On the path to minimal-length descriptions, guided by Kolia and Sasha

Marius Zimand (Towson University)

Moscow, June 2019

Some good books.

This talk in one slide

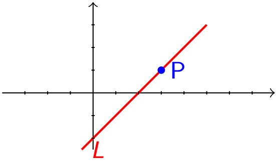
Common wisdom: Kolmogorov complexity is about optimal compression/decompression, where compression is not effective, but decompression is.

> *"... in the framework of Kolmogorov complexity we have no compression algorithm and deal only with decompression algorithms."*
>
> — Alexander Shen, Around Kolmogorov complexity: basic notions and results, 2015.

We shall see natural circumstances where compression to close to minimum description length is not only effective but actually **efficient** (and decompression is effective but not efficient).

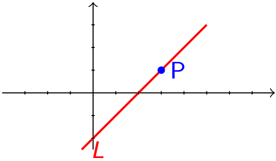
A preparatory puzzle

- Kolia and Sasha want to agree on a secret key.
- Problem is that we hear everything they say.
- Kolia knows line $L : y = a_1 x + a_0$;
  Sasha knows point P: (b_1, b_2);
- L : $2n$ bits of information (intercept, slope in $\mathbb{F}_{2^n}$).
- P: $2n$ bits of information (the 2 coord. in $\mathbb{F}_{2^n}$).
- Total information in $(L, P) = 3n$ bits; mutual information of L and $P = n$ bits.



A preparatory puzzle

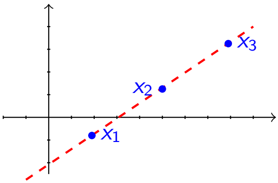
- Kolia and Sasha want to agree on a secret key.
- Problem is that we hear everything they say.
- Kolia knows line $L : y = a_1 x + a_0$;
  Sasha knows point P: (b_1, b_2);
- L : $2n$ bits of information (intercept, slope in $\mathbb{F}_{2^n}$).
- P: $2n$ bits of information (the 2 coord. in $\mathbb{F}_{2^n}$).
- Total information in $(L, P) = 3n$ bits; mutual information of L and $P = n$ bits.



SOLUTION:

- Kolia tells a_1 to Sasha.
- Sasha, knowing that $P \in L$, finds L.
- Kolia and Sasha use a_0 as a secret key.
- It works! We have heard a_1, but a_1 and a_0 are independent.

The real puzzle

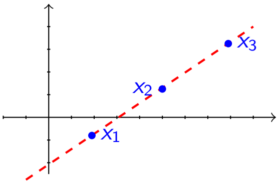


Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

The real puzzle



Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

Kolmogorov complexity: measuring information in a string

$C(x) :=$ length(shortest description of x)

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

:= size of a shortest program generating x

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

:= size of a shortest program generating x

Formal Definition:

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

:= size of a shortest program generating x

Formal Definition:

chose an algorithm $\mathcal{A}$

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

:= size of a shortest program generating x

Formal Definition:

chose an algorithm $\mathcal{A}$

$C_{\mathcal{A}}(x) := \min\{\text{length}(p) \ : \ \mathcal{A}(p) = x\}$

p is called a program for x if $\mathcal{A}(p) = x$.

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

 := size of a shortest program generating x

Formal Definition:

chose an algorithm $\mathcal{A}$

$C_{\mathcal{A}}(x) := \min\{\text{length}(p) \; : \; \mathcal{A}(p) = x\}$

p is called a program for x if $\mathcal{A}(p) = x$.

Invariance Theorem:

There exists an optimal $\mathcal{U}$

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

:= size of a shortest program generating x

Formal Definition:

chose an algorithm $\mathcal{A}$

$C_{\mathcal{A}}(x) := \min\{\text{length}(p) \ : \ \mathcal{A}(p) = x\}$

p is called a program for x if $\mathcal{A}(p) = x$.

Invariance Theorem:

There exists an <u>optimal</u> $\mathcal{U}$

such that $C_{\mathcal{U}}(x) \leq C_{\mathcal{A}}(x) + O(1)$ for all other $\mathcal{A}$

Kolmogorov complexity: measuring information in a string

$C(x)$:= length(shortest description of x)

:= size of a shortest program generating x

Formal Definition:

chose an algorithm $\mathcal{A}$

$C_{\mathcal{A}}(x) := \min\{\text{length}(p) \ : \ \mathcal{A}(p) = x\}$

p is called a program for x if $\mathcal{A}(p) = x$.

Invariance Theorem:

There exists an optimal $\mathcal{U}$

such that $C_{\mathcal{U}}(x) \leq C_{\mathcal{A}}(x) + O(1)$ for all other $\mathcal{A}$

We fix some optimal $\mathcal{U}$ once and forever.

Kolmogorov complexity: measuring information in a string

$C(x)$:= size of a shortest program generating x

Kolmogorov complexity: measuring information in a string

$C(x)$:= size of a shortest program generating x

$$x = \underbrace{110111001\ldots101}_{n \text{ bits}}$$

x has a description of length $n + O(1)$.

Kolmogorov complexity: measuring information in a string

$C(x)$:= size of a shortest program generating x

$$x = \underbrace{110111001\ldots101}_{n \text{ bits}}$$

x has a description of length $n + O(1)$.

- $C(x) \leq n + \text{const}$ for all x of length n

Kolmogorov complexity: measuring information in a string

$C(x)$:= size of a shortest program generating x

$$x = \underbrace{110111001\ldots101}_{n \text{ bits}}$$

x has a description of length $n + O(1)$.

- $C(x) \leq n + \text{const}$ for <u>all</u> x of length n
- $C(x) \geq n - \text{const}$ for <u>most</u> x of length n

Kolmogorov complexity: measuring information in a string

$C(x)$:= size of a shortest program generating x

$$x = \underbrace{110111001\ldots101}_{n \text{ bits}}$$

x has a description of length $n + O(1)$.

- $C(x) \leq n + \text{const}$ for <u>all</u> x of length n
- $C(x) \geq n - \text{const}$ for <u>most</u> x of length n

$$x = \underbrace{000000000\ldots000}_{n \text{ bits}}$$

$C(x) \leq \log n + O(1)$

Information quantities for two strings x, y

- $C(x) :=$ length(shortest description of x)
  $:=$ size of a shortest program generating x

  Other quantities: $C(y)$, $C(x, y)$

Information quantities for two strings x, y

- $C(x)$:= length(shortest description of x)
  := size of a shortest program generating x

  Other quantities: $C(y)$, $C(x, y)$

- $C(x \mid y)$:= length(shortest description of x given y)
  := size of a shortest program generating x given y

  Another quantity: $C(y \mid x)$

Information quantities for two strings x, y

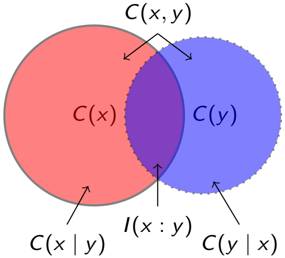
- $C(x)$:= length(shortest description of x)
  := size of a shortest program generating x

  Other quantities: $C(y)$, $C(x, y)$

- $C(x \mid y)$:= length(shortest description of x given y)
  := size of a shortest program generating x given y

  Another quantity: $C(y \mid x)$

- Mutual information of x and y :
  $I(x : y) := C(x) - C(x \mid y)$.



Information quantities for two strings x, y

- $C(x) :=$ length(shortest description of x)
  $:=$ size of a shortest program generating x

  Other quantities: $C(y)$, $C(x, y)$

- $C(x \mid y) :=$ length(shortest description of x given y)
  $:=$ size of a shortest program generating x given y

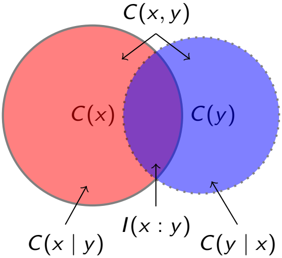
  Another quantity: $C(y \mid x)$

- Mutual information of x and y :
  $I(x : y) := C(x) - C(x \mid y)$.

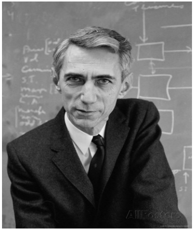
- **Chain Rule** [Kolmogorov, Levin] $C(x, y) =^{+} C(x) + C(y \mid x)$

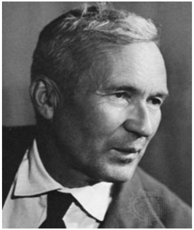
  where the notation $=^{+}$ hides $\pm O(\log n)$

Corollary. $I(x : y) =^{+} C(x) + C(y) - C(x, y) =^{+} I(y : x)$



IT vs. AIT (or Shannon vs. Kolmogorov)





The word **random** is used in computer science in two ways:

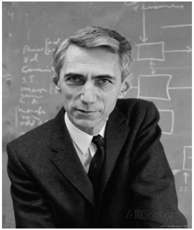
(1) **random** process: a process whose outcome is uncertain, e.g. a series of coin tosses.

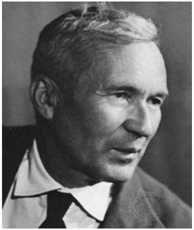
(2) **random** object: something that lacks regularities, patterns, is incompressible.

Information Theory (IT) focuses on (1).

Algorithmic Information Theory (AIT, also known as Kolmogorov complexity) focuses on (2).

IT vs. AIT





IT (à la Shannon)

- Data is the realization of a random variable X.
- The model: a stochastic process generates the data.
- Amount of information in the data:

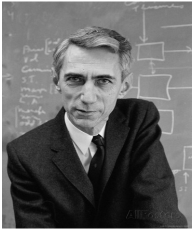
$H(X) = \sum p_i \log(1/p_i)$ (Shannon entropy).

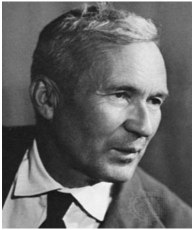
AIT (Kolmogorov complexity)

- Data is just an individual string x
- There is no generative model.
- Amount of information in the data:

$C(x) =$ minimum description length.

IT vs. AIT





IT (à la Shannon)

- Data is t… n of a ra… X.
- The model: a stochastic process generates the data.
- Amount of information in the data:

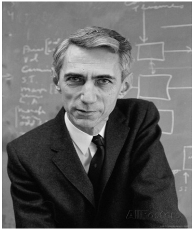
$H(X) = \sum p_i \log(1/p_i)$ (Shannon entropy).

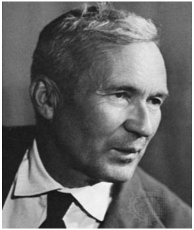
AIT (Kolmogorov complexity)

- … just an individual string x
- There is no generative model.
- Amount of information in the data:

$C(x)$ = minimum description length.

IT vs. AIT





IT (à la Shannon)

- Data is the realization of a random variable X.
- Th... cess gene... the data.
- Amount of information in the data: $H(X) = \sum p_i \log(1/p_i)$ (Shannon entropy).

(1/4,1/4,1/4,1/4)

AIT (Kolmogorov complexity)

- Data is just an individual string x
- ...enerative model.
- Amount of information in the data: $C(x) =$ minimum description length.

101101000110010

Short programs and communication protocols

Alice has x Bob has y.

They run an interactive protocol.

$$\begin{array}{c} \xrightarrow{\quad t_1 \quad} \\ \xleftarrow{\quad t_2 \quad} \\ \xrightarrow{\quad t_3 \quad} \\ \vdots \\ \xrightarrow{\quad t_k \quad} \end{array}$$

Bob has x

QUESTION: What is the communication complexity?

Can it be $C(x \mid y)$? Is there a protocol that comes close to this?

Scenario: Alice and Bob are computationally unbounded

Alice has x, Bob has y. They run a protocol. At the end, Bob has x.

- If the protocol is **deterministic**, Alice needs to send $C(x)$ bits.

Scenario: Alice and Bob are computationally unbounded

Alice has x, Bob has y. They run a protocol. At the end, Bob has x.

- If the protocol is **deterministic**, Alice needs to send $C(x)$ bits.
- (Buhrman, Koucky, Vereshchagin, 2014) There is a **randomized** protocol with communication complexity $C(x \mid y) + O(\sqrt{C(x \mid y)})$.

Scenario: Alice and Bob are computationally unbounded

Alice has x, Bob has y. They run a protocol. At the end, Bob has x.

- If the protocol is **deterministic**, Alice needs to send $C(x)$ bits.
- (Buhrman, Koucky, Vereshchagin, 2014) There is a **randomized** protocol with communication complexity $C(x \mid y) + O(\sqrt{C(x \mid y)})$.
- The difficult part: Alice needs to find $C(x \mid y)$.

Scenario: Alice and Bob are computationally unbounded

Alice has x, Bob has y. They run a protocol. At the end, Bob has x.

- If the protocol is **deterministic**, Alice needs to send $C(x)$ bits.
- (Buhrman, Koucky, Vereshchagin, 2014) There is a **randomized** protocol with communication complexity $C(x \mid y) + O(\sqrt{C(x \mid y)})$.
- The difficult part: Alice needs to find $C(x \mid y)$.
- (Vereshchagin, 2014) The randomized communication complexity of computing $C(x \mid y)$ with precision ϵn is $0.99n$.

Scenario: Alice is algorithmically bounded

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- A program p for x given y is c-short, if $|p| \leq C(x) + c$.

Scenario: Alice is algorithmically bounded

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- A program p for x given y is c-short, if $|p| \leq C(x) + c$.
- (Bauwens, Makhlin, Vereshchagin, Zimand, 2013) Alice can **effectively compute** on input x a list with $O(n^2)$ elements that contains a $O(1)$-short program for x given y.

Scenario: Alice is algorithmically bounded

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- A program p for x given y is c-short, if $|p| \leq C(x) + c$.
- (Bauwens, Makhlin, Vereshchagin, Zimand, 2013) Alice can **effectively compute** on input x a list with $O(n^2)$ elements that contains a $O(1)$-short program for x given y.
- Such a list must have size $\Omega(n^2)$.

Scenario: Alice is algorithmically bounded

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- A program p for x given y is c-short, if $|p| \leq C(x) + c$.
- (Bauwens, Makhlin, Vereshchagin, Zimand, 2013) Alice can **effectively compute** on input x a list with $O(n^2)$ elements that contains a $O(1)$-short program for x given y.
- Such a list must have size $\Omega(n^2)$.
- (Teutsch, 2014) Alice can compute on input x **in polynomial time** a list with $O(n^2)$ elements that contains a $O(1)$-short program for x given y.

Scenario: Alice is algorithmically bounded

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- A program p for x given y is c-short, if $|p| \leq C(x) + c$.
- (Bauwens, Makhlin, Vereshchagin, Zimand, 2013) Alice can **effectively compute** on input x a list with $O(n^2)$ elements that contains a $O(1)$-short program for x given y.
- Such a list must have size $\Omega(n^2)$.
- (Teutsch, 2014) Alice can compute on input x **in polynomial time** a list with $O(n^2)$ elements that contains a $O(1)$-short program for x given y.
- (Zimand, 2014) The size of the list in Teutsch's result is $O(n^{6+\epsilon})$.

Dagstuhl 2003

Scenario: Alice is algorithmically bounded and holds advice information

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has some information about x and y (called **advice**).

Scenario: Alice is algorithmically bounded and holds advice information

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has some information about x and y (called **advice**).
- **Muchnik's theorem, 2001**: Alice on input x and some $O(\log n)$-long advice can compute a 0-short program for x given y.

Scenario: Alice is algorithmically bounded and holds advice information

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has some information about x and y (called **advice**).
- **Muchnik's theorem, 2001**: Alice on input x and some $O(\log n)$-long advice can compute a 0-short program for x given y.
- (Musatov, Romashchenko, Shen, 2009) **Space-bounded version of Muchnik's Th.**:

  For every space bound s, Alice on x and some $O(\log^3 n)$-long advice can compute in **polynomial space** a program p for x given y with space complexity $O(s) + \mathrm{poly}(n)$ and $|p| = C^{\mathrm{space}=s}(x \mid y) + O(\log n)$.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.
- Consider the special case $y =$ empty string.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.
- Consider the special case $y =$ empty string.
- Alice on input x and $C(x)$ can find a 0-short program for x by exhaustive search, but this is VERY SLOW.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.
- Consider the special case $y =$ empty string.
- Alice on input x and $C(x)$ can find a 0-short program for x by exhaustive search, but this is VERY SLOW.
- (Bauwens, Zimand, 2014) Let t be any computable function. If an algorithm on input $(x, C(x))$ finds a program p for x in time $t(n)$, then for infinitely many x, $|p| = C(x) + \Omega(n)$.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.
- Consider the special case $y =$ empty string.
- Alice on input x and $C(x)$ can find a 0-short program for x by exhaustive search, but this is VERY SLOW.
- (Bauwens, Zimand, 2014) Let t be any computable function. If an algorithm on input $(x, C(x))$ finds a program p for x in time $t(n)$, then for infinitely many x, $|p| = C(x) + \Omega(n)$.
- (Bauwens, Zimand, 2014) Alice on input $(x, C(x))$ can compute in **probabilistic polynomial time** a $O(\log^2(n/\epsilon))$-short program for x given y, with probability error ϵ.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.
- Consider the special case $y =$ empty string.
- Alice on input x and $C(x)$ can find a 0-short program for x by exhaustive search, but this is VERY SLOW.
- (Bauwens, Zimand, 2014) Let t be any computable function. If an algorithm on input $(x, C(x))$ finds a program p for x in time $t(n)$, then for infinitely many x, $|p| = C(x) + \Omega(n)$.
- (Bauwens, Zimand, 2014) Alice on input $(x, C(x))$ can compute in **probabilistic polynomial time** a $O(\log^2(n/\epsilon))$-short program for x given y, with probability error ϵ.
- If we drop the poly time requirement, the overhead can be reduced to $O(\log n)$.

Scenario: Alice is algorithmically bounded and knows $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has ~~some information about x and y~~ $C(x|y)$.
- Consider the special case y = empty string.
- Alice on input x and $C(x)$ can find a 0-short program for x by exhaustive search, but this is VERY SLOW.
- (Bauwens, Zimand, 2014) Let t be any computable function. If an algorithm on input $(x, C(x))$ finds a program p for x in time $t(n)$, then for infinitely many x, $|p| = C(x) + \Omega(n)$.
- (Bauwens, Zimand, 2014) Alice on input $(x, C(x))$ can compute in **probabilistic polynomial time** a $O(\log^2(n/\epsilon))$-short program for x given y, with probability error ϵ.
- If we drop the poly time requirement, the overhead can be reduced to $O(\log n)$.
- The overhead cannot be less than $\log n - \log\log n - O(1)$, for total computable compressors.

Scenario: Alice is algorithmically bounded and knows an upper bound of $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has $\cancel{C(x|y)}$ $m \geq C(x \mid y)$

Scenario: Alice is algorithmically bounded and knows an upper bound of $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has $\cancel{C(x|y)}$ $\quad m \geq C(x \mid y)$
- Consider the special case $y =$ empty string.

Scenario: Alice is algorithmically bounded and knows an upper bound of $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has $\cancel{C(x|y)}$ $\quad m \geq C(x \mid y)$
- Consider the special case $y =$ empty string.
- Alice on input x and k can find a program p for x with $|p| \leq m$ by exhaustive search.

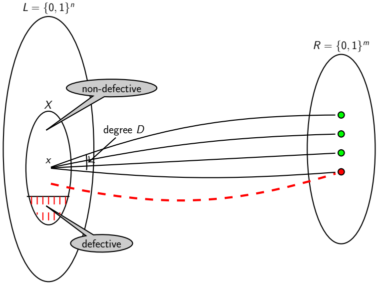
Scenario: Alice is algorithmically bounded and knows an upper bound of $C(x \mid y)$.

Alice has x, Bob has y. Alice wants a program for x given y (which she can send to Bob, to communicate x).

- Assumption: Besides x, Alice has $\cancel{C(x|y)}$ $\quad m \geq C(x \mid y)$
- Consider the special case $y =$ empty string.
- Alice on input x and k can find a program p for x with $|p| \leq m$ by exhaustive search.
- (Zimand, 2017) (Bauwens, Zimand, 2019) Alice on input (x, m) can compute in **probabilistic polynomial time** a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.

Theorem (Bauwens, Zimand, 2019)

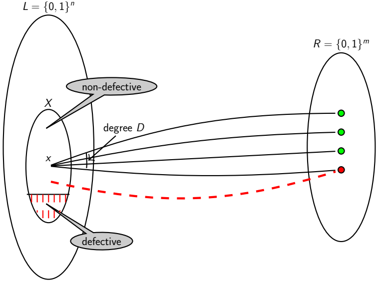
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- $f : L \times [D] \to R$, used for fingerprinting.
- $f(x, 1), \ldots, f(x, D)$ are the fingerprints of x.
- X is the list of candidates, we want to identify which candidate is x.
- A fingerprint is heavy for X, if it has more $2D$ pre-images in X.
- x is ϵ-defective for X if it has more than ϵD heavy fingerprints.

Theorem (Bauwens, Zimand, 2019)

(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



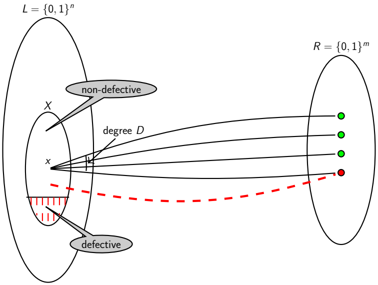
- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser , if for every r.v. X with min - entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.

Theorem (Bauwens, Zimand, 2019)

(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*

$\Pr[X = x] \leq 2^{-k}$ for all x

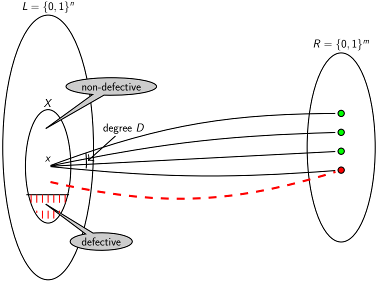
- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser , if for every r.v. X with min-entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.



Theorem (Bauwens, Zimand, 2019)

(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probabili*

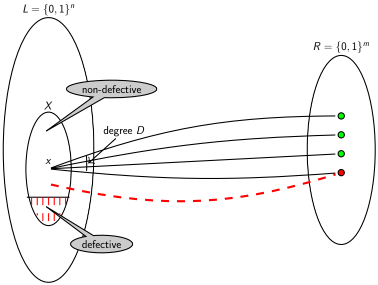
Buhrman, Fortnow, Laplante 2001
used extractors for a related problem



- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser, if for every r.v. X with min - entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.

Theorem (Bauwens, Zimand, 2019)

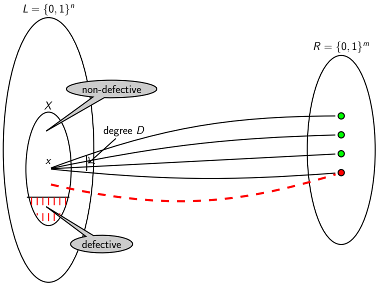
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser , if for every r.v. X with min - entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.
- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is an ϵ conductor, if it is a $k \to_\epsilon k$ condenser for every $k \leq m$.

Theorem (Bauwens, Zimand, 2019)

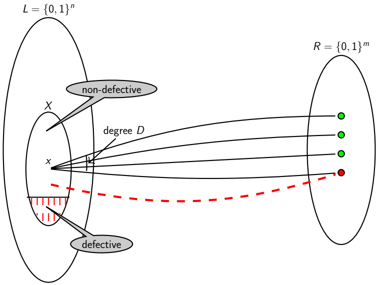
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser , if for every r.v. X with min - entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.
- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is an ϵ conductor, if it is a $k \to_\epsilon k$ condenser for every $k \leq m$.
- If $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is an ϵ conductor, for every X, the fraction of 4ϵ-defective elements in X is at most $1/2$

Theorem (Bauwens, Zimand, 2019)

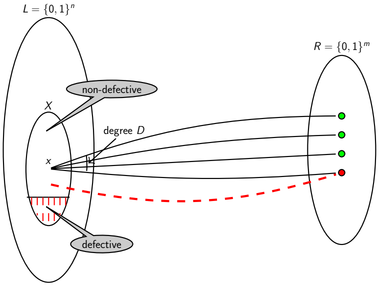
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser , if for every r.v. X with min -entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.
- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is an ϵ conductor, if it is a $k \to_\epsilon k$ condenser for every $k \leq m$.
- If $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is an ϵ conductor, for every X, the fraction of 4ϵ-defective elements in X is at most $1/2$
- (Bauwens, Zimand 2019) There exists poly-time ϵ conductor with $D = 2^{\log^2(n/\epsilon)}$, for every $m \leq n$.

Theorem (Bauwens, Zimand, 2019)

(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*

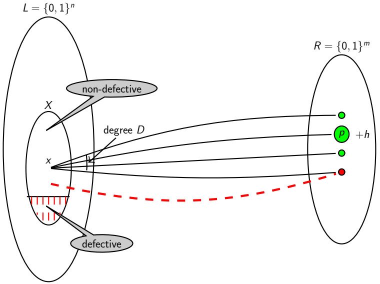


- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$ is a $k \to_\epsilon k$ condenser , if for every r.v. X with min - entropy k, $f(X, U_D)$ is ϵ-close to having min-entropy k.
- $f : \{0,1\}^n \times [D]$ ... if it is a $k \to_\epsilon$ ...
- If $f : \{0,1\}^n$ ... for every X, the fra... elements in X is ... most $1/2$
- (Bauwens, Zimand 2019) There exists poly-time ϵ conductor with $D = 2^{\log^2(n/\epsilon)}$, for every $m \leq n$.

building on the Guruswami-Umans-Vadhan extractor

Theorem (Bauwens, Zimand, 2019)

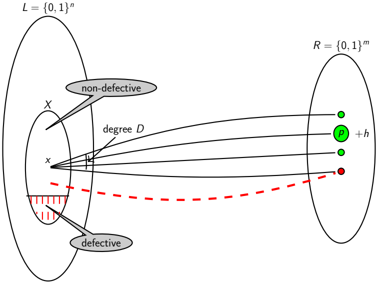
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- x has complexity $C(x) \leq m$.
- $f : \{0,1\}^n \times [D] \to \{0,1\}^m$, the explicit ϵ-conductor.
- x has complexity $C(x) \leq m$.
- $f(x, 1), \ldots, f(x, D)$ are the fingerprints of x.
- **Compress x:** pick randomly p, one of the fingerprints. Append h, a short hash-code of x. Output (p, h). Length: $m + |h|$.
- **Decompression:** we want to reconstruct x from (p, h).
- X the set of strings with complexity $\leq m$ (list of candidates). we want to identify which candidate is x.

Theorem (Bauwens, Zimand, 2019)

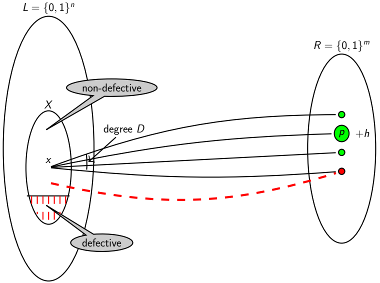
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- Decompression: we want to reconstruct x from (p, h).
- X the set of strings with complexity $\leq m$ (list of candidates). We want to identify which candidate is x.
- Case 1: x is non-defective. With prob $1 - \epsilon$, we reduce the list of candidates to the $2D$-preimages of p.
- Case 2: x is defective. We reduce the list of candidates to the set of defective elements, so we reduce the list by $1/2$.
- Continue recursively with fewer candidates.

Theorem (Bauwens, Zimand, 2019)

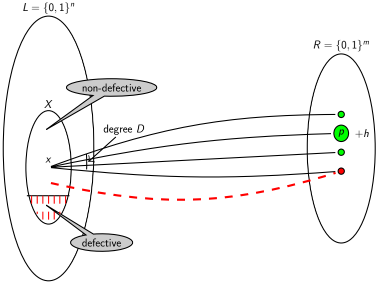
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- Decompression: we want to reconstruct x from (p, h).
- X the set of strings with complexity $\leq m$ (list of candidates). We want to identify which candidate is x.
- Case 1: x is non-defective. With prob $1 - \epsilon$, we reduce the list of candidates to the $2D$-preimages of p.
- Case 2: x is defective. We reduce the list of candidates to the set of defective elements, so we reduce the list by $1/2$.
- Continue recursively with fewer candidates.
- Problem: We do not know which of Case 1 or Case 2 is true.

Theorem (Bauwens, Zimand, 2019)

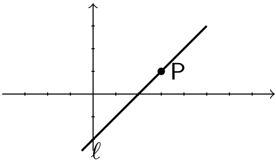
(Bauwens, Zimand, 2019) Alice on input (x, m), where $m \geq C(x \mid y)$, can compute in **probabilistic polynomial time** *a program for x given y of length $m + O(\log^2(n/\epsilon))$, with probability error ϵ.*



- Case 1: x is non-defective. With prob $1 - \epsilon$, we reduce the list of candidates to the $2D$-preimages of p.
- Case 2: x is defective. We reduce the list of candidates to the set of defective elements, so we reduce the list by $1/2$.
- Problem: We do not know which of Case 1 or Case 2 is true.
- We collect the candidates as if Case 1 is true, so we keep only the first $2D$ preimages of p. Then reduce as in Case 2.
- At the end we have collected $m \times 2D$ candidates.
- We identify x using h, the short hash code.

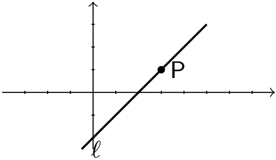
Distributed compression: a simple example

- Alice knows a line ℓ; Bob knows a point $P \in \ell$; They want to send ℓ and P to Zack.
- ℓ : $2n$ bits of information (intercept, slope in GF[2^n]).
- P : $2n$ bits of information (the 2 coord. in GF[2^n]).
- Total information in $(\ell, P) = 3n$ bits; mutual information of ℓ and $P = n$ bits.
- If Alice and Bob get together, they need to send $3n$ bits. What if they compress separately?



Distributed compression: a simple example

- Alice knows a line ℓ; Bob knows a point $P \in \ell$; They want to send ℓ and P to Zack.
- ℓ : $2n$ bits of information (intercept, slope in GF[2^n]).
- P : $2n$ bits of information (the 2 coord. in GF[2^n]).
- Total information in $(\ell, P) = 3n$ bits; mutual information of ℓ and $P = n$ bits.
- If Alice and Bob get together, they need to send $3n$ bits. What if they compress separately?



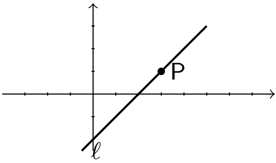
QUESTION 1:

Alice can send $2n$ bits, and Bob n bits. Is the geometric correlation between ℓ and P crucial for these compression lengths?

Ans: No. Same is true (modulo a $\mathrm{polylog}(n)$ overhead.) if Alice and Bob each have $2n$ bits of information, with mutual information n, in the sense of Kolmogorov complexity.

Distributed compression: a simple example

- Alice knows a line ℓ; Bob knows a point $P \in \ell$; They want to send ℓ and P to Zack.
- ℓ : $2n$ bits of information (intercept, slope in GF[2^n]).
- P : $2n$ bits of information (the 2 coord. in GF[2^n]).
- Total information in $(\ell, P) = 3n$ bits; mutual information of ℓ and $P = n$ bits.
- If Alice and Bob get together, they need to send $3n$ bits. What if they compress separately?



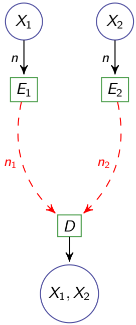
QUESTION 2:

Can Alice send $1.5n$ bits, and Bob $1.5n$ bits? Can Alice send $1.74n$ bits, and Bob $1.26n$ bits?

Ans: Yes and Yes (modulo a $\mathrm{polylog}(n)$ overhead.)

Distributed compression (IT view): Slepian-Wolf Theorem

- The classic Slepian-Wolf Th. is the analog of Shannon Source Coding Th. for the distributed compression of **memoryless** sources.
- Memoryless source: (X_1, X_2) consists of n independent draws from a joint distribution $p(b_1, b_2)$ on pair of bits.
- Encoding: $E_1 : \{0,1\}^n \to \{0,1\}^{n_1}$, $E_2 : \{0,1\}^n \to \{0,1\}^{n_2}$.
- Decoding: $D : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}^n \times \{0,1\}^n$.
- Goal: $D(E_1(X_1), E_2(X_2)) = (X_1, X_2)$ with probability $1 - \epsilon$.
- It is necessary that $n_1 + n_2 \geq H(X_1, X_2) - \epsilon n$, $n_1 \geq H(X_1 \mid X_2) - \epsilon n$, $n_2 \geq H(x_2 \mid x_1) - \epsilon n$.



Theorem (Slepian, Wolf, 1973)

There exist encoding/decoding functions E_1, E_2 and D satisfying the goal for all n_1, n_2 satisfying

$$n_1 + n_2 \geq H(X_1, X_2) + \epsilon n,\ n_1 \geq H(X_1 \mid X_2) + \epsilon n,\ n_2 \geq H(X_2 \mid X_1) + \epsilon n.$$

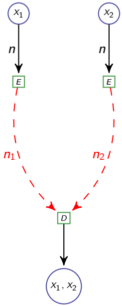
It holds for any constant number of sources.

Slepian-Wolf Th.: Some comments

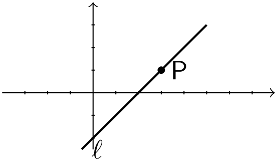
Theorem (Slepian, Wolf, 1973)

There exist encoding/decoding functions E_1, E_2 *and* D *such that* $n_1 + n_2 \geq H(X_1, X_2) + \epsilon n$, $n_1 \geq H(X_1 \mid X_2) + \epsilon n$, $n_2 \geq H(X_2 \mid X_1) + \epsilon n$.

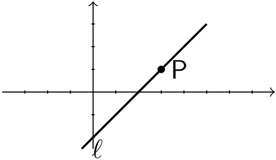
- Even if (X_1, X_2) are compressed together, the sender still needs to send $\approx H(X_1, X_2)$ many bits.
- **Strength of S.-W. Th. :** distributed compression = centralized compression, for memoryless sources.
- **Shortcoming of S.-W. Th. :** Memoryless sources are very simple. The theorem has been extended to stationary and ergodic sources (Cover, 1975), which are still pretty lame.



- Recall: Alice knows a line ℓ; Bob knows a point $P \in \ell$; They want to send ℓ and P to Zack.
- There is no generative model.
- Correlation can be described with the complexity profile: $C(\ell) = 2n, C(P) = 2n, C(\ell, P) = 3n$.
- Is it possible to have distributed compression based only on the complexity profile?
- If yes, what are the possible compression lengths?



- Recall: Alice knows a line ℓ; Bob knows a point $P \in \ell$; They want to send ℓ and P to Zack.
- There is no generative model.
- Correlation can be described with the complexity profile: $C(\ell) = 2n, C(P) = 2n, C(\ell, P) = 3n$.
- Is it possible to have distributed compression based only on the complexity profile?
- If yes, what are the possible compression lengths?



Necessary conditions: Suppose we want encoding/decoding procedures so that $D(E_1(x_1), E_2(x_2)) = (x_1, x_2)$ with probability $1 - \epsilon$, for all strings x_1, x_2. Then, for infinitely many x_1, x_2,

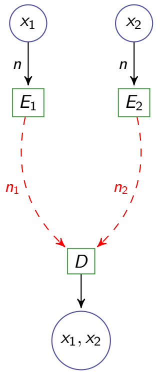
$$
\begin{aligned}
|E_1(x_1)| + |E_2(x_2)| &\geq C(x_1, x_2) + \log(1 - \epsilon) - O(1) \\
|E_1(x_1)| &\geq C(x_1 \mid x_2) + \log(1 - \epsilon) - O(1) \\
|E_2(x_2)| &\geq C(x_2 \mid x_1) + \log(1 - \epsilon) - O(1)
\end{aligned}
$$

Kolmogorov complexity version of the Slepian-Wolf Theorem

Theorem ((Z. 2017), (Bauwens, Z. 2019))

There exist probabilistic poly.-time algorithms E and algorithm D such that for all integers n_1, n_2 and n-bit strings x_1, x_2,
if $n_1 + n_2 \geq C(x_1, x_2)$, $n_1 \geq C(x_1 \mid x_2)$,
$n_2 \geq C(x_2 \mid x_1)$,
then

- *E on input (x_i, n_i) outputs a string p_i of length $n_i + O(\log^2 n)$, for $i = 1, 2$,*
- *D on input (p_1, p_2) outputs (x_1, x_2) with probability 0.99.*



There is an analogous version for any constant number of sources.

Proof sketch (1/2)

- Alice has x_1 and n_1.

Proof sketch (1/2)

- Alice has x_1 and n_1.
- Bob has x_2 and n_2.

Proof sketch (1/2)

- Alice has x_1 and n_1.
- Bob has x_2 and n_2.
- n_1, n_2 satisfy the Slepian-Wolf constraints:

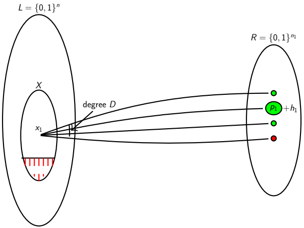
  $n_1+n_2 \geq C(x_1, x_2), n_1 \geq C(x_1 \mid x_2), n_2 \geq C(x_2 \geq x_1)$.

Proof sketch (1/2)

- Alice has x_1 and n_1.
- Bob has x_2 and n_2.
- n_1, n_2 satisfy the Slepian-Wolf constraints:

  $$n_1 + n_2 \geq C(x_1, x_2), n_1 \geq C(x_1 \mid x_2), n_2 \geq C(x_2 \geq x_1).$$

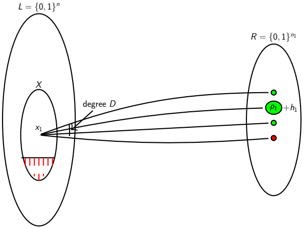
- Alice uses a conductor with output size $= n_1$.

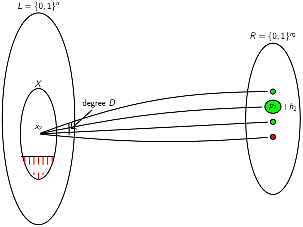


Proof sketch (1/2)

- Alice has x_1 and n_1.
- Bob has x_2 and n_2.
- n_1, n_2 satisfy the Slepian-Wolf constraints:

  $n_1 + n_2 \geq C(x_1, x_2), n_1 \geq C(x_1 \mid x_2), n_2 \geq C(x_2 \geq x_1)$.
- Alice uses a conductor with output size $= n_1$.
- Bob uses a conductor with output size $= n_2$.



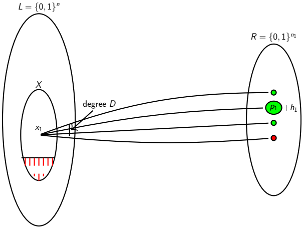


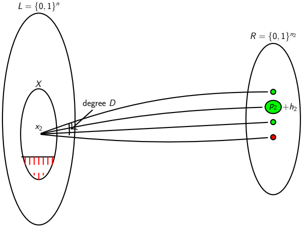
Proof sketch (1/2)

- Alice has x_1 and n_1.
- Bob has x_2 and n_2.
- n_1, n_2 satisfy the Slepian-Wolf constraints:

  $$n_1 + n_2 \geq C(x_1, x_2), n_1 \geq C(x_1 \mid x_2), n_2 \geq C(x_2 \geq x_1).$$

- Alice uses a conductor with output size $= n_1$.
- Bob uses a conductor with output size $= n_2$.
- Alice compresses x_1 by choosing a random neighbor p_1 + short hash-code h_1.



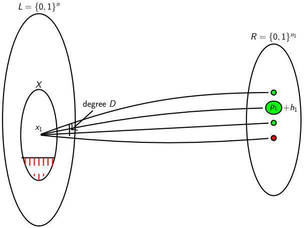


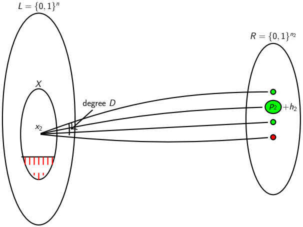
Proof sketch (1/2)

- Alice has x_1 and n_1.
- Bob has x_2 and n_2.
- n_1, n_2 satisfy the Slepian-Wolf constraints:

  $$n_1 + n_2 \geq C(x_1, x_2), n_1 \geq C(x_1 \mid x_2), n_2 \geq C(x_2 \geq x_1).$$

- Alice uses a conductor with output size $= n_1$.
- Bob uses a conductor with output size $= n_2$.
- Alice compresses x_1 by choosing a random neighbor p_1 + short hash-code h_1.
- Bob compresses x_2 by choosing a random neighbor p_2 + short hash-code h_2.



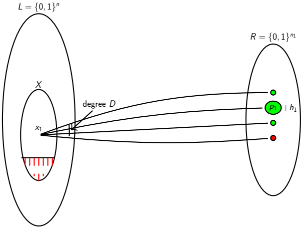


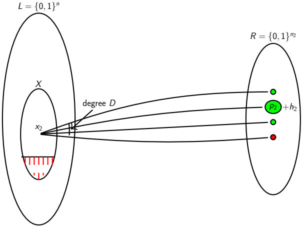
Proof-sketch (2/2)

- How to reconstruct (x_1, x_2) from (p_1, h_1) and (p_2, h_2)
- Enumerate the initial list of candidates: all pairs x_1', x_2' with

  $$n_1 + n_2 \geq C(x_1', x_2'), n_1 \geq C(x_1' \mid x_2'), n_2 \geq C(x_2' \geq x_1').$$

- Apply a cascade of two filters to each enumerated pair.
- Pair $(x_1', *)$ passes the first filter if (p_1, h_1) is the compressed code of x_1'.
- Pair $(*, x_2')$ passes the second filter if (p_2, h_2) is the compressed code of x_2'.
- With high probability, only (x_1, x_2) survive the two filters.





Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).

Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.

Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.

Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.
- At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.

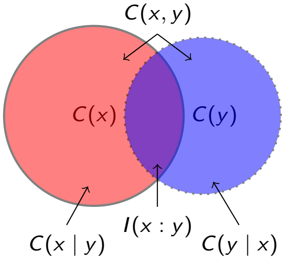
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.
- At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.
- The classical S.-W. Th. can be obtained from the Kolmogorov complexity version (because if X is memoryless, $H(X) - c_\epsilon \sqrt{n} \leq C(X) \leq H(X) + c_\epsilon \sqrt{n}$ with prob. $1 - \epsilon$).

Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.
- At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.
- The classical S.-W. Th. can be obtained from the Kolmogorov complexity version (because if X is memoryless, $H(X) - c_\epsilon\sqrt{n} \leq C(X) \leq H(X) + c_\epsilon\sqrt{n}$ with prob. $1-\epsilon$).
- The $O(\log^2 n)$ overhead can be reduced to $O(\log n)$, but compression is no longer in polynomial time.

Operational characterization of mutual information



$C(x)$ = length of a shortest description of x.
$C(x \mid y)$ = length of a shortest description of x given y.

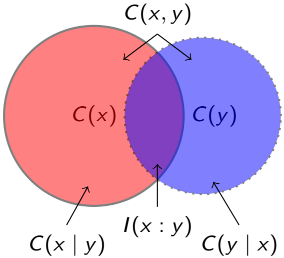
$\vdots$

Mutual information of x and y is defined by a formula:
$I(x:y) = C(x) + C(y) - C(x,y)$.
Also, $I(x:y) =^+ C(x) - C(x \mid y)$,
$I(x:y) =^+ C(y) - C(y \mid x)$

($=^+$ hides $\pm O(\log n)$)

All the regions except the center have an operational meaning.

Operational characterization of mutual information



$C(x)$ = length of a shortest description of x.
$C(x \mid y)$ = length of a shortest description of x given y.

$\vdots$

Mutual information of x and y is defined by a formula:
$I(x:y) = C(x) + C(y) - C(x,y)$.
Also, $I(x:y) =^{+} C(x) - C(x \mid y)$,
$I(x:y) =^{+} C(y) - C(y \mid x)$

($=^{+}$ hides $\pm O(\log n)$)

All the regions except the center have an operational meaning.

Does $I(x:y)$ have an operational meaning?

Mutual information and secret key agreement

- Question: Can mutual information be "materialized"?

Mutual information and secret key agreement

- Question: Can mutual information be “materialized”?
- Answer: YES.

Mutual information and secret key agreement

- Question: Can mutual information be "materialized"?
- Answer: YES.
- Mutual information of strings x, y = length of the longest shared secret key that Alice having x and Bob having y can establish via a randomized protocol.

Mutual information and secret key agreement

- Question: Can mutual information be "materialized"?
- Answer: YES.
- Mutual information of strings x, y = length of the longest shared secret key that Alice having x and Bob having y can establish via a randomized protocol.
- This was known in the setting of Information Theory (Shannon entropy, etc.) for memoryless and stationary ergodic sources.
- (Romashchenko, Z., 2018) Characterization holds in the framework of Kolmogorov complexity.

Secret key agreement protocol:

- Alice knows x
- Bob knows y
- they exchange messages and compute a shared secret key z
- z must be random conditioned by the transcript of the protocol

Secret key agreement protocol:

- Alice knows x
- Bob knows y
- they exchange messages and compute a shared secret key z
- z must be random conditioned by the transcript of the protocol

Our setting:

(1) Alice and Bob also know how their x and y are correlated.

Secret key agreement protocol:

- Alice knows x
- Bob knows y
- they exchange messages and compute a shared secret key z
- z must be random conditioned by the transcript of the protocol

Our setting:

(1) Alice and Bob also know how their x and y are correlated.
Technically, they know the complexity profile of x and y : $(C(x), C(y), C(x, y))$.

(2) Alice and Bob use randomized algorithms to compute their messages.

Secret key agreement protocol:

- Alice knows x
- Bob knows y
- they exchange messages and compute a shared secret key z
- z must be random conditioned by the transcript of the protocol

Our setting:

(1) Alice and Bob also know how their x and y are correlated.
Technically, they know the complexity profile of x and y : $(C(x), C(y), C(x,y))$.

(2) Alice and Bob use randomized algorithms to compute their messages.

Theorem (Characterization of the mutual information)

1. *There is a protocol that for every n-bit strings x and y allows to compute with high probability a shared secret key of length $I(x:y)$ (up to $-O(\log n)$).*
2. *No protocol can produce a longer shared secret key (up to $+O(\log n)$).*

Characterization of mutual information: the positive part

Theorem

There exists a secret key agreement protocol with the following property: if

- *Alice knows* x, ϵ, *and the complexity profile of* (x, y),
- *Bob knows* y, ϵ, *and the complexity profile of* (x, y),

then with probability $1 - \epsilon$ *they obtain a string* z *such that,*

$|z| \geq I(x : y) - O(\log(n/\epsilon))$

and $C(z \mid \mathrm{transcript}) \geq |z| - O(\log(1/\epsilon))$.

Characterization of mutual information: the positive part

Theorem

There exists a secret key agreement protocol with the following property: if

- *Alice knows* x, ϵ, *and the complexity profile of* (x, y),
- *Bob knows* y, ϵ, *and the complexity profile of* (x, y),

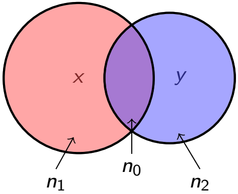
then with probability $1 - \epsilon$ *they obtain a string* z *such that,*

$|z| \geq I(x : y) - O(\log(n/\epsilon))$ /* common key of size $\geq^+ I(x : y)$ */

and $C(z \mid \text{transcript}) \geq |z| - O(\log(1/\epsilon))$. /* no information leakage */

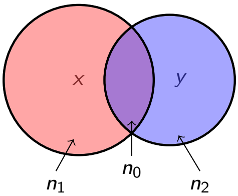
Secret key agreement: sketch of the general protocol

- Alice and Bob want to agree on a secret key.
- they can only communicate through a public channel.
- Alice knows x; Bob knows y;
- $C(x \mid y) =^{+} n_1$
- $C(y \mid x) =^{+} n_2$
- $I(x : y) =^{+} n_0$.



Secret key agreement: sketch of the general protocol

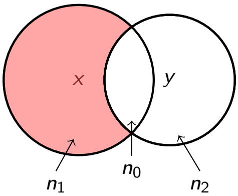
- Alice and Bob want to agree on a secret key.
- they can only communicate through a public channel.
- Alice knows x; Bob knows y;
- $C(x \mid y) =^{+} n_1$
- $C(y \mid x) =^{+} n_2$
- $I(x : y) =^{+} n_0$.



Protocol:

Secret key agreement: sketch of the general protocol

- Alice and Bob want to agree on a secret key.
- they can only communicate through a public channel.
- Alice knows x; Bob knows y;
- $C(x \mid y) =^+ n_1$
- $C(y \mid x) =^+ n_2$
- $I(x : y) =^+ n_0$.

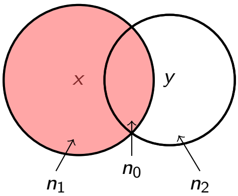


Protocol:

- Alice sends to Bob a program p of x given y of size $=^+ n_1$.

Secret key agreement: sketch of the general protocol

- Alice and Bob want to agree on a secret key.
- they can only communicate through a public channel.
- Alice knows x; Bob knows y;
- $C(x \mid y) =^{+} n_1$
- $C(y \mid x) =^{+} n_2$
- $I(x : y) =^{+} n_0$.

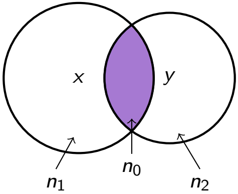


Protocol:

- Alice sends to Bob a program p of x given y of size $=^{+} n_1$.
- Bob (knowing y) reconstructs x.

Secret key agreement: sketch of the general protocol

- Alice and Bob want to agree on a secret key.
- they can only communicate through a public channel.
- Alice knows x; Bob knows y;
- $C(x \mid y) =^+ n_1$
- $C(y \mid x) =^+ n_2$
- $I(x : y) =^+ n_0$.

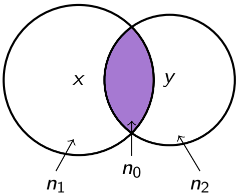


Protocol:

- Alice sends to Bob a program p of x given y of size $=^+ n_1$.
- Bob (knowing y) reconstructs x.
- Alice and Bob compute (independently) a program z of x given p of size $=^+ n_0$.

Secret key agreement: sketch of the general protocol

- Alice and Bob want to agree on a secret key.
- they can only communicate through a public channel.
- Alice knows x; Bob knows y;
- $C(x \mid y) =^{+} n_1$
- $C(y \mid x) =^{+} n_2$
- $I(x : y) =^{+} n_0$.



Protocol:

- Alice sends to Bob a program p of x given y of size $=^{+} n_1$.
- Bob (knowing y) reconstructs x.
- Alice and Bob compute (independently) a program z of x given p of size $=^{+} n_0$.
- Adversary gets p but learns nothing about z.

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x:y) + \delta(n) + O(\log(n/\epsilon))$.

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x:y) + \delta(n) + O(\log(n/\epsilon))$. /* common key of size $\leq^{+} I(x:y)$ */

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1 - \epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x : y) + \delta(n) + O(\log(n/\epsilon))$.

Under the hood:

Conditional information inequality

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x:y) + \delta(n) + O(\log(n/\epsilon))$.

Under the hood:

Conditional information inequality

- simple part: if no communication, then **key** $\leq I(x:y)$

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x : y) + \delta(n) + O(\log(n/\epsilon))$.

Under the hood:

Conditional information inequality

- simple part: if no communication, then **key** $\leq I(x : y)$
- still simple: with communication, **key** $\leq I(x : y \mid \text{transcript})$

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x : y) + \delta(n) + O(\log(n/\epsilon))$.

Under the hood:

Conditional information inequality

- simple part: if no communication, then **key** $\leq I(x : y)$
- still simple: with communication, **key** $\leq I(x : y \mid \text{transcript})$
- hard part: $I(x : y \mid \text{transcript}) \leq I(x : y)$

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have
$|z| \leq I(x:y) + \delta(n) + O(\log(n/\epsilon))$.

Under the hood:

Conditional information inequality

- simple part: if no communication, then **key** $\leq I(x:y)$
- still simple: with communication, **key** $\leq I(x:y \mid \mathrm{transcript})$
- hard part: $I(x:y \mid \mathrm{transcript}) \leq I(x:y)$
- technical lemma: $C(\mathrm{transcript} \mid x) + C(\mathrm{transcript} \mid y) \leq C(\mathrm{transcript})$

Characterization of mutual information: the negative part.

Theorem

Let x and y be input strings of length n on which the protocol succeeds with error probability ϵ so that with prob $1-\epsilon$ Alice and Bob have at the end the same z, and $C(z \mid t) \geq |z| - \delta(n)$.

Then with probability $\geq 1 - O(\epsilon)$ we have $|z| \leq I(x:y) + \delta(n) + O(\log(n/\epsilon))$.

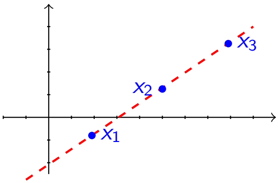
Under the hood:

Conditional information ineq

Kaced-Romashchenko-Vereshchagin 2017
(Shannon's entropy version)

- simple part: if no com
- still simple: with communication **key** $\leq I(x:y \mid \text{transcript})$
- hard part: $I(x:y \mid \text{transcript}) \leq I(x:y)$
- technical lemma: $C(\text{transcript} \mid x) + C(\text{transcript} \mid y) \leq C(\text{transcript})$

The puzzle

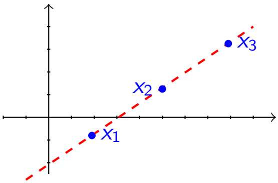


Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

The puzzle



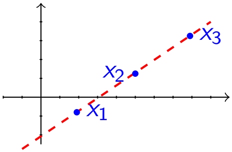
Kolia: x_1
Sasha: x_2
Andrei: x_3

points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

The puzzle: Solution



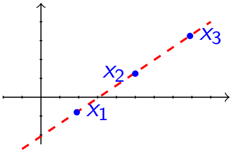
Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

Points x_1, x_2, x_3 belong to one line
in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

The puzzle: Solution



Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

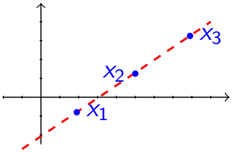
Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

- Kolia, Sasha, Andrei send, respectively, n_1, n_2, n_3 bits, so that at the end they all know all points.

The puzzle: Solution



Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

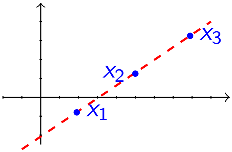
Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

- Kolia, Sasha, Andrei send, respectively, n_1, n_2, n_3 bits, so that at the end they all know all points.
- Information requirements: $n_i \geq n$, $n_i + n_j \geq 3n$, for all $i, j \in \{1, 2, 3\}$.

The puzzle: Solution



Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

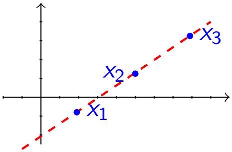
Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

- Kolia, Sasha, Andrei send, respectively, n_1, n_2, n_3 bits, so that at the end they all know all points.
- Information requirements: $n_i \geq n$, $n_i + n_j \geq 3n$, for all $i, j \in \{1, 2, 3\}$.
- $n_1, n_2, n_3 = 1.5n$ satisfy the requirements. By S.-W. Th., they can each send $1.5n$ bits.

The puzzle: Solution



Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

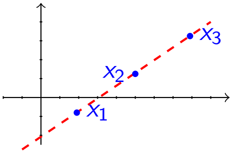
Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

- Kolia, Sasha, Andrei send, respectively, n_1, n_2, n_3 bits, so that at the end they all know all points.
- Information requirements: $n_i \geq n$, $n_i + n_j \geq 3n$, for all $i, j \in \{1, 2, 3\}$.
- $n_1, n_2, n_3 = 1.5n$ satisfy the requirements. By S.-W. Th., they can each send $1.5n$ bits.
- We have heard $4.5n$ bits, but they have $5n$ bits.

The puzzle: Solution



Kolia:	x_1
Sasha:	x_2
Andrei:	x_3

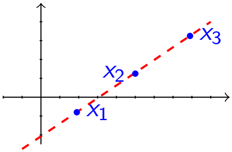
Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

- Kolia, Sasha, Andrei send, respectively, n_1, n_2, n_3 bits, so that at the end they all know all points.
- Information requirements: $n_i \geq n$, $n_i + n_j \geq 3n$, for all $i, j \in \{1, 2, 3\}$.
- $n_1, n_2, n_3 = 1.5n$ satisfy the requirements. By S.-W. Th., they can each send $1.5n$ bits.
- We have heard $4.5n$ bits, but they have $5n$ bits.
- They compress their $5n$ bits conditional to our $4.5n$ bits, and obtain $0.5n$ secret bits.

The puzzle: Solution



Kolia: x_1
Sasha: x_2
Andrei: x_3

Points x_1, x_2, x_3 belong to one line in the affine plane over $\mathbb{F}_{2^n}$

Each point has $2n$ points of information, but together they have $5n$ bits of information.

QUESTION: Can they agree on a secret key by discussing in this room, where we all hear what they say?

- Kolia, Sasha, Andrei send, respectively, n_1, n_2, n_3 bits, so that at the end they all know all points.
- Information requirements: $n_i \geq n$, $n_i + n_j \geq 3n$, for all $i, j \in \{1, 2, 3\}$.
- $n_1, n_2, n_3 = 1.5n$ satisfy the requirements. By S.-W. Th., they can each send $1.5n$ bits.
- We have heard $4.5n$ bits, but they have $5n$ bits.
- They compress their $5n$ bits conditional to our $4.5n$ bits, and obtain $0.5n$ secret bits.
- (Romashchenko, Z. 2018) This is the best they can do, they cannot obtain a longer secret key.

References

- M. Zimand, Kolmogorov complexity version of Slepian-Wolf coding, STOC 2017, available on arxiv `https://arxiv.org/abs/1511.03602`
- A. Romashchenko and M. Zimand, An operational characterization of mutual information in algorithmic information theory, ICALP 2018, available at ECCC `https://eccc.weizmann.ac.il/report/2018/043`
- B. Bauwens and M. Zimand, Almost minimum-description length compression and Slepian-Wolf coding in probabilistic polynomial time, 2019, in preparation.

Happy Birthday, Kolia and Sasha

https://www.youtube.com/watch?v=lKIGZsuHQ4U&t=201s