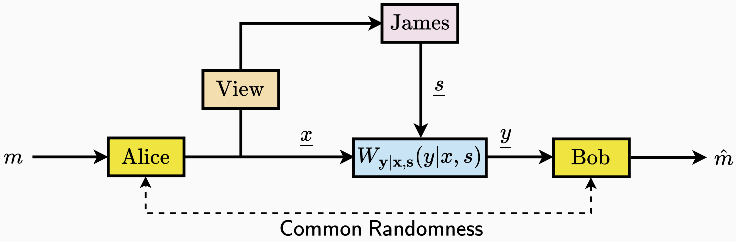
Common randomness makes the problem easier



Blackwell et al. (1960) proposed the AVC model and studied **randomized codes**, where Alice and Bob share common randomness. James just minimizes the mutual information over equivalent DMCs:

- **Oblivious:** find $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)Q_{\mathbf{s}}(s)$ with lowest Shannon capacity.

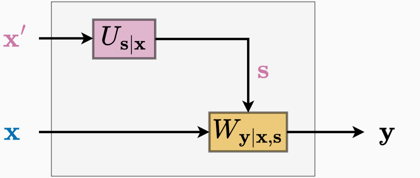
Common randomness makes the problem easier



Blackwell et al. (1960) proposed the AVC model and studied **randomized codes**, where Alice and Bob share common randomness. James just minimizes the mutual information over equivalent DMCs:

- **Oblivious:** find $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)Q_{\mathbf{s}}(s)$ with lowest Shannon capacity.
- **Omniscient:** find $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)U_{\mathbf{s}|\mathbf{x}}(s|x)$ with lowest Shannon capacity.

Without Common Randomness: Symmetrization Attacks

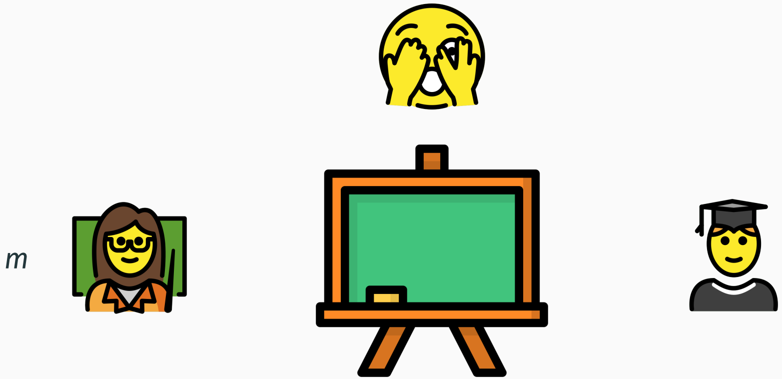


An AVC is **Ericson-Csiszár-Narayan (ECN) symmetrizable** if James can spoof Alice's codeword. That is, for all $(\mathbf{y}, \mathbf{x}, \mathbf{x}')$, we have

$$\sum_{s} U_{\mathbf{s}|\mathbf{x}'} W_{\mathbf{y}|\mathbf{x},\mathbf{s}} = \sum_{s} U_{\mathbf{s}|\mathbf{x}} W_{\mathbf{y}|\mathbf{x}',\mathbf{s}}.$$

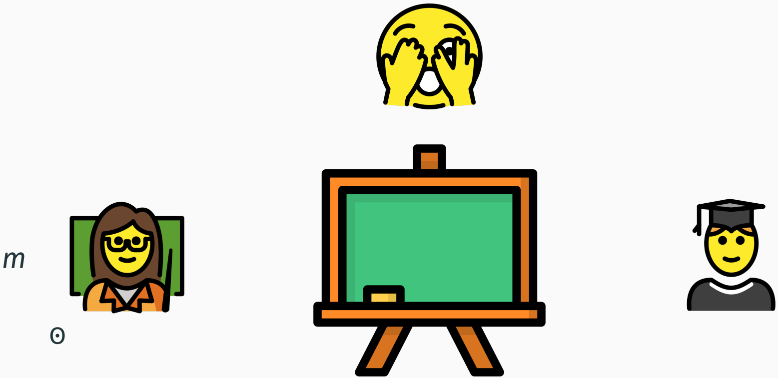
Without common randomness, the capacity of a symmetrizable AVC $C_{\mathrm{obl}} = 0$.

Intermediate model 1: delayed information for James



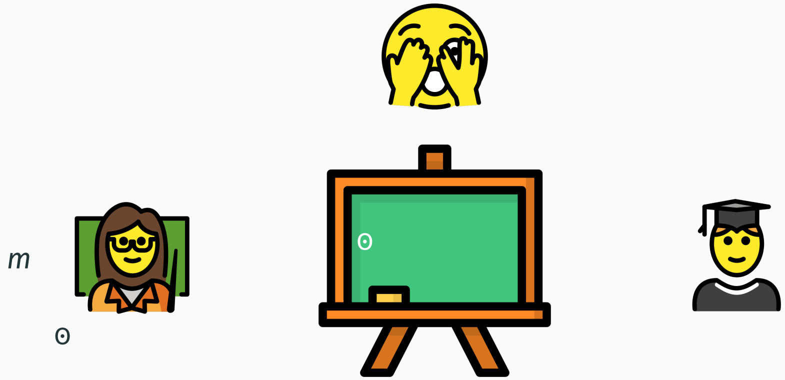
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



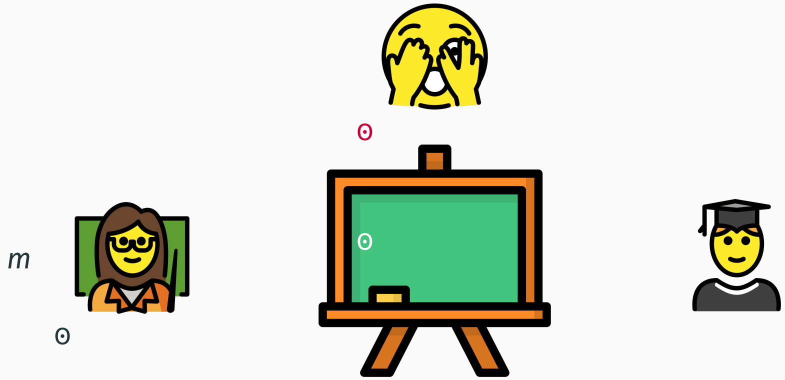
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



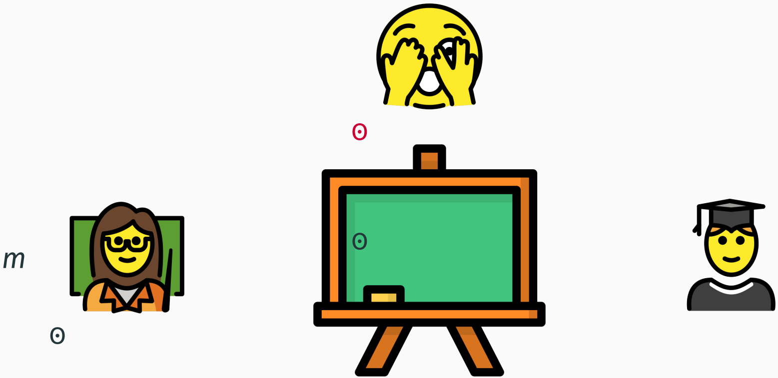
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



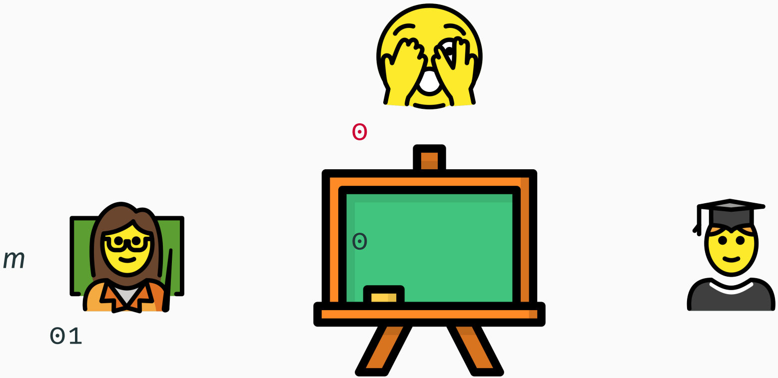
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



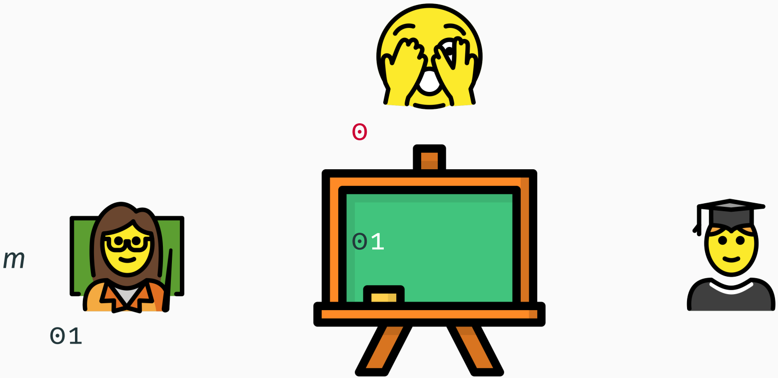
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



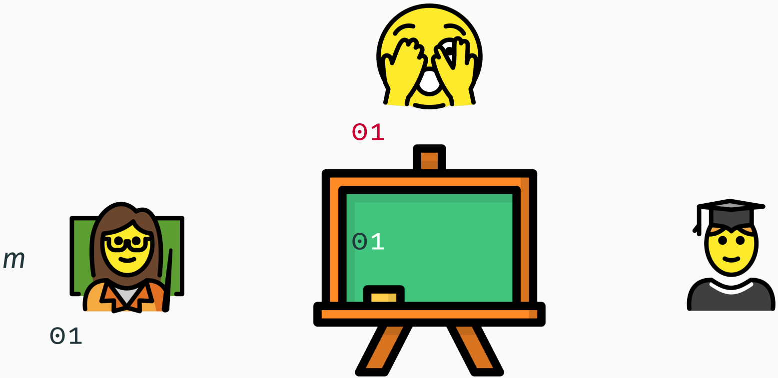
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



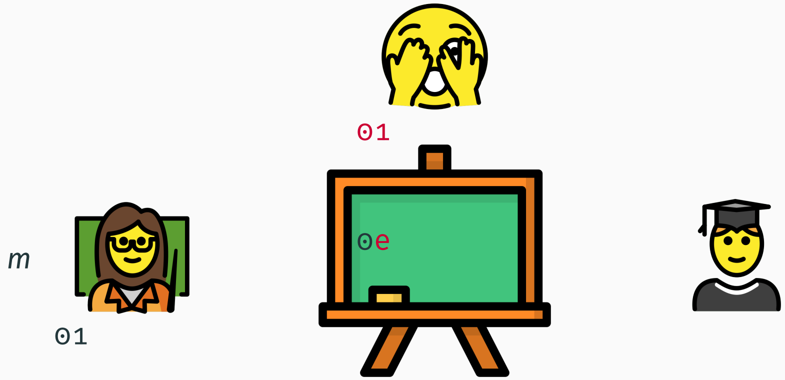
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



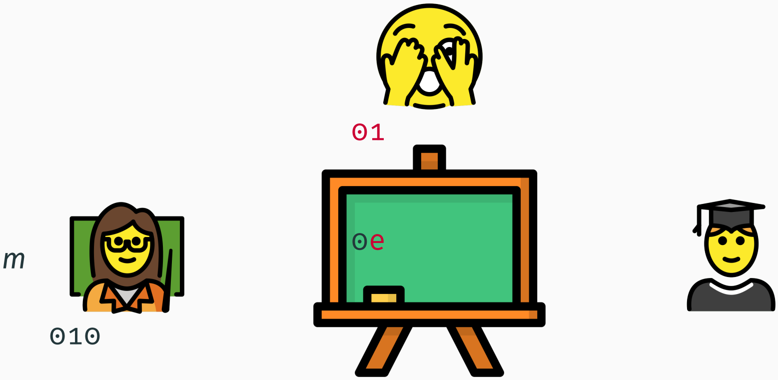
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



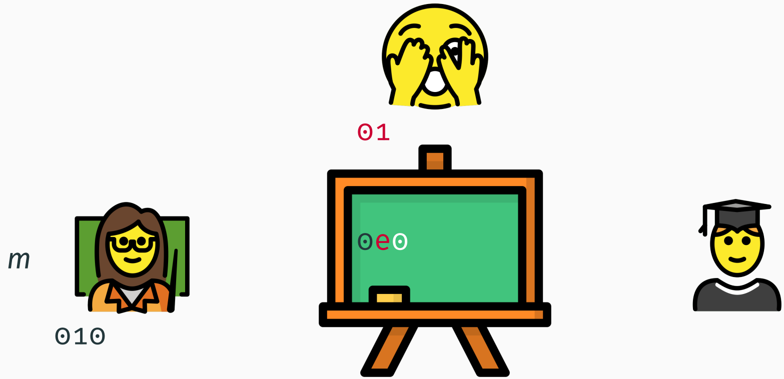
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



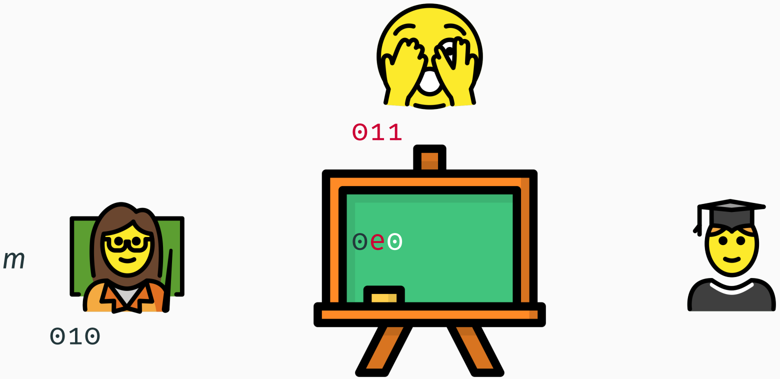
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



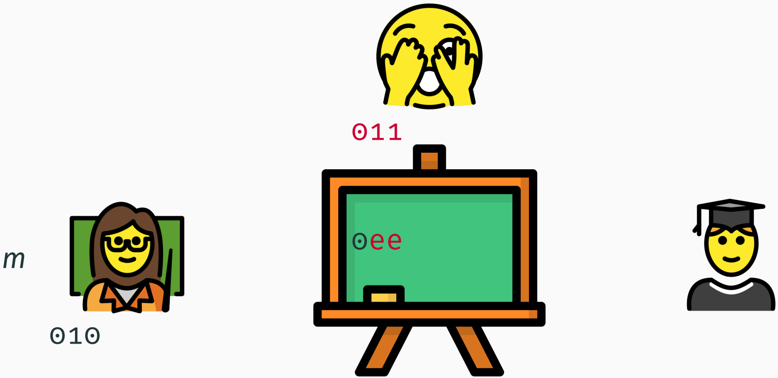
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



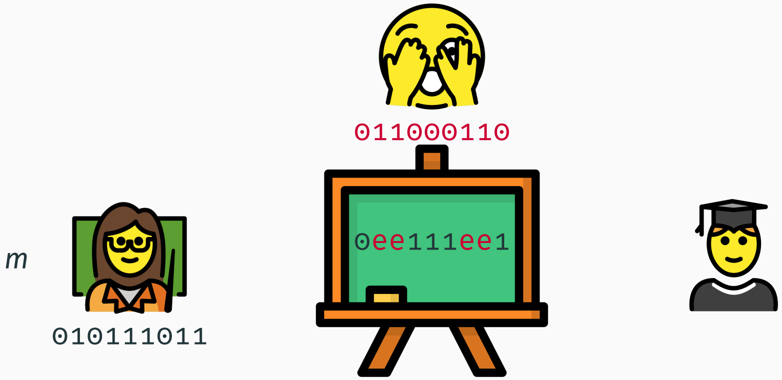
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



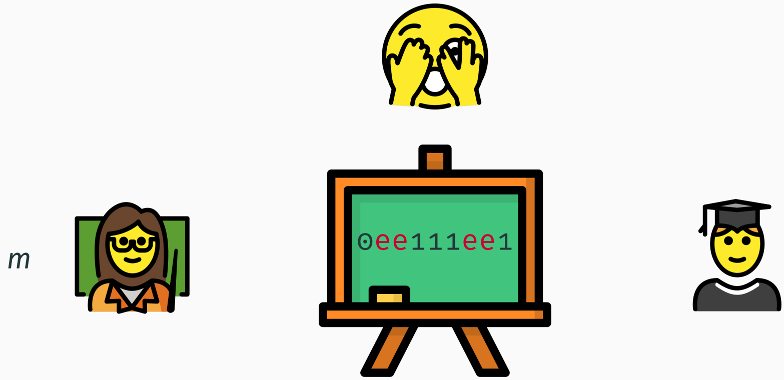
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



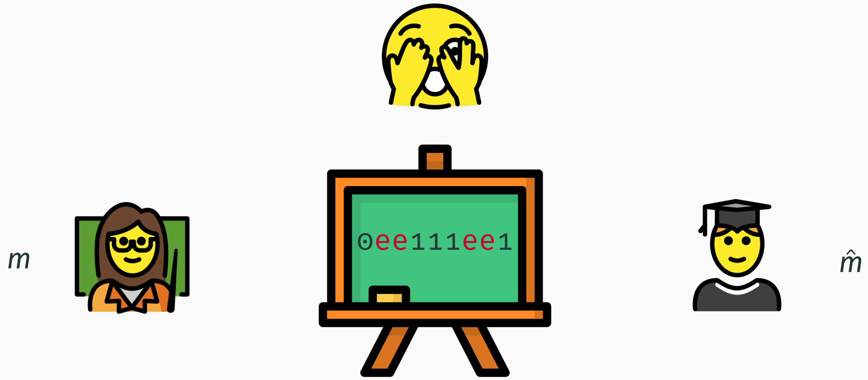
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James



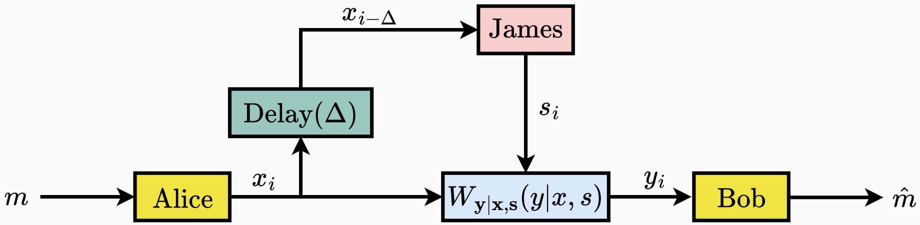
Delayed: James gets a delayed view of the transmitted codeword.

Intermediate model 1: delayed information for James

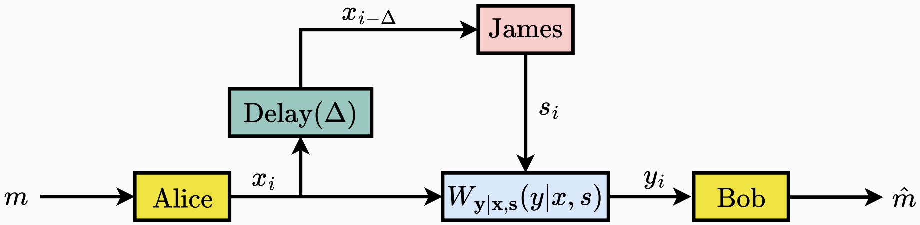


Delayed: James gets a delayed view of the transmitted codeword.

Delay interpolates between oblivious and omniscient

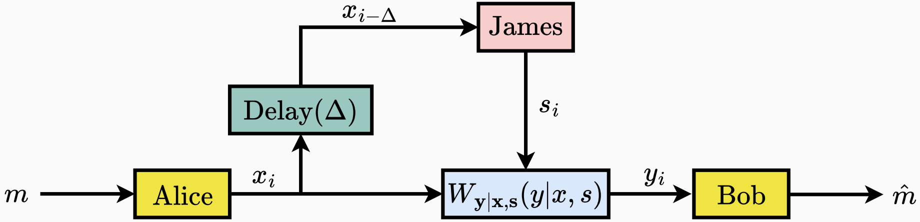


Delay interpolates between oblivious and omniscient



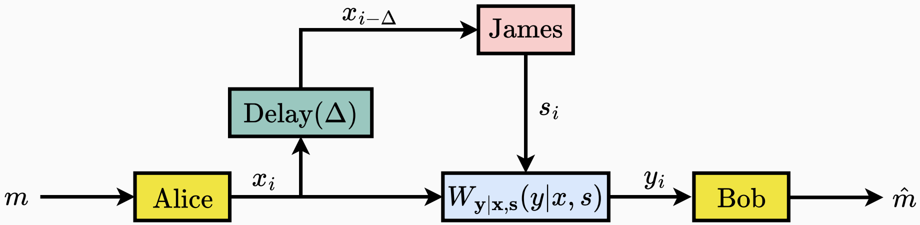
- $\Delta = n$ **(oblivious):** capacity $= 1 - p$ ("Shannon")

Delay interpolates between oblivious and omniscient



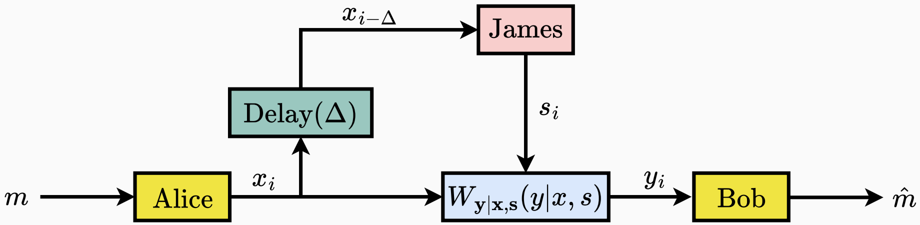
- $\Delta = n$ **(oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$

Delay interpolates between oblivious and omniscient



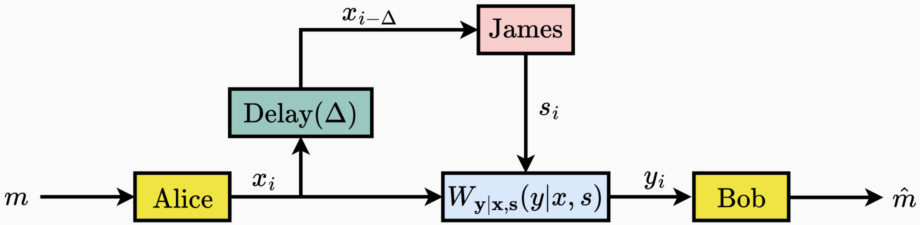
- $\Delta = n$ **(oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$
- $\Delta = 0$ **("causal"):** capacity $= 1 - 2p$

Delay interpolates between oblivious and omniscient



- $\Delta = n$ **(oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$
- $\Delta = 0$ **("causal"):** capacity $= 1 - 2p$
- $\Delta = -n$ **(omniscient):** capacity $\leq 1 - 2p$ ("Hamming")

Delay interpolates between oblivious and omniscient

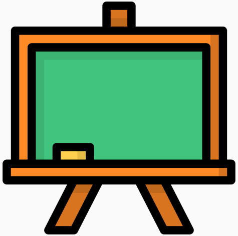


- $\Delta = n$ **(oblivious):** capacity $= 1 - p$ (“Shannon”)
- $\Delta = 1$ **(“one bit delay”):** capacity $= 1 - p$
- $\Delta = 0$ **(“causal”):** capacity $= 1 - 2p$
- $\Delta = -n$ **(omniscient):** capacity $\leq 1 - 2p$ (“Hamming”)

Knowing just the current input gives James a lot of power!

Intermediate model 2: Myopic adversarial models

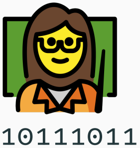
m 

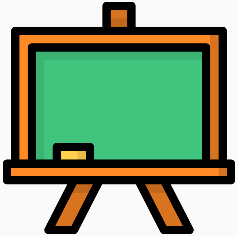


Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

m

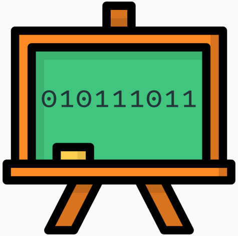




Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

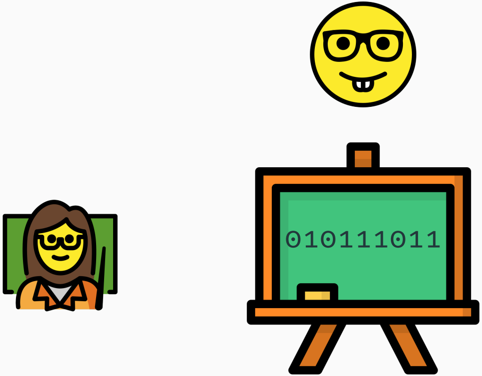
m



Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

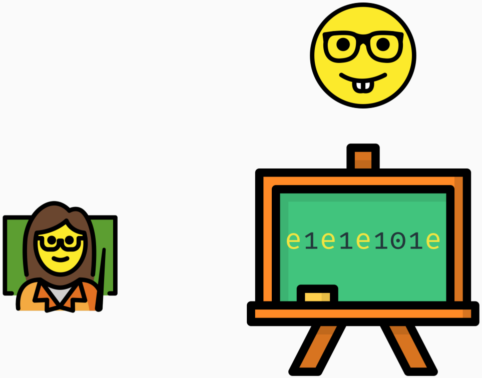
m



Myopic: James gets a noisy view of the transmitted codeword.

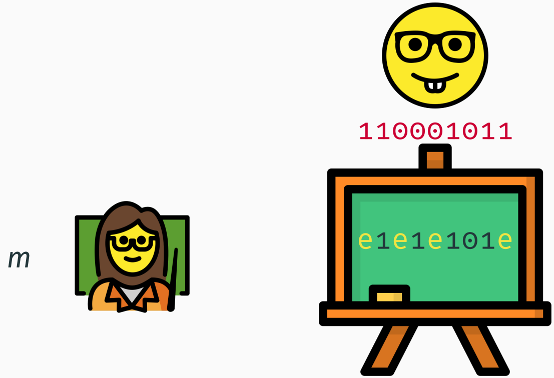
Intermediate model 2: Myopic adversarial models

m



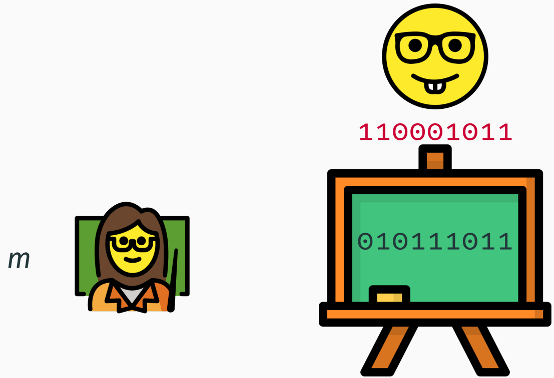
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models



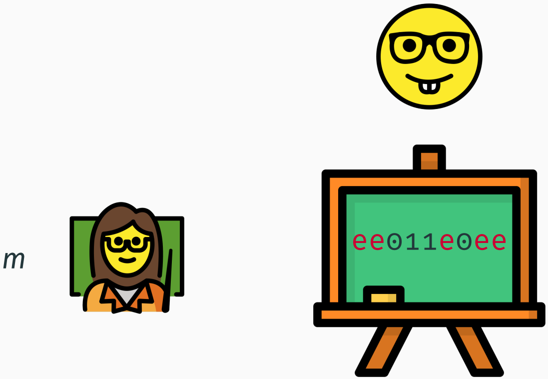
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models



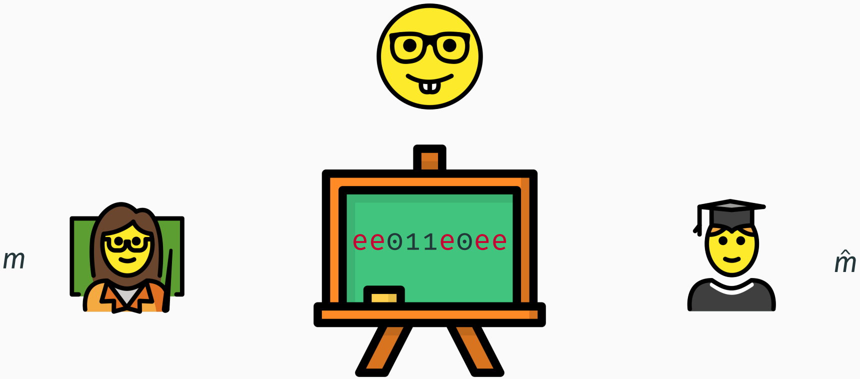
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models



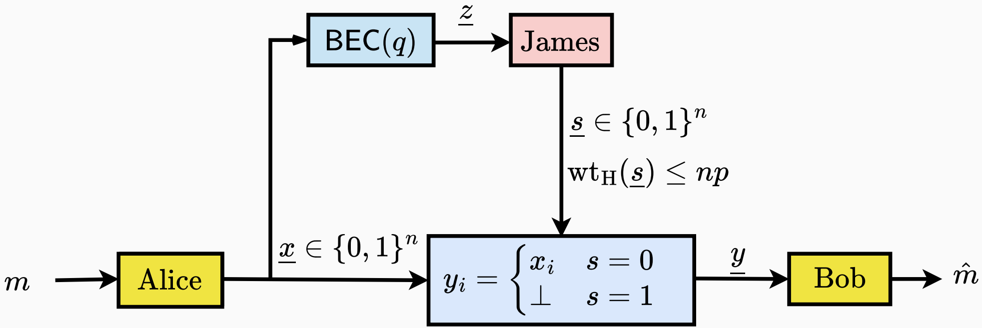
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

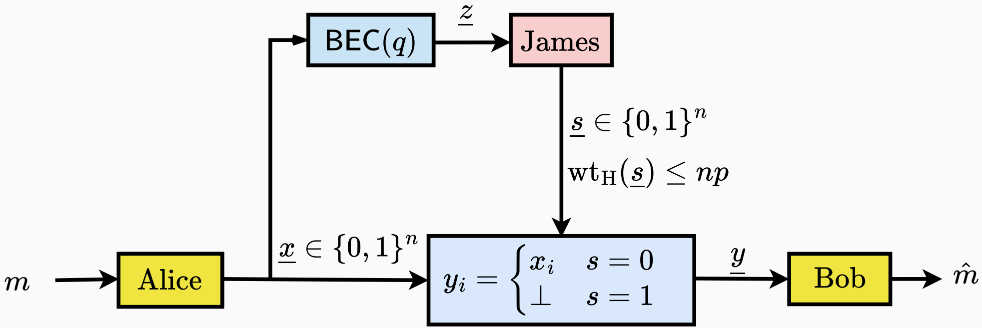


Myopic: James gets a noisy view of the transmitted codeword.

The impact of myopia in the erasure setting

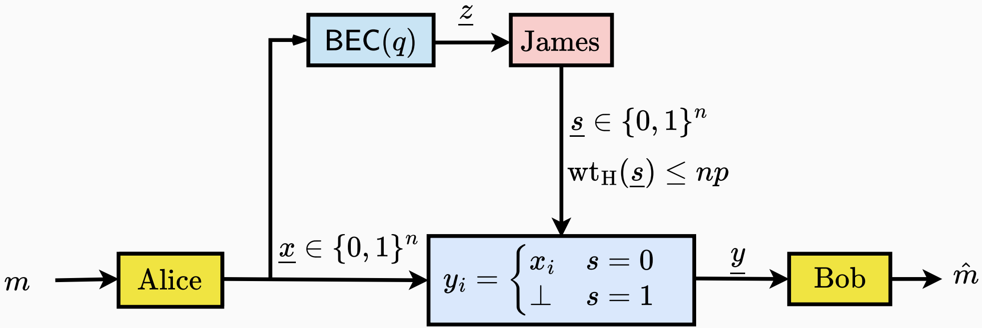


The impact of myopia in the erasure setting



- **Sufficiently myopic:** ($p < q$): capacity $= 1 - p$

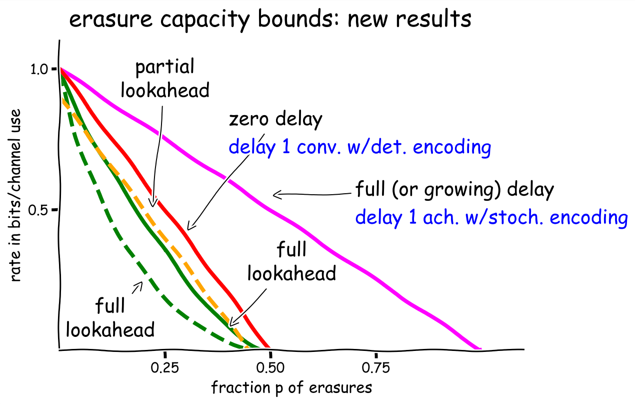
The impact of myopia in the erasure setting



- **Sufficiently myopic:** ($p < q$): capacity $= 1 - p$
- **Otherwise:** ($p > q$): it's more complicated...

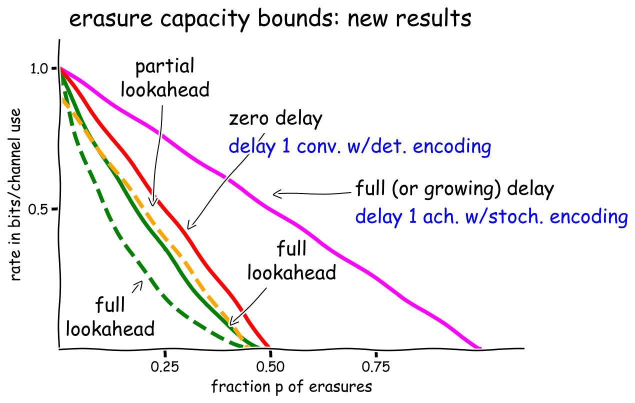
Some key ingredients

Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

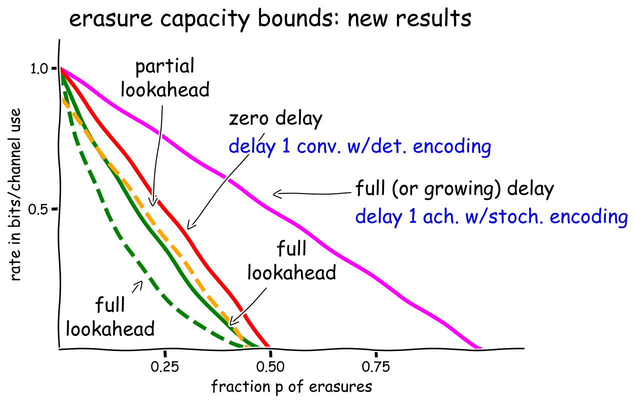
Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).

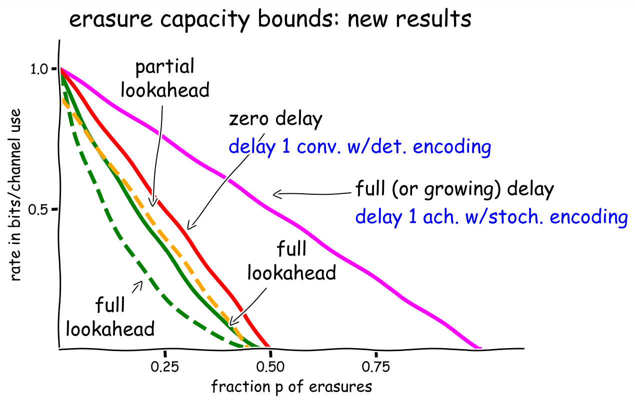
Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).
- Low weight “fuzz” as a side channel.

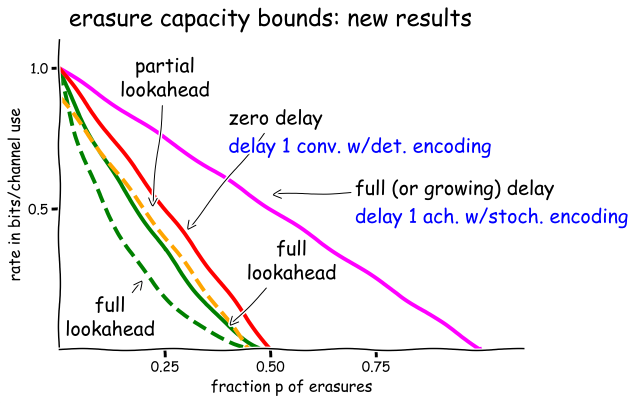
Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).
- Low weight “fuzz” as a side channel.
- Select a codebook from a smaller “library”.

Ingredient 1: stochastic encoding

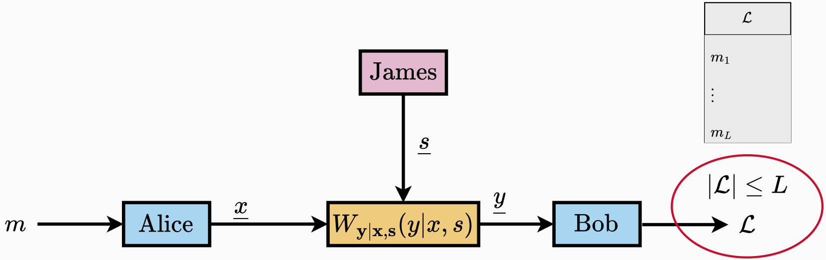


In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).
- Low weight "fuzz" as a side channel.
- Select a codebook from a smaller "library".

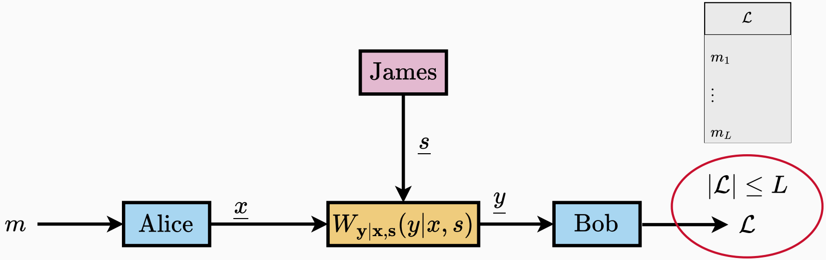
It can be **necessary**: deterministic erasure codes cannot do better than $1 - 2p$ against a James who has a single bit of delay.

Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

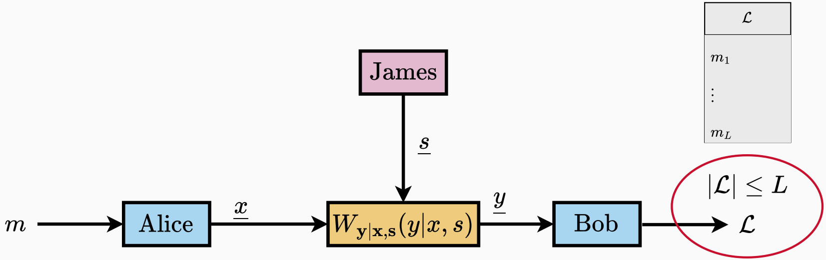
Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.

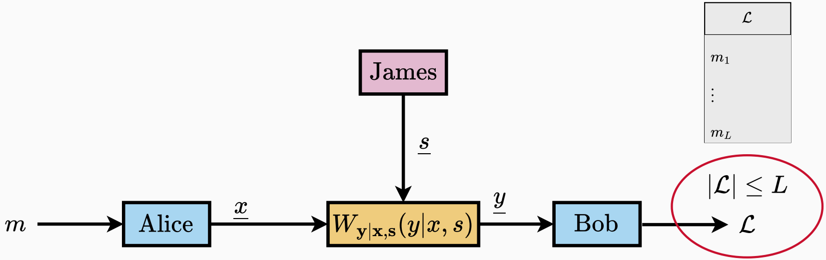
Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.
- Require the list size is no larger than than L.

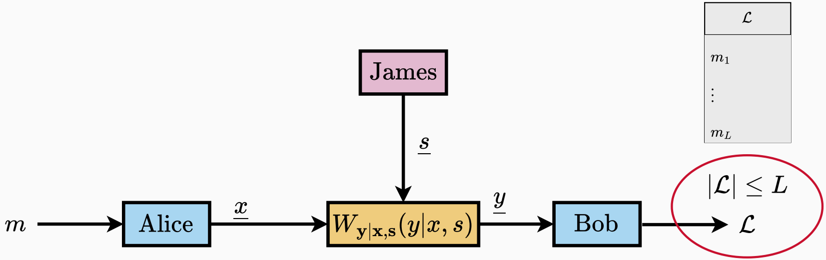
Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.
- Require the list size is no larger than than L.
- Different L are useful in different cases: constant, $\mathrm{poly}(n)$, or $\exp(\epsilon n)$

Ingredient 2: list decoding



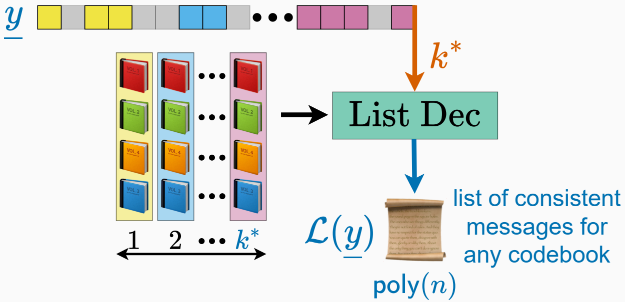
In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.
- Require the list size is no larger than than L.
- Different L are useful in different cases: constant, $\mathrm{poly}(n)$, or $\exp(\epsilon n)$

In some cases the list decoding capacity allows **strictly larger** rates:

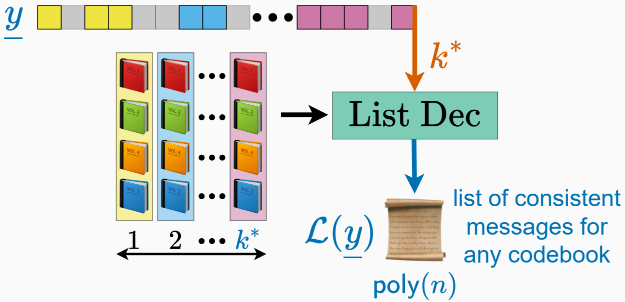
$$C_{\mathrm{list}}(L) > C_{\mathrm{obl}}.$$

List decoding appears in many ways



List decoding to a “small list” is useful in decoding and jamming:

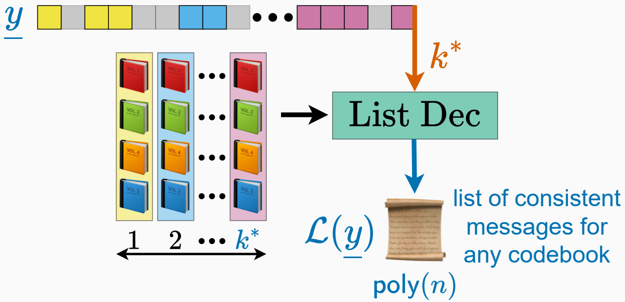
List decoding appears in many ways



List decoding to a "small list" is useful in decoding and jamming:

- List codes mean $O(\log n)$ bits of common randomness is sufficient.

List decoding appears in many ways



List decoding to a "small list" is useful in decoding and jamming:

- List codes mean $O(\log n)$ bits of common randomness is sufficient.
- James can list decode to jam more effectively.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

- Start with a marginal distribution $P_{\mathbf{x}} \in \Delta(\mathcal{X})$.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

- Start with a marginal distribution $P_{\mathbf{x}} \in \Delta(\mathcal{X})$.
- A **self-coupling** is a joint distribution $P_{\mathbf{x},\mathbf{x}'}$ where each marginal is $P_{\mathbf{x}}$.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

- Start with a marginal distribution $P_{\mathbf{x}} \in \Delta(\mathcal{X})$.
- A **self-coupling** is a joint distribution $P_{\mathbf{x},\mathbf{x}'}$ where each marginal is $P_{\mathbf{x}}$.
- A self-coupling is **completely positive** if it is a mixture of independent self-couplings:

$$P_{\mathbf{x},\mathbf{x}'}(x, x') = \sum_{i=1}^{|\mathcal{U}|} P_{\mathbf{u}}(i) P_{\mathbf{x}_i}(x) P_{\mathbf{x}_i}(x').$$

Generalizing the Plotkin bound

Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}$$

Generalizing the Plotkin bound

Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}$$

- It turns out that any codes with this property cannot be too large (for large n)!

Generalizing the Plotkin bound

Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}$$

- It turns out that any codes with this property cannot be too large (for large n)!
- Compare this to the Plotkin bound: an upper bound on the size of binary codes with a given distance.

Generalizing the Plotkin bound

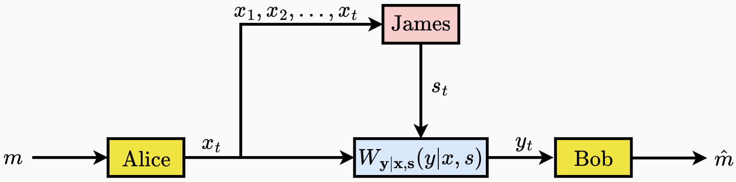
Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}$$

- It turns out that any codes with this property cannot be too large (for large n)!
- Compare this to the Plotkin bound: an upper bound on the size of binary codes with a given distance.
- If our rate is too high, then there will a constant fraction of codeword pairs whose type is close to CP.

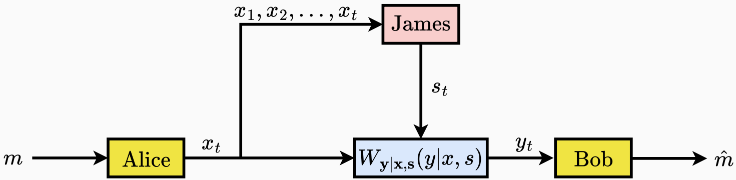
Causal adversarial models

Causal adversaries: James can see the current input



What is "symmetrizability" here and how should James act?

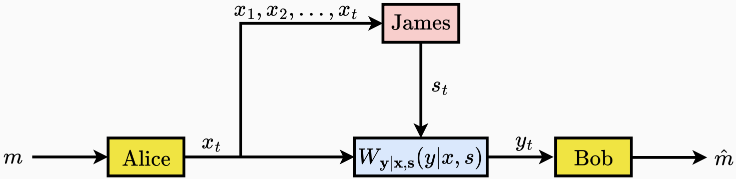
Causal adversaries: James can see the current input



What is “symmetrizability” here and how should James act?

- Spend less power at the beginning to save it up and then push hard in the second half? Bob will get a better initial estimate.

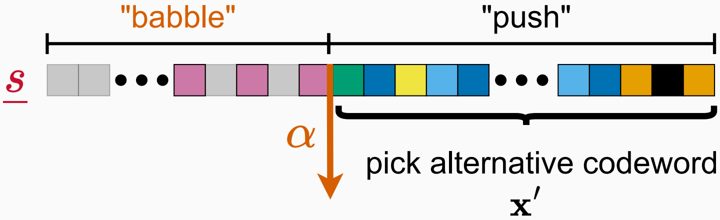
Causal adversaries: James can see the current input



What is "symmetrizability" here and how should James act?

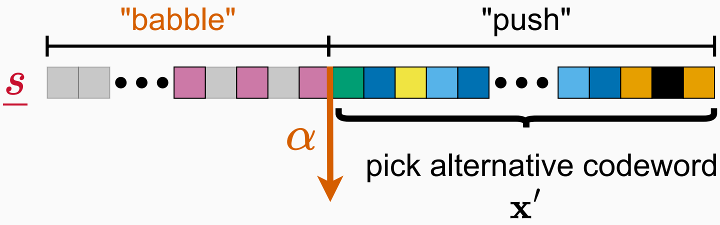
- Spend less power at the beginning to save it up and then push hard in the second half? Bob will get a better initial estimate.
- Spend more power at the beginning in the hope of leading Bob astray? But then the suffix might resolve Bob's uncertainty.

Babble and push: an attack for James



Alice and Bob pick a coding strategy and reveal it to James, who...

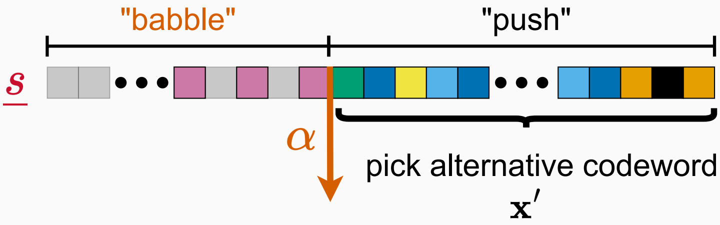
Babble and push: an attack for James



Alice and Bob pick a coding strategy and reveal it to James, who...

1. Splits time into K blocks of length $\epsilon_c n$.

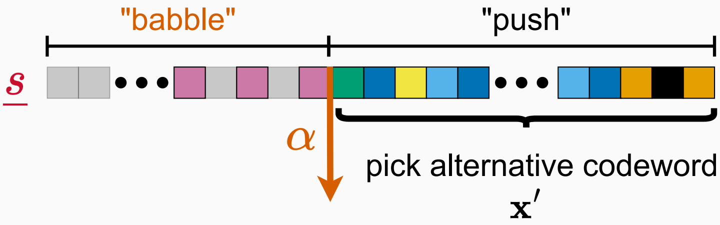
Babble and push: an attack for James



Alice and Bob pick a coding strategy and reveal it to James, who...

1. Splits time into K blocks of length $\epsilon_c n$.
2. "Distills" a large (constant fraction) subcode where $\underline{x} \approx$ the same type.

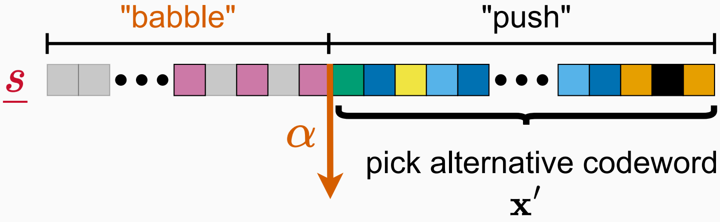
Babble and push: an attack for James



Alice and Bob pick a coding strategy and reveal it to James, who...

1. Splits time into K blocks of length $\epsilon_c n$.
2. "Distills" a large (constant fraction) subcode where $\underline{x} \approx$ the same type.
3. "Babbles" by using a random attack $V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}$ for $u \leq \alpha K$.

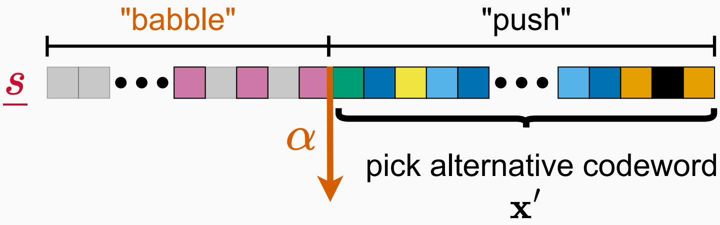
Babble and push: an attack for James



Alice and Bob pick a coding strategy and reveal it to James, who...

1. Splits time into K blocks of length $\epsilon_c n$.
2. "Distills" a large (constant fraction) subcode where $\underline{x} \approx$ the same type.
3. "Babbles" by using a random attack $V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}$ for $u \leq \alpha K$.
4. "Pushes" using codeword-dependent $V_{\mathbf{s}|\mathbf{x},\mathbf{x}'\mathbf{u}=u}$ for $u > \alpha K$.

Babble and push: an attack for James

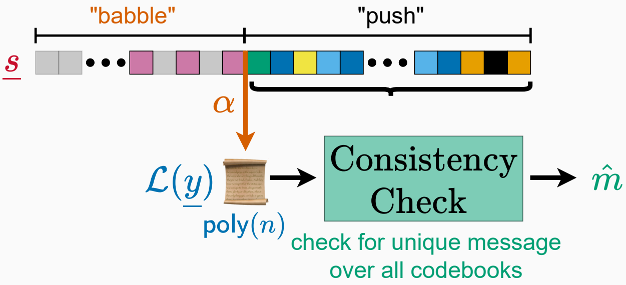


Alice and Bob pick a coding strategy and reveal it to James, who...

1. Splits time into K blocks of length $\epsilon_c n$.
2. "Distills" a large (constant fraction) subcode where $\underline{x} \approx$ the same type.
3. "Babbles" by using a random attack $V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}$ for $u \leq \alpha K$.
4. "Pushes" using codeword-dependent $V_{\mathbf{s}|\mathbf{x},\mathbf{x}'\mathbf{u}=u}$ for $u > \alpha K$.

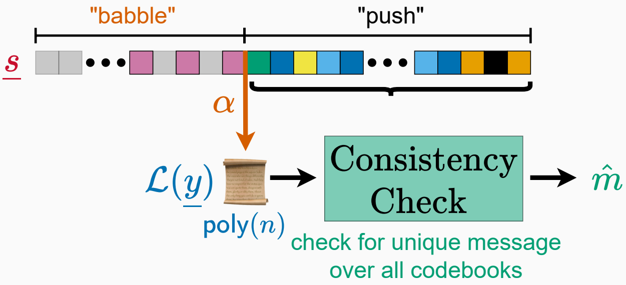
Use the generalized Plotkin bound (plus more) to show this will work.

Achievability



Achievable scheme also uses a block structure:

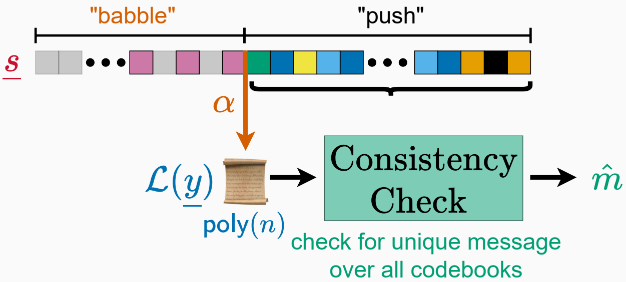
Achievability



Achievable scheme also uses a block structure:

- Alice encodes m using independent randomness in each chunk.

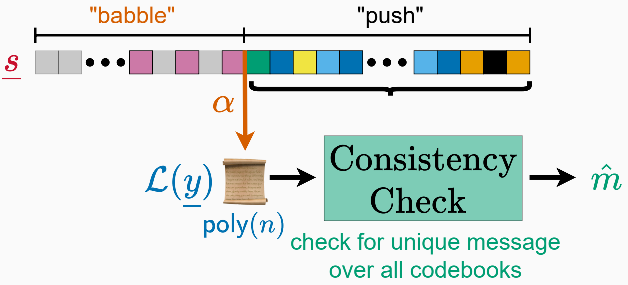
Achievability



Achievable scheme also uses a block structure:

- Alice encodes m using independent randomness in each chunk.
- Bob list decodes after each chunk assuming a random attack $\{V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}\}$.

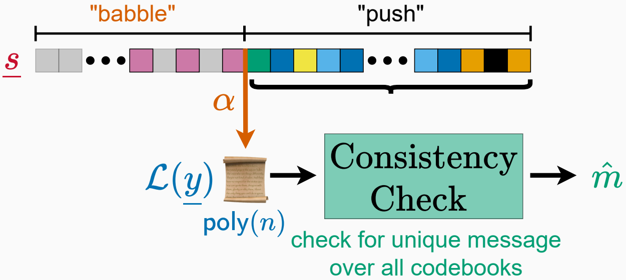
Achievability



Achievable scheme also uses a block structure:

- Alice encodes m using independent randomness in each chunk.
- Bob list decodes after each chunk assuming a random attack $\{V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}\}$.
- Stop if there is $\hat{m}$ and $\underline{s}$ s.t. observed $\underline{y}$ is “feasible.” Else try another attack.

Achievability



Achievable scheme also uses a block structure:

- Alice encodes m using independent randomness in each chunk.
- Bob list decodes after each chunk assuming a random attack $\{V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}\}$.
- Stop if there is $\hat{m}$ and $\underline{s}$ s.t. observed $\underline{y}$ is “feasible.” Else try another attack.

Basically have to define what “feasible” means in this setting (quite involved).

A multi-letter block characterization

Pros and cons:

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\left\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}), \right.$$

$$\left. \min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}},V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_{\alpha}(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}) \right\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\left\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}), \right.$$

$$\left. \min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_{\alpha}(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}) \right\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.
- ✔ Significantly generalizes prior arguments to general channels.

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\left\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}), \right.$$

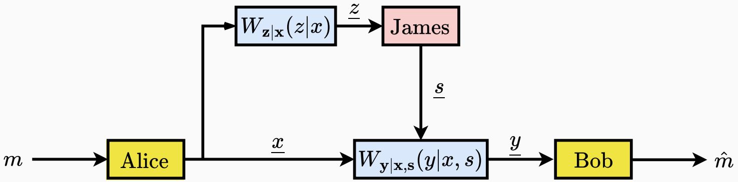
$$\left. \min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_{\alpha}(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}) \right\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.
- ✔ Significantly generalizes prior arguments to general channels.
- ✔ Plotkin results may be useful elsewhere.

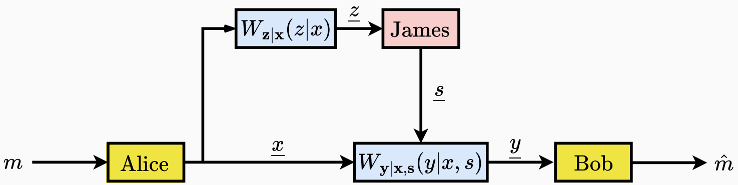
Myopic adversarial models

Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

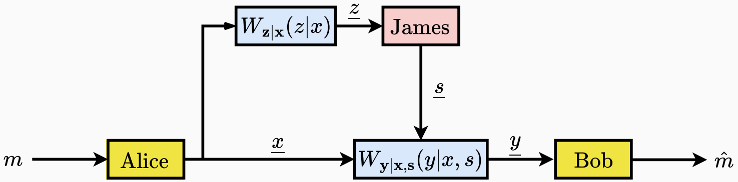
Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

- Jamming strategies are maps $[M] \times \mathcal{Z}^n \to \mathcal{S}^n$.

Myopic adversaries: James sees the whole codeword in noise

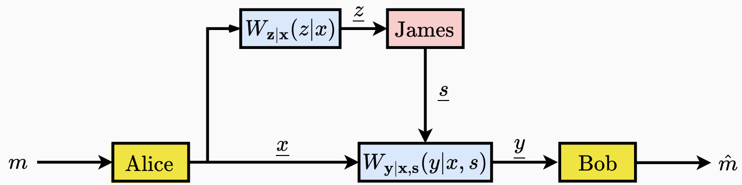


In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

- Jamming strategies are maps $[M] \times \mathcal{Z}^n \to \mathcal{S}^n$.
- For randomized codes we can again look for the worst DMC:

$$\sum_{s,z} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}}(z|x) V_{\mathbf{s}|\mathbf{z}}(s|z).$$

Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

- Jamming strategies are maps $[M] \times \mathcal{Z}^n \to \mathcal{S}^n$.
- For randomized codes we can again look for the worst DMC:

$$\sum_{s,z} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}}(z|x) V_{\mathbf{s}|\mathbf{z}}(s|z).$$

- By changing $W_{\mathbf{z}|\mathbf{x}}$ we can recover the oblivious and omniscient settings.

Symmetrizability for myopic AVCs

A myopic AVC is said to be **symmetrizable** under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

Symmetrizability for myopic AVCs

A myopic AVC is said to be **symmetrizable** under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

$$\sum_{z,s} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$$
$$= \sum_{z',s'} P_{\mathbf{x}}(x') W_{\mathbf{z}|\mathbf{x}}(z'|x') U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x, s'|z') W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x', s'),$$

Symmetrizability for myopic AVCs

A myopic AVC is said to be **symmetrizable** under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

$$\sum_{z,s} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x, s)$$
$$= \sum_{z',s'} P_{\mathbf{x}}(x') W_{\mathbf{z}|\mathbf{x}}(z'|x') U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x, s'|z') W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x', s'),$$

and the resulting marginal state distribution given by

$$P_{\mathbf{s}}(s) = \sum_{x,z,x'} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) \in \Lambda$$

Symmetrizability for myopic AVCs

A myopic AVC is said to be **symmetrizable** under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

$$\sum_{z,s} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x, s)$$
$$= \sum_{z',s'} P_{\mathbf{x}}(x') W_{\mathbf{z}|\mathbf{x}}(z'|x') U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x, s'|z') W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x', s'),$$

and the resulting marginal state distribution given by

$$P_{\mathbf{s}}(s) = \sum_{x,z,x'} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) \in \Lambda$$

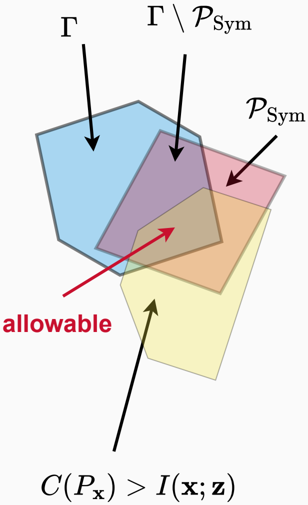
This means **some input distributions are disallowed**:

$$\mathcal{P}_{\mathrm{Sym}} = \{P_{\mathbf{x}} \in \Gamma : P_{\mathbf{x}} \text{ is symmetrizable}\}.$$

Sufficient myopia and achievability

James can create an "effective DMC"

$$\mathcal{W} = \{W_{\mathbf{y}|\mathbf{x}}(y|x) = \sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}}(z|x) V_{\mathbf{s}|\mathbf{z}}(s|z)\}.$$



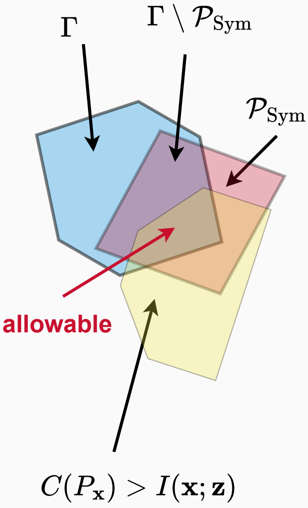
Sufficient myopia and achievability

James can create an "effective DMC"

$$\mathcal{W} = \{W_{\mathbf{y}|\mathbf{x}}(y|x) = \sum_{s} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}}(z|x) V_{\mathbf{s}|\mathbf{z}}(s|z)\}.$$

Alice/Bob cannot use any $P_{\mathbf{x}} \in \mathcal{P}_{\text{Sym}}$. If they choose $P_{\mathbf{x}} \in \Gamma \setminus \mathcal{P}_{\text{Sym}}$ they could target the mutual information

$$C(P_{\mathbf{x}}) = \min_{\mathcal{W}} I(\mathbf{x};\mathbf{y}).$$



Sufficient myopia and achievability

James can create an "effective DMC"

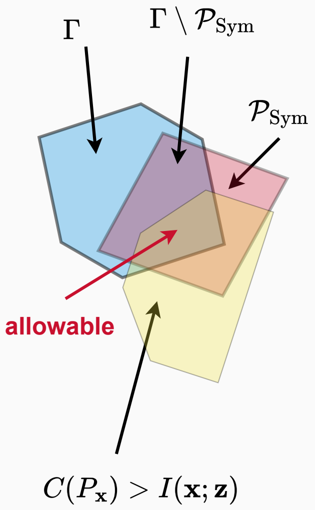
$$\mathcal{W} = \{W_{\mathbf{y}|\mathbf{x}}(y|x) = \sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}}(z|x) V_{\mathbf{s}|\mathbf{z}}(s|z)\}.$$

Alice/Bob cannot use any $P_{\mathbf{x}} \in \mathcal{P}_{\mathrm{Sym}}$. If they choose $P_{\mathbf{x}} \in \Gamma \setminus \mathcal{P}_{\mathrm{Sym}}$ they could target the mutual information

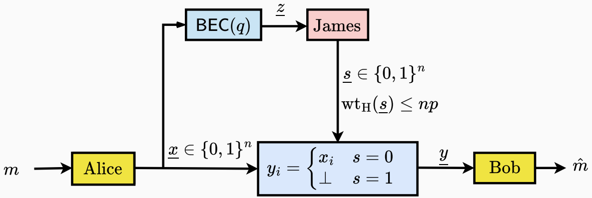
$$C(P_{\mathbf{x}}) = \min_{\mathcal{W}} I(\mathbf{x};\mathbf{y}).$$

If $I(\mathbf{z};\mathbf{x}) < C(P_{\mathbf{x}})$ we say James is **sufficiently myopic**. In that case we can achieve any rate

$$R < \max_{P_{\mathbf{x}} \in \Gamma \setminus \mathcal{P}_{\mathrm{Sym}}} C(P_{\mathbf{x}}).$$



Myopic adversaries in the erasure setting

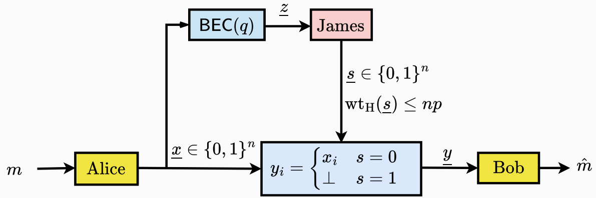


In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

Myopic adversaries in the erasure setting

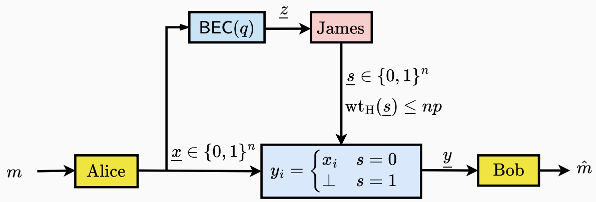
If $p < q$ (sufficiently myopic),

$$C = 1 - p.$$



In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

Myopic adversaries in the erasure setting



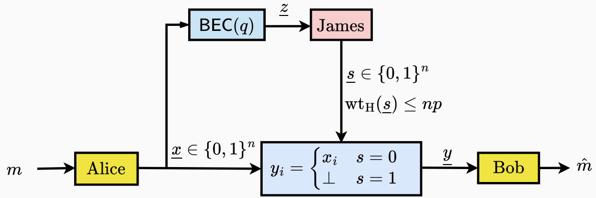
In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

If $p < q$ (sufficiently myopic),

$$C = 1 - p.$$

If $p > q$ we have two cases:

Myopic adversaries in the erasure setting



In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

If $p < q$ (sufficiently myopic),

$$C = 1 - p.$$

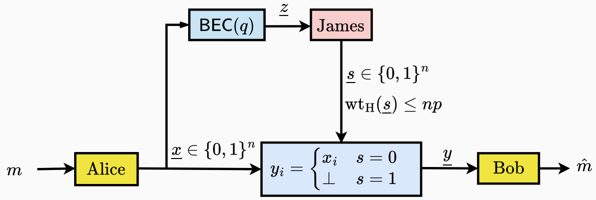
If $p > q$ we have two cases:

1. If $q > 2p - 1$,

$$C \in \left(0, (1-q)\bar{\alpha}\left(\frac{p-q}{1-q}\right)\right],$$

where $\bar{\alpha}$ is the LP bound for normalized distance.

Myopic adversaries in the erasure setting



In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

If $p < q$ (sufficiently myopic),

$$C = 1 - p.$$

If $p > q$ we have two cases:

1. If $q > 2p - 1$,

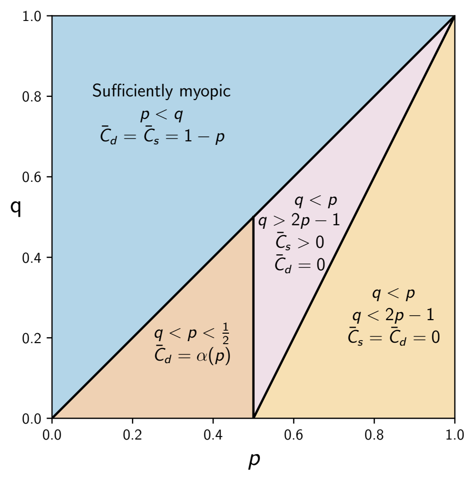
$$C \in \left(0, (1-q)\bar{\alpha}\left(\frac{p-q}{1-q}\right)\right],$$

where $\bar{\alpha}$ is the LP bound for normalized distance.

2. If $q < 2p - 1$,

$$C = 0.$$

Myopic adversaries in the erasure setting



Computationally efficient codes for causal adversaries

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

- **random codes** are inefficient to decode but **linear codes** are too easy jam!
  $\longrightarrow$ use a **library of linear codebooks**.

Main questions we address

Can we design **efficient codes** **for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

- **random codes** are inefficient to decode but **linear codes** are too easy jam!
  $\longrightarrow$ use a **library of linear codebooks**.
- **common randomness** is unrealistic.
  $\longrightarrow$ use **limited encoder randomization** to **confuse the adversary**.

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

- **random codes** are inefficient to decode but **linear codes** are too easy jam!
  $\longrightarrow$ use a **library of linear codebooks**.
- **common randomness** is unrealistic.
  $\longrightarrow$ use **limited encoder randomization** to **confuse the adversary**.
- **minimum distance coding** is not efficient in general.
  $\longrightarrow$ use **list decoding** to permit **efficient decoding**.

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a
- **library of random linear codes** and

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a
- **library of random linear codes** and
- uses **list decoding** to reduce the search space

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a
- **library of random linear codes** and
- uses **list decoding** to reduce the search space

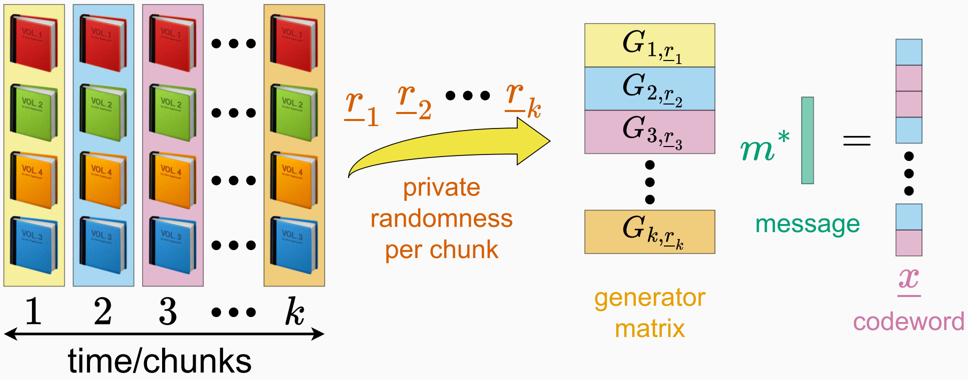
There are different types of complexity we would like to control:

- **Design**: how many bits do we need to generate the code?
- **Storage**: how many bits do we need to store the code?
- **Encoding**: how many operations are needed to encode a message?
- **Decoding**: how many operations are needed to decode the message?

Main results

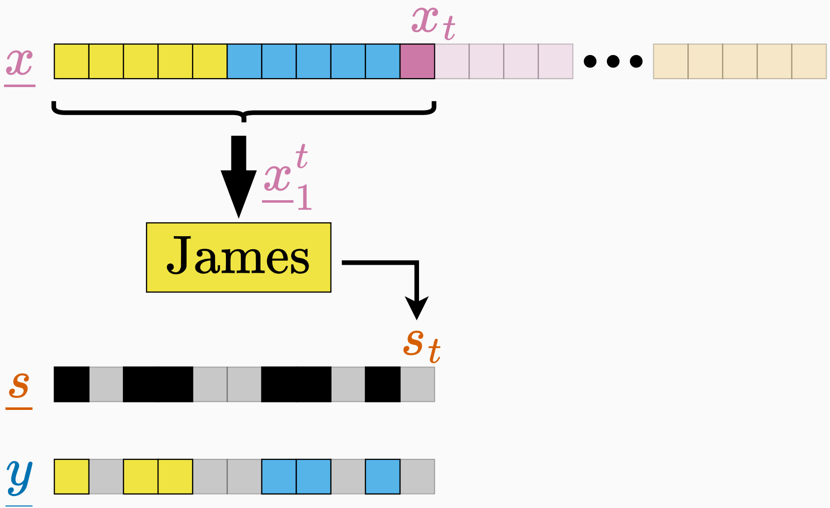
Model **rate**	Randomness	Enc/Storage	Decoding	$\mathbf{P}_{\text{error}}$
Causal $\mathbf{1-2p-\epsilon}$	$O\left(\frac{\gamma \log n}{\epsilon}\right)$	$O(n^3 \log\log n)$	$O(n^{32/\epsilon})$	$O(n^{-(\gamma-1)})$
Myopic $p<q$ $\mathbf{1-p-\epsilon}$	$\lambda_{SM}\log(n)$	$O(n^{2+\lambda_{SM}})$	$O(n^{3+\lambda_{SM}})$	$O(n^{-\lambda_{SM}})$
Myopic $q<p$ **small rate**	$O(n \log\log n)$	$O(n^2 \log\log n)$	$O(n^3 \log\log n)$	$O(n^{-4/5})$

Encode splits block into a constant $k = \lceil \frac{n}{\epsilon} \rceil$ chunks

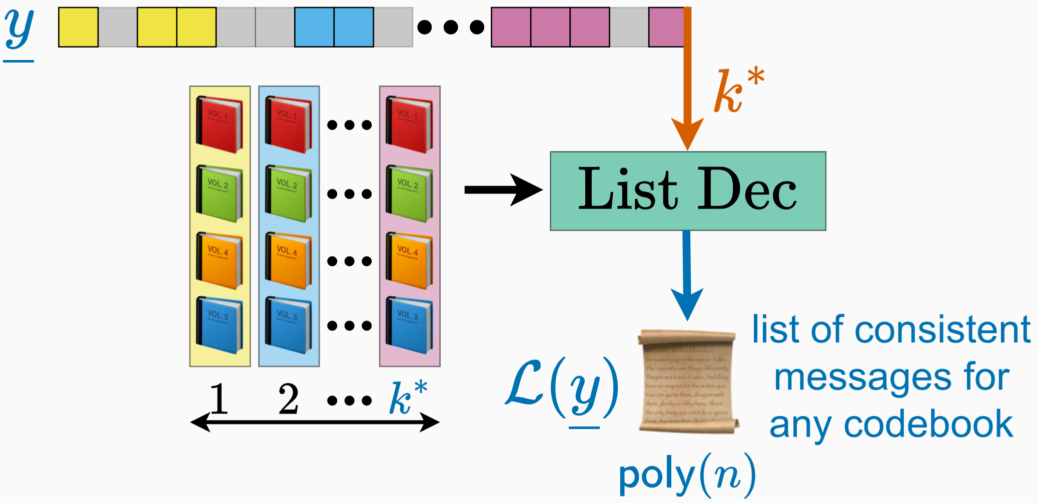


Generate a **library of linear codebooks** independently for each chunk.

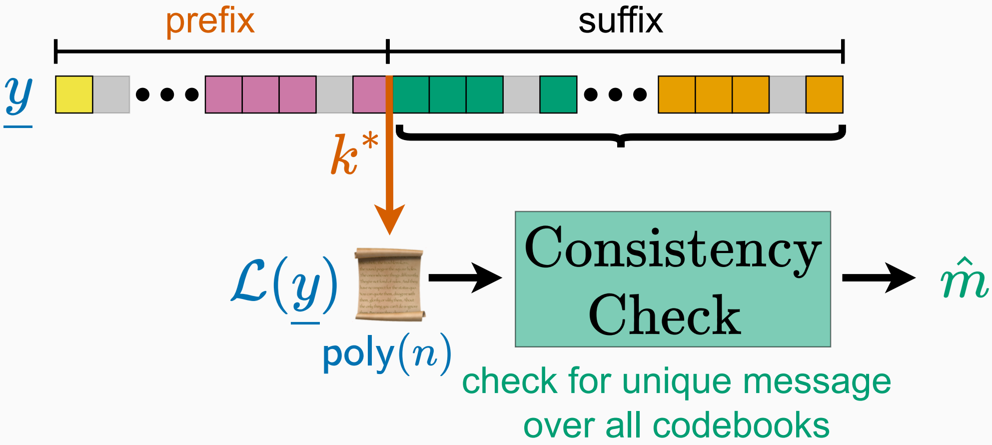
James can erase with causal information only



Bob decodes to a polynomial list



Bob uses suffix to disambiguate the list

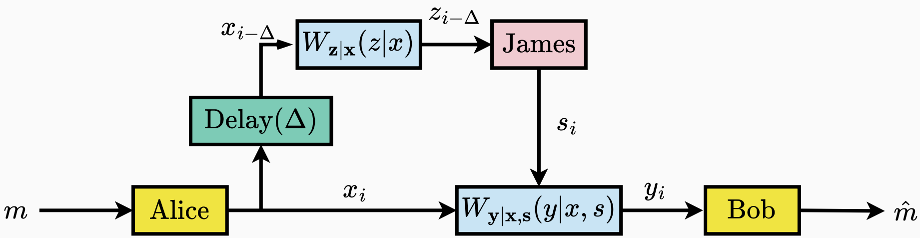


Why does this work?

1. Bob can **track James's erasure budget**.
2. List decoding creates **a smaller set of messages** to check for consistency.
3. James has a choice to **make the list larger** (erase more earlier, less later) or **conserve his budget** (erase less earlier, more later).
4. **Poor James, he can't win.**

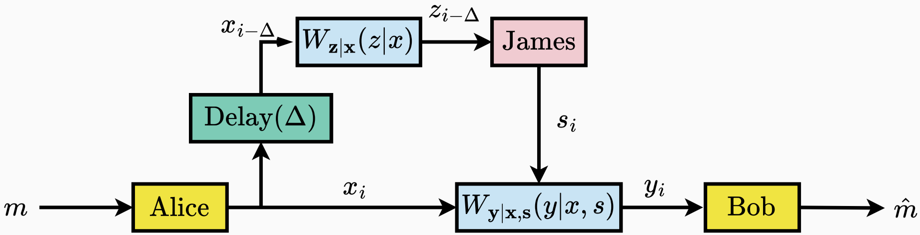
Looking forward

Other intermediate models



There are lots of other **intermediate models** one could look at:

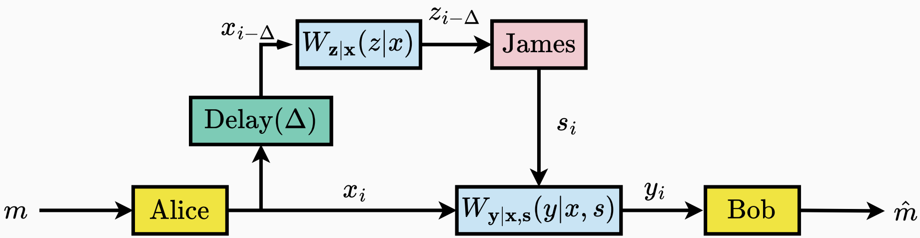
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!

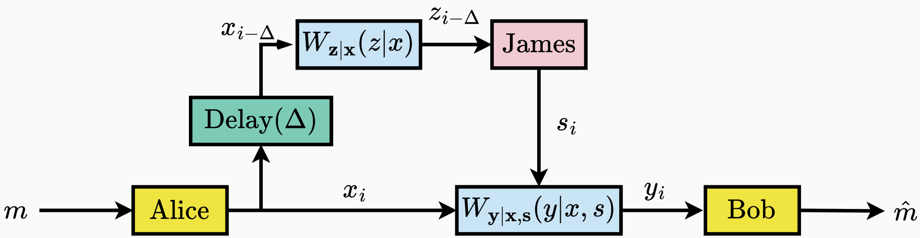
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)

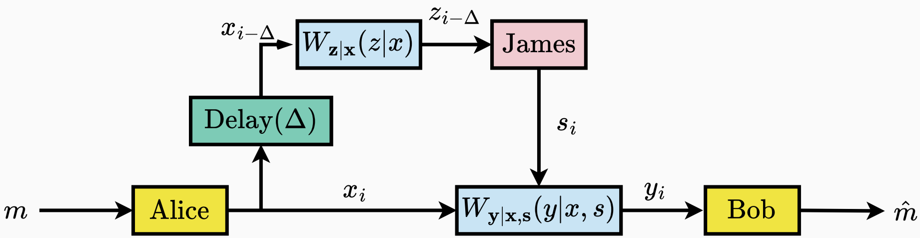
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)
- Allow James to pick a fraction of locations to observe before acting.

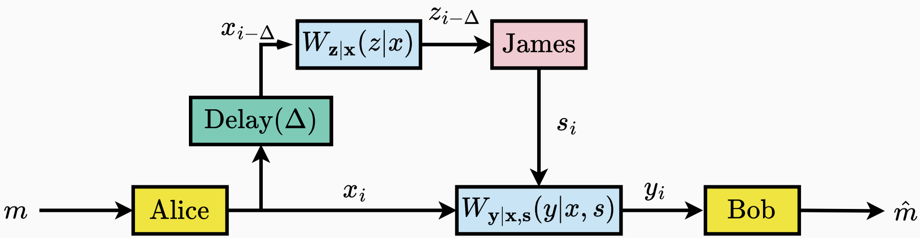
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)
- Allow James to pick a fraction of locations to observe before acting.
- Etc. etc.

Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)
- Allow James to pick a fraction of locations to observe before acting.
- Etc. etc.

Each model will reveal something about what the **worst-case channel** looks like.

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models
- adversarial machine learning

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

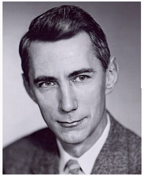
- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models
- adversarial machine learning
- extremal graph theory

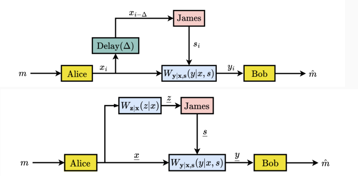
And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models
- adversarial machine learning
- extremal graph theory
- other fun combinatorial problems

A final recap and takeaways





AVCs can capture models between average and worst-case channels.

- **Causal:** capacity depends on **what James knows about the current input.**
- **Myopic:** capacity depends on **whether James can (partially) "decode."**

Some insights:

- **Stochastic encoding** and **list decoding** can help!
- Worst-case attacks are ones that "push" **at the end of decoding**.

Thank you!