Computing: from classical to quantum

Bibhas Adhikari

Department of Mathematics
IIT Kharagpur

July 25, 2023

What is computing

What is computing

- Digital Representation of numbers/texts/audio/video... almost everything!!

What is computing

- Digital Representation of numbers/texts/audio/video... almost everything!!
- Physics of this representation!!

What is computing

- Digital Representation of numbers/texts/audio/video... almost everything!!
- Physics of this representation!!
- Devising a method of measurement!!

What is computing

- Digital Representation of numbers/texts/audio/video... almost everything!!
- Physics of this representation!!
- Devising a method of measurement!!
- Communication - reproducing at one point either exactly or approximately a message selected at another point (Shannon)

What is computing

- ▷ Digital Representation of numbers/texts/audio/video... almost everything!!
- ▷ Physics of this representation!!
- ▷ Devising a method of measurement!!
- ▷ Communication - reproducing at one point either exactly or approximately a message selected at another point (Shannon)

  Models of communication

What is computing

- Digital Representation of numbers/texts/audio/video... almost everything!!
- Physics of this representation!!
- Devising a method of measurement!!
- Communication - reproducing at one point either exactly or approximately a message selected at another point (Shannon)

  Models of communication

  Storage and Transmission

What is computing

- Digital Representation of numbers/texts/audio/video... almost everything!!
- Physics of this representation!!
- Devising a method of measurement!!
- Communication - reproducing at one point either exactly or approximately a message selected at another point (Shannon)

  Models of communication
  Storage and Transmission
  Message = information (??)

Models of computation

- Turing machine - it formalizes the intuitive notion of an algorithm

Models of computation

- Turing machine - it formalizes the intuitive notion of an algorithm

  Stimulated by a profound question (David Hilbert) - Whether an algorithm exists that has the potential to solve all mathematical problems, in theory!!

Models of computation

- ▷ Turing machine - it formalizes the intuitive notion of an algorithm

  Stimulated by a profound question (David Hilbert) - Whether an algorithm exists that has the potential to solve all mathematical problems, in theory!!

- ▷ Circuit model of computation - equivalent to the Turing machine and close to real computers

Models of computation

- Turing machine - it formalizes the intuitive notion of an algorithm

  Stimulated by a profound question (David Hilbert) - Whether an algorithm exists that has the potential to solve all mathematical problems, in theory!!

- Circuit model of computation - equivalent to the Turing machine and close to real computers

  information is carried by wires

Models of computation

- ▷ Turing machine - it formalizes the intuitive notion of an algorithm

  Stimulated by a profound question (David Hilbert) - Whether an algorithm exists that has the potential to solve all mathematical problems, in theory!!

- ▷ Circuit model of computation - equivalent to the Turing machine and close to real computers

  information is carried by wires

  a small set of elementary logical operations (gate) facilitates complex computation

Models of computation

- ▷ Turing machine - it formalizes the intuitive notion of an algorithm

  Stimulated by a profound question (David Hilbert) - Whether an algorithm exists that has the potential to solve all mathematical problems, in theory!!

- ▷ Circuit model of computation - equivalent to the Turing machine and close to real computers

  information is carried by wires

  a small set of elementary logical operations (gate) facilitates complex computation

  Resources: computer memory, time and energy

Turing machine

- Introduced by the mathematician Alan Turing in 1930

Turing machine

- ▷ Introduced by the mathematician Alan Turing in 1930

Main elements of a Turing machine M:

1. (scratch pad) k tapes: each is infinite and divided into cells, each cell holds one letter $a \in \Gamma = \{0, 1, \square, \triangleright\}$, called the alphabet of M. Each tape is quipped with a head that can read or write letters to the tape one cell at a time. The first tape is read-only, the input tape and the $k-1$ tapes are read-write, called the work tapes. The last one is the output tape, on which it writes the final answer

Turing machine

- ▷ Introduced by the mathematician Alan Turing in 1930

Main elements of a Turing machine M:

1. (scratch pad) k tapes: each is infinite and divided into cells, each cell holds one letter $a \in \Gamma = \{0, 1, \Box, \triangleright\}$, called the alphabet of M. Each tape is quipped with a head that can read or write letters to the tape one cell at a time. The first tape is read-only, the input tape and the $k-1$ tapes are read-write, called the work tapes. The last one is the output tape, on which it writes the final answer
2. A control unit/register: a finite number of possible states $Q = \{q_s, q_1, \ldots, q_l, q_h\}$, q_s and q_h are the start state and the halting state, respectively. The state determines its action at the next computational step:
   (i) real the letters
   (ii) for the $k-1$ read-write tapes, replace each letter with a new letter
   (iii) change its register to contain another state from Q
   (iv) move each head one cell to left or right or stay at the same place

Working of a Turing machine

Program: a finite set of instructions for each tape

1. the transition of the control unit from a state q_i to q_j

Working of a Turing machine

Program: a finite set of instructions for each tape

1. the transition of the control unit from a state q_i to q_j
2. the transition of the cell is addressed by the read/write head from a letter a_k to a letter a_l

Working of a Turing machine

Program: a finite set of instructions for each tape

1. the transition of the control unit from a state q_i to q_j
2. the transition of the cell is addressed by the read/write head from a letter a_k to a letter a_l
3. the displacement of the read/write head one cell left or right or stay

Three functions

$$q_j = f_q(q_i, a_k) \tag{1}$$
$$a_l = f_a(q_i, a_k) \tag{2}$$
$$d = f_d(q_i, a_k), \tag{3}$$

d denotes the displacement: left or right or stay

Working of a Turing machine

Program: a finite set of instructions for each tape

1. the transition of the control unit from a state q_i to q_j
2. the transition of the cell is addressed by the read/write head from a letter a_k to a letter a_l
3. the displacement of the read/write head one cell left or right or stay

Three functions

$$
\begin{aligned}
q_j &= f_q(q_i, a_k) && (1)\\
a_l &= f_a(q_i, a_k) && (2)\\
d &= f_d(q_i, a_k), && (3)
\end{aligned}
$$

d denotes the displacement: left or right or stay

Transition function $\delta : Q \times \Gamma^k \to Q \times \Gamma^{k-1} \times \{L, S, R\}^k, \; k \geq 2$

Working of a Turing machine

Program: a finite set of instructions for each tape

1. the transition of the control unit from a state q_i to q_j
2. the transition of the cell is addressed by the read/write head from a letter a_k to a letter a_l
3. the displacement of the read/write head one cell left or right or stay

Three functions

$$\begin{aligned} q_j &= f_q(q_i, a_k) && (1) \\ a_l &= f_a(q_i, a_k) && (2) \\ d &= f_d(q_i, a_k), && (3) \end{aligned}$$

d denotes the displacement: left or right or stay

Transition function $\delta : \ Q \times \Gamma^k \to Q \times \Gamma^{k-1} \times \{L, S, R\}^k, \ k \geq 2$

Question Does a TM halt at every input in a finite number of steps?

Turing machine

Thus the working of a Turing machine at each tape is described by

$$(q_i, a_k) \mapsto (q_j, a_l, d)$$

Question Does it have any resemblance in mordern day computers?

[1] Arora, S. and Barak, B., 2009. Computational complexity: a modern approach. Cambridge University Press.

Turing machine

Thus the working of a Turing machine at each tape is described by

$$(q_i, a_k) \mapsto (q_j, a_l, d)$$

Question Does it have any resemblance in mordern day computers?
Question Which computational tasks/functions are computable?

[1] Arora, S. and Barak, B., 2009. Computational complexity: a modern approach. Cambridge University Press.

Turing machine

Thus the working of a Turing machine at each tape is described by

$$(q_i, a_k) \mapsto (q_j, a_l, d)$$

Question Does it have any resemblance in mordern day computers?
Question Which computational tasks/functions are computable?

Computing a function and running time[1] Let $f : \{0,1\}^* \to \{0,1\}^*$ and let $T : \mathbb{N} \to \mathbb{N}$ be some function, and let M be a Turing machine. We say that M computes f if for every $x \in \{0,1\}^*$, whenever M is initialized to the start configuration on input x, then it halts with $f(x)$ written on its output tape. We say M computes f in $T(n)$-time if its computation on every input x requires at most $T(|x|)$ steps.

[1] Arora, S. and Barak, B., 2009. Computational complexity: a modern approach. Cambridge University Press.

Turing machine

The Church-Turing thesis: (which problems TMs are capable of solving?)

Turing machine

The Church-Turing thesis: (which problems TMs are capable of solving?)The class of all functions computable by a Turing machine is equivalent to the class of functions comutable by means of an algorithm.

Turing machine

The Church-Turing thesis: (which problems TMs are capable of solving?)The class of all functions computable by a Turing machine is equivalent to the class of functions comutable by means of an algorithm.

Note The thesis is formulated in 1936 and has never been disproved as we we are not aware of any algorithm that computes a function not computable by a TM.

The universal Turing machine

Turing machine

The Church-Turing thesis: (which problems TMs are capable of solving?)The class of all functions computable by a Turing machine is equivalent to the class of functions comutable by means of an algorithm.

Note The thesis is formulated in 1936 and has never been disproved as we we are not aware of any algorithm that computes a function not computable by a TM.

- The universal Turing machine
- The probabilistic Turing machine

Turing machine

The Church-Turing thesis: (which problems TMs are capable of solving?)The class of all functions computable by a Turing machine is equivalent to the class of functions comutable by means of an algorithm.

Note The thesis is formulated in 1936 and has never been disproved as we we are not aware of any algorithm that computes a function not computable by a TM.

- The universal Turing machine
- The probabilistic Turing machine
- The halting problem (undecidable!!)

Circuit model of computation

- Bit - the elementary unit of classical information

Circuit model of computation

- Bit - the elementary unit of classical information
- Bit is a binary variable, takes the values 0 and 1

Circuit model of computation

- Bit - the elementary unit of classical information
- Bit is a binary variable, takes the values 0 and 1
- Circuit - made of wires and gates, each wire carries one bit of information, and gates perform logic operations on these bits

Circuit model of computation

- Bit - the elementary unit of classical information
- Bit is a binary variable, takes the values 0 and 1
- Circuit - made of wires and gates, each wire carries one bit of information, and gates perform logic operations on these bits
- Classical computer - a digital device, the input and output are sequences of $0'$s and $1'$s

Circuit model of computation

- Bit - the elementary unit of classical information
- Bit is a binary variable, takes the values 0 and 1
- Circuit - made of wires and gates, each wire carries one bit of information, and gates perform logic operations on these bits
- Classical computer - a digital device, the input and output are sequences of $0'$s and $1'$s

For instance: a positive integer $N < 2^n$ can be written as

$$N = \sum_{k=0}^{n-1} a_k \, 2^k$$

and hence equivalently

$$N = a_{n-1} \dots a_1 a_0$$

Circuit model of computation

- Bit - the elementary unit of classical information
- Bit is a binary variable, takes the values 0 and 1
- Circuit - made of wires and gates, each wire carries one bit of information, and gates perform logic operations on these bits
- Classical computer - a digital device, the input and output are sequences of $0'$s and $1'$s

For instance: a positive integer $N < 2^n$ can be written as

$$N = \sum_{k=0}^{n-1} a_k \, 2^k$$

and hence equivalently

$$N = a_{n-1} \ldots a_1 a_0$$

The binary codes for non-integer numbers:

$$5.5 = 101.1,\ 5.25 = 101.01,\ 5.125 = 101.001$$

Circuit model of computation

- The advantage of binary numbers is that they can be stored in electrical devices with two possible values - such as high and low voltages or switches with only two positions on and off can be used to load one bit of information

Elementary logic gates Logical function with n-bit input and m-bit output:

$$f : \{0,1\}^n \to \{0,1\}^m$$

Universal gates: Any function $f : \{0,1\}^n \to \{0,1\}^m$ can be constructuted from the elementary gates AND, OR, NOT, and COPY. Thus these gates constitute a universal model of computation.

Computational complexity

Resources to execute an algorithm in a computer: space, time and energy

Computational complexity

Resources to execute an algorithm in a computer: space, time and energy

Computational complexity- find the minimum resources to solve a problem with the best possible algorithm

Notation Given two functions $f(n)$ and $g(n)$, we write $f = O(g)$ if

$$c_1 \leq |f(n)/g(n)| \leq c_2,$$

with $0 \leq c_1 \leq c_2 < \infty$.

Computational complexity

Resources to execute an algorithm in a computer: space, time and energy

Computational complexity- find the minimum resources to solve a problem with the best possible algorithm

Notation Given two functions $f(n)$ and $g(n)$, we write $f = O(g)$ if

$$c_1 \leq |f(n)/g(n)| \leq c_2,$$

with $0 \leq c_1 \leq c_2 < \infty$.

Question What is the complexity of multiplying two n-digit numbers on a Turing machine?

Computational complexity

Resources to execute an algorithm in a computer: space, time and energy

Computational complexity- find the minimum resources to solve a problem with the best possible algorithm

Notation Given two functions $f(n)$ and $g(n)$, we write $f = O(g)$ if

$$c_1 \leq |f(n)/g(n)| \leq c_2,$$

with $0 \leq c_1 \leq c_2 < \infty$.
Question What is the complexity of multiplying two n-digit numbers on a Turing machine?
An answer In 1971 Schonhage and Strassen discovered an algorithm that requires $O(n \log n \log \log n)$

Computational complexity

Assuming n as the input size, the number of bits required to specify the input, the solvable problems into two classes:

Computational complexity

Assuming n as the input size, the number of bits required to specify the input, the solvable problems into two classes:

- ▷ Efficient/tractable/feasible: problems that can be solved using resources that are bounded by a polynomial in n - called the polynomial class

Computational complexity

Assuming n as the input size, the number of bits required to specify the input, the solvable problems into two classes:

- Efficient/tractable/feasible: problems that can be solved using resources that are bounded by a polynomial in n - called the polynomial class
- Difficult/intractable/unfeasible: problems that are superpolynomial i.e. it grows faster than any polynomial in n

Example

1. The best known algorithm for the factorization of an integer N requires $\exp(O(n^{1/3}(\log n)^{2/3}))$ operations, where $n = \log N$.

Computational complexity

Assuming n as the input size, the number of bits required to specify the input, the solvable problems into two classes:

- Efficient/tractable/feasible: problems that can be solved using resources that are bounded by a polynomial in n - called the polynomial class
- Difficult/intractable/unfeasible: problems that are superpolynomial i.e. it grows faster than any polynomial in n

Example

1. The best known algorithm for the factorization of an integer N requires $\exp(O(n^{1/3}(\log n)^{2/3}))$ operations, where $n = \log N$. Thus the factorization of a number 250 digits long would take 10 million years on a 200-MIPS computer
2. However, a polynomial algorithm scaling as n^α, $\alpha \gg 1$, like $\alpha = 10^3$ can hardly be regarded be easy

Computational complexity

Does the complexity depend on the model?

Computational complexity

Does the complexity depend on the model?

The strong Church-Turing thesis A probabilistic Turing machine can simulate any model of computation with at most a polynomial increase in the number of elementary operations required.

Computational complexity

Does the complexity depend on the model?

The strong Church-Turing thesis A probabilistic Turing machine can simulate any model of computation with at most a polynomial increase in the number of elementary operations required.

Question What does this mean?

Computational complexity

Does the complexity depend on the model?

The strong Church-Turing thesis A probabilistic Turing machine can simulate any model of computation with at most a polynomial increase in the number of elementary operations required.

Question What does this mean?

Observation Shor's quantum algorithm with polynomial resource can solve the factorization problem, however if such a classical algorithm does not exist then only we will be able to say that quantum model of computation is powerful than classical!!

Computational complexity

Limits of computation In a logical system defined by a set of axioms and rules, there is a fundamental question regarding whether all conceivable propositions can, in principle, be proven as either true or false.

Computational complexity

Limits of computation In a logical system defined by a set of axioms and rules, there is a fundamental question regarding whether all conceivable propositions can, in principle, be proven as either true or false.

In 1930, Kurt Godel proved a theorem that there always exists a proposition in any logical system that is undecidable i.e. it can neither be proved nor disproved using the axioms and rules in the logical system

Computational complexity

Limits of computation In a logical system defined by a set of axioms and rules, there is a fundamental question regarding whether all conceivable propositions can, in principle, be proven as either true or false.

In 1930, Kurt Godel proved a theorem that there always exists a proposition in any logical system that is undecidable i.e. it can neither be proved nor disproved using the axioms and rules in the logical system - thus any logical system is incomplete and this is the limit of computation, not ALL arithmetical questions cannot be answered !!

Computational complexity

Limits of computation In a logical system defined by a set of axioms and rules, there is a fundamental question regarding whether all conceivable propositions can, in principle, be proven as either true or false.

In 1930, Kurt Godel proved a theorem that there always exists a proposition in any logical system that is undecidable i.e. it can neither be proved nor disproved using the axioms and rules in the logical system - thus any logical system is incomplete and this is the limit of computation, not ALL arithmetical questions cannot be answered !!

Complexity class - is a set of (Boolean) functions that can be computed within given resource bounds.

Computational complexity

Limits of computation In a logical system defined by a set of axioms and rules, there is a fundamental question regarding whether all conceivable propositions can, in principle, be proven as either true or false.

In 1930, Kurt Godel proved a theorem that there always exists a proposition in any logical system that is undecidable i.e. it can neither be proved nor disproved using the axioms and rules in the logical system - thus any logical system is incomplete and this is the limit of computation, not ALL arithmetical questions cannot be answered !!

Complexity class - is a set of (Boolean) functions that can be computed within given resource bounds.

Language - $L \subseteq \{0,1\}^*$ and a machine decides a language L if it computes the function $f : \{0,1\}^* \to \{0,1\}$, where $f_L(x) = 1$ if and only if $x \in L$

Computational complexity

Limits of computation In a logical system defined by a set of axioms and rules, there is a fundamental question regarding whether all conceivable propositions can, in principle, be proven as either true or false.

In 1930, Kurt Godel proved a theorem that there always exists a proposition in any logical system that is undecidable i.e. it can neither be proved nor disproved using the axioms and rules in the logical system - thus any logical system is incomplete and this is the limit of computation, not ALL arithmetical questions cannot be answered !!

Complexity class - is a set of (Boolean) functions that can be computed within given resource bounds.

Language - $L \subseteq \{0,1\}^*$ and a machine decides a language L if it computes the function $f : \{0,1\}^* \to \{0,1\}$, where $f_L(x) = 1$ if and only if $x \in L$
Question Boolean functions and languages are equivalent!!

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)
  Question Does the "integer multiplication" belong to **P**?

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)
  Question Does the "integer multiplication" belong to **P**?
  Note The class **P** contains only the decision problems!!
  Question What is the decision version of the "integer multiplication"?
- **NP** - class of problems whose solution can be verified in polynomial time

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)
  Question Does the "integer multiplication" belong to **P**?
  Note The class **P** contains only the decision problems!!
  Question What is the decision version of the "integer multiplication"?
- **NP** - class of problems whose solution can be verified in polynomial time
  Example Finding maximum independent set in a given graph

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)
  Question Does the "integer multiplication" belong to **P**?
  Note The class **P** contains only the decision problems!!
  Question What is the decision version of the "integer multiplication"?
- **NP** - class of problems whose solution can be verified in polynomial time
  Example Finding maximum independent set in a given graph
- **NPC** - a problem in **NP** is called **NP**-complete if any problem in **NP** is polynomially reducible to it

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)
  Question Does the "integer multiplication" belong to **P**?
  Note The class **P** contains only the decision problems!!
  Question What is the decision version of the "integer multiplication"?
- **NP** - class of problems whose solution can be verified in polynomial time
  Example Finding maximum independent set in a given graph
- **NPC** - a problem in **NP** is called **NP**-complete if any problem in **NP** is polynomially reducible to it

  Example Travelling salesman problem

Complexity classes

- **P** - a problem in this class can be solved in polynomial time i.e. in a polynomial of input size number of steps
  Example Graph connectivity problem (depth-first-search)
  Question Does the "integer multiplication" belong to **P**?
  Note The class **P** contains only the decision problems!!
  Question What is the decision version of the "integer multiplication"?
- **NP** - class of problems whose solution can be verified in polynomial time
  Example Finding maximum independent set in a given graph
- **NPC** - a problem in **NP** is called **NP**-complete if any problem in **NP** is polynomially reducible to it
  Example Travelling salesman problem

Question Under what condition $\mathbf{P} = \mathbf{NP}$ or $\mathbf{P} \neq \mathbf{NP}$
Note The factorization problem and graph isomorphism problem are not known to be in **P** nor **NPC**

Computational complexity

Reduction, **NP**-hardness and **NP**-completeness A language L is polynomial-time reducible to a language L', denoted as $L \leq_p L'$, if there is a polynomial-time computable function such that for every input x, $x \in L$ if and only if $f(x) \in L'$. Then we say

L' is **NP**-hard if $L \leq_p L'$ for every $L \in$ **NP.**

L' is **NP** complete if L' is **NP**-hard and $L' \in$ **NP**

Computational complexity

Reduction, **NP**-hardness and **NP**-completeness A language L is polynomial-time reducible to a language L', denoted as $L \leq_p L'$, if there is a polynomial-time computable function such that for every input x, $x \in L$ if and only if $f(x) \in L'$. Then we say

L' is **NP**-hard if $L \leq_p L'$ for every $L \in$ **NP.**

L' is **NP** complete if L' is **NP**-hard and $L' \in$ **NP**

Question Can you explain **NP**-hard languages in one line?

Computational complexity

Reduction, **NP**-hardness and **NP**-completeness A language L is polynomial-time reducible to a language L', denoted as $L \leq_p L'$, if there is a polynomial-time computable function such that for every input x, $x \in L$ if and only if $f(x) \in L'$. Then we say

> L' is **NP**-hard if $L \leq_p L'$ for every $L \in$ **NP.**
>
> L' is **NP** complete if L' is **NP**-hard and $L' \in$ **NP**

Question Can you explain **NP**-hard languages in one line?

Question Why is the notion of **NPC** significant?

Space complexity

- **PSPACE** - class of problems which can be solved by means of space resources that are polynomial in the input size, independently of the computation time

  Conjecture $\mathbf{P} \neq \mathbf{PSPACE}$

Space complexity

- **PSPACE** - class of problems which can be solved by means of space resources that are polynomial in the input size, independently of the computation time

  Conjecture $\mathbf{P} \neq \mathbf{PSPACE}$

  Question $\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}$

Space complexity

- **PSPACE** - class of problems which can be solved by means of space resources that are polynomial in the input size, independently of the computation time

  Conjecture $\mathbf{P} \neq \mathbf{PSPACE}$

  Question $\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}$

- **BPP** - a decision problem is in in this class if there exists a polynomial-time algorithm (in a probabilistic Turing machine) such that the probability of getting the right answer is larger than $\frac{1}{2} + \delta$ for every possible input and $\delta > 0$
- **BQP** - a decision problem is in this class if there is a polynomial-time quantum algorithm that gives the right answer with probability larger than $\frac{1}{2} + \delta$, $\delta > 0$.
  Example Shor's algorithm belongs to this class with $O(n^2 \log n \log \log n \log(1/\epsilon))$, ϵ is the probability of error.

Complexity classes

Question $\mathbf{P} \subseteq \mathbf{BPP} \subseteq \mathbf{BQP} \subseteq \mathbf{PSPACE}$

Complexity classes

Question $\mathbf{P} \subseteq \mathbf{BPP} \subseteq \mathbf{BQP} \subseteq \mathbf{PSPACE}$

Question Can we say that a quantum computer would be better than a classical computer?

Boolean circuits

For every $n \in \mathbb{N}$, an n-input, single-output Boolean circuit is a directed acyclic graph with m resources (vertices with no incoming edges) and one sink (vertex with no outgoing edges).

Boolean circuits

For every $n \in \mathbb{N}$, an n-input, single-output Boolean circuit is a directed acyclic graph with m resources (vertices with no incoming edges) and one sink (vertex with no outgoing edges).

- All nonsource vertices are called gates labelled with one of $\wedge, \vee, \neg$
- The vertices labelled with $\wedge, \vee$ have number of incoming edges equal to 2
- The vertices labelled with $\neg$ have one incoming edge
- The size of a circuit is the number of vertices in it

Boolean circuits

For every $n \in \mathbb{N}$, an n-input, single-output Boolean circuit is a directed acyclic graph with m resources (vertices with no incoming edges) and one sink (vertex with no outgoing edges).

- All nonsource vertices are called gates labelled with one of $\wedge, \vee, \neg$
- The vertices labelled with $\wedge, \vee$ have number of incoming edges equal to 2
- The vertices labelled with $\neg$ have one incoming edge
- The size of a circuit is the number of vertices in it

Note $x\, XORy = (x \wedge (\neg y)) \vee ((\neg x) \wedge y)$

Boolean circuits

For every $n \in \mathbb{N}$, an n-input, single-output Boolean circuit is a directed acyclic graph with m resources (vertices with no incoming edges) and one sink (vertex with no outgoing edges).

- All nonsource vertices are called gates labelled with one of $\wedge, \vee, \neg$
- The vertices labelled with $\wedge, \vee$ have number of incoming edges equal to 2
- The vertices labelled with $\neg$ have one incoming edge
- The size of a circuit is the number of vertices in it

Note $x\, XOR y = (x \wedge (\neg y)) \vee ((\neg x) \wedge y)$

Note The circuits in silicon chips used in modern computers are not acyclic and use cycles to implement memory. However, any computation that runs on a silicon chip with g gates and finishes in time t, can also be performed by a Boolean circuit of size $O(gt)$

Boolean circuits

For every $n \in \mathbb{N}$, an n-input, single-output Boolean circuit is a directed acyclic graph with m resources (vertices with no incoming edges) and one sink (vertex with no outgoing edges).

- All nonsource vertices are called gates labelled with one of $\wedge, \vee, \neg$
- The vertices labelled with $\wedge, \vee$ have number of incoming edges equal to 2
- The vertices labelled with $\neg$ have one incoming edge
- The size of a circuit is the number of vertices in it

Note $x\, XORy = (x \wedge (\neg y)) \vee ((\neg x) \wedge y)$

Note The circuits in silicon chips used in modern computers are not acyclic and use cycles to implement memory. However, any computation that runs on a silicon chip with g gates and finishes in time t, can also be performed by a Boolean circuit of size $O(gt)$

Homework Uniform vs non-uniform models

Boolean circuits

A $T(n)$-size circuit family is a sequence $\{C_n\}_{n\in\mathbb{N}}$ of Boolean circuits, where C_n has n inputs and single output, and its size $\leq T(n)$ for every n.

Boolean circuits

A $T(n)$-size circuit family is a sequence $\{C_n\}_{n\in\mathbb{N}}$ of Boolean circuits, where C_n has n inputs and single output, and its size $\leq T(n)$ for every n.

A language L is in $\mathbf{SIZE}(T(n))$ if there exists a $T(n)$-size circuit family $\{C_n\}$ such that for every $x \in \{0,1\}^n$, $x \in L$ if and only if $C_n(x) = 1$

Boolean circuits

A $T(n)$-size circuit family is a sequence $\{C_n\}_{n\in\mathbb{N}}$ of Boolean circuits, where C_n has n inputs and single output, and its size $\leq T(n)$ for every n.

A language L is in $\mathbf{SIZE}(T(n))$ if there exists a $T(n)$-size circuit family $\{C_n\}$ such that for every $x \in \{0,1\}^n$, $x \in L$ if and only if $C_n(x) = 1$

$\mathbf{P}_{/poly}$ - the class of languages decidable by polynomial-sized circuit families, i.e. $\mathbf{P}_{/}poly = \cup_c \mathbf{SIZE}(n^c)$

Results

1. Every function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a Boolean circuit of size $O(2^n/n)$

Boolean circuits

A $T(n)$-size circuit family is a sequence $\{C_n\}_{n\in\mathbb{N}}$ of Boolean circuits, where C_n has n inputs and single output, and its size $\leq T(n)$ for every n.

A language L is in $\mathbf{SIZE}(T(n))$ if there exists a $T(n)$-size circuit family $\{C_n\}$ such that for every $x \in \{0,1\}^n$, $x \in L$ if and only if $C_n(x) = 1$

$\mathbf{P}_{/poly}$ - the class of languages decidable by polynomial-sized circuit families, i.e. $\mathbf{P}_{/}poly = \cup_c \mathbf{SIZE}(n^c)$

Results

1. Every function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a Boolean circuit of size $O(2^n/n)$
2. $\mathbf{P} \subseteq \mathbf{P}_{/poly}$
3. (Hard functions) For every $n > 1$, there exists a function $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by a circuit of size $2^n/(10n)$ (Shannon)

Boolean circuits

A $T(n)$-size circuit family is a sequence $\{C_n\}_{n\in\mathbb{N}}$ of Boolean circuits, where C_n has n inputs and single output, and its size $\leq T(n)$ for every n.

A language L is in $\mathbf{SIZE}(T(n))$ if there exists a $T(n)$-size circuit family $\{C_n\}$ such that for every $x \in \{0,1\}^n$, $x \in L$ if and only if $C_n(x) = 1$

$\mathbf{P}_{/poly}$ - the class of languages decidable by polynomial-sized circuit families, i.e. $\mathbf{P}_{/}poly = \cup_c \mathbf{SIZE}(n^c)$

Results

1. Every function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a Boolean circuit of size $O(2^n/n)$
2. $\mathbf{P} \subseteq \mathbf{P}_{/poly}$
3. (Hard functions) For every $n > 1$, there exists a function $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by a circuit of size $2^n/(10n)$ (Shannon)

Homework Depth complexity of a Boolean function

Quantum computation

Question TM and Circuit models are equivalent!!

[2] Bernstein, E. and Vazirani, U., 1993, June. Quantum complexity theory. In Proceedings of the twenty-fifth annual ACM symposium on Theory of computing (pp. 11-20).

Quantum computation

Question TM and Circuit models are equivalent!!

Recall The Shor's polynomial-time quantum algorithm for factorizing integers pose a serious challenge to the strong Church-Turing thesis since no polynomial time algorithm is known for deterministic or probabilistic Turing machines. Thus if quantum computers are physically realizable then the strong Church-Turing thesis is wrong.

[2] Bernstein, E. and Vazirani, U., 1993, June. Quantum complexity theory. In Proceedings of the twenty-fifth annual ACM symposium on Theory of computing (pp. 11-20).

Quantum computation

Question TM and Circuit models are equivalent!!

Recall The Shor's polynomial-time quantum algorithm for factorizing integers pose a serious challenge to the strong Church-Turing thesis since no polynomial time algorithm is known for deterministic or probabilistic Turing machines. Thus if quantum computers are physically realizable then the strong Church-Turing thesis is wrong.

Note[2] TM fails to capture all physically realizable computing devices for a fundamental reason: the TM is based on a classical physics model of the universe, whereas current physical theory asserts that the universe is quantum physical.

[2] Bernstein, E. and Vazirani, U., 1993, June. Quantum complexity theory. In Proceedings of the twenty-fifth annual ACM symposium on Theory of computing (pp. 11-20).

Quantum computation models

- Configuration of a TM - complete description of the contents of the tape, the location of the tape head, and the state $q \in Q$ of the control

[3] Deutsch, D., 1985. Quantum theory, the Church–Turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 400(1818), pp.97-117.

Quantum computation models

- ▷ Configuration of a TM - complete description of the contents of the tape, the location of the tape head, and the state $q \in Q$ of the control
- ▷ At any time only a finite number of tape cells may contain nonblank symbols

[3] Deutsch, D., 1985. Quantum theory, the Church–Turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 400(1818), pp.97-117.

Quantum computation models

- Configuration of a TM - complete description of the contents of the tape, the location of the tape head, and the state $q \in Q$ of the control
- At any time only a finite number of tape cells may contain nonblank symbols
- Probabilistic TM - can be described as infinite dimensional stochastic matrix with rows and columns are indexed by configurations

[3] Deutsch, D., 1985. Quantum theory, the Church–Turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 400(1818), pp.97-117.

Quantum computation models

- Configuration of a TM - complete description of the contents of the tape, the location of the tape head, and the state $q \in Q$ of the control
- At any time only a finite number of tape cells may contain nonblank symbols
- Probabilistic TM - can be described as infinite dimensional stochastic matrix with rows and columns are indexed by configurations
- Consequently, if a probability distribution is represented as $|v\rangle$ then the distribution at the next step is $M|v\rangle$
- M is refereed to as 1time evolution operator'

Quantum Turing machine[3]

[3] Deutsch, D., 1985. Quantum theory, the Church–Turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 400(1818), pp.97-117.

Quantum Turing machine (QTM)

Let $\widetilde{\mathbb{C}}$ denote the set of complex numbers α such that there is a deterministic algorithm that computes the real and imaginary parts of α to within 2^{-n} in time polynomial in n.

[4]Vazirani, U., 2002. A survey of quantum complexity theory. In Proceedings of Symposia in Applied Mathematics (Vol. 58, pp. 193-220).

Quantum Turing machine (QTM)

Let $\widetilde{\mathbb{C}}$ denote the set of complex numbers α such that there is a deterministic algorithm that computes the real and imaginary parts of α to within 2^{-n} in time polynomial in n.

Then a QTM[4] (single tape) is a triplet (Γ, Q, d) with the quantum transition function

$$\delta : Q \times \Gamma \to \widetilde{\mathbb{C}}^{\Gamma \times Q \times \{L,R\}}$$

[4]Vazirani, U., 2002. A survey of quantum complexity theory. In Proceedings of Symposia in Applied Mathematics (Vol. 58, pp. 193-220).

Quantum Turing machine (QTM)

Let $\widetilde{\mathbb{C}}$ denote the set of complex numbers α such that there is a deterministic algorithm that computes the real and imaginary parts of α to within 2^{-n} in time polynomial in n.

Then a QTM[4] (single tape) is a triplet (Γ, Q, d) with the quantum transition function

$$\delta : \ Q \times \Gamma \to \widetilde{\mathbb{C}}^{\Gamma \times Q \times \{L,R\}}$$

Let S be the inner product space of finite linear combinations of configurations with the Euclidean norm. Then QTM M defines a linear operator $U_M : \ S \to S$: if M starts in configuration c with current state p and scanned symbol σ, then after one step M will be in superposition of configurations $\psi = \sum_i \alpha_i \, c_i$, where α_i corresponds to the transition $\delta(p, ...)$, and c_i is the new configuration that results from applying this transition c. Extending this map to the entire space S through linearity gives the liner time evolution operator U_M

[4]Vazirani, U., 2002. A survey of quantum complexity theory. In Proceedings of Symposia in Applied Mathematics (Vol. 58, pp. 193-220).

Quantum circuit model

Quantum computation process:

- prepare - initial state $|\psi_i\rangle$

Quantum circuit model

Quantum computation process:

- prepare - initial state $|\psi_i\rangle$
- manipulate - unitary transformation

Quantum circuit model

Quantum computation process:

- prepare - initial state $|\psi_i\rangle$
- manipulate - unitary transformation
- measurement - wrt a basis or observable
- A quantum circuit on n qubits implements a unitary transformation on the Hilbert space $(\mathbb{C}^2)^{\otimes n}$
- Some elementary quantum gates:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, R(\delta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\delta} \end{bmatrix}$$

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Quantum circuit model

Quantum computation process:

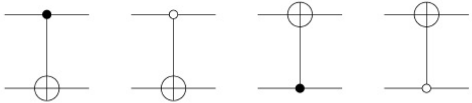
- prepare - initial state $|\psi_i\rangle$
- manipulate - unitary transformation
- measurement - wrt a basis or observable
- A quantum circuit on n qubits implements a unitary transformation on the Hilbert space $(\mathbb{C}^2)^{\otimes n}$
- Some elementary quantum gates:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \; H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, R(\delta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\delta} \end{bmatrix}$$

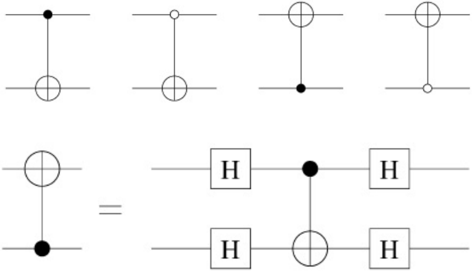
$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Note $CNOT(\alpha|0\rangle + \beta|1\rangle)|0\rangle = \alpha|00\rangle + \beta|11\rangle$, which is not separable when $\alpha, \beta \neq 0$.

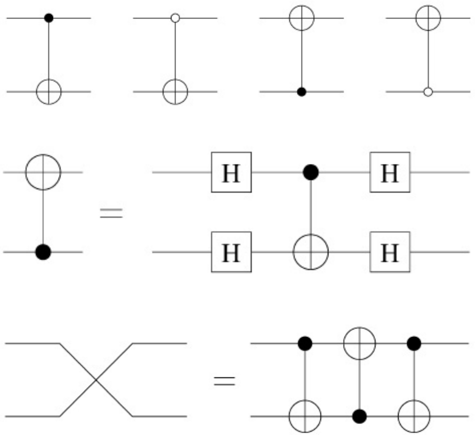
Quantum gates



Quantum gates

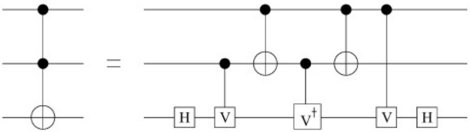


Quantum gates



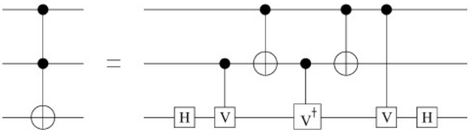
Quantum gates

Toffolli gate: $V = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$

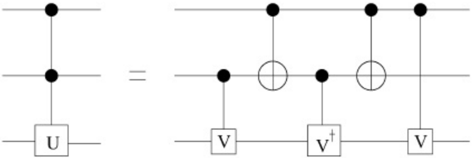


Quantum gates

Toffolli gate: $V = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$



C^2-U gate: $V^2 = U$



Quantum circuit model

Universal quantum gates

Quantum circuit model

Universal quantum gates

- A generic unitary operator on n-qubit systems can be decomposed by means of C^k-U gates,

Quantum circuit model

Universal quantum gates

- A generic unitary operator on n-qubit systems can be decomposed by means of C^k-U gates,
- any C^k-U gate ($k > 2$) can be decomposed using Toffoli gate and controlled-U gates,

Quantum circuit model

Universal quantum gates

- A generic unitary operator on n-qubit systems can be decomposed by means of C^k-U gates,
- any C^k-U gate $(k > 2)$ can be decomposed using Toffoli gate and controlled-U gates,
- the Toffoli gate can be implemented using CNOT, controlled-U, and Hadamard gates

Quantum circuit model

Universal quantum gates

- A generic unitary operator on n-qubit systems can be decomposed by means of C^k-U gates,
- any C^k-U gate ($k > 2$) can be decomposed using Toffoli gate and controlled-U gates,
- the Toffoli gate can be implemented using CNOT, controlled-U, and Hadamard gates
- any single-qubit rotation U, the controlled-U can be decomposed into single-qubit and CNOT gates

Quantum circuit model

Universal quantum gates

- A generic unitary operator on n-qubit systems can be decomposed by means of C^k-U gates,
- any C^k-U gate ($k > 2$) can be decomposed using Toffoli gate and controlled-U gates,
- the Toffoli gate can be implemented using CNOT, controlled-U, and Hadamard gates
- any single-qubit rotation U, the controlled-U can be decomposed into single-qubit and CNOT gates

Equivalence A k tape QTM running for T steps can be simulated by a quantum circuit with accuracy ϵ, and size $O(T^2 \log^{O(1)} \epsilon)$.

Quantum circuit model

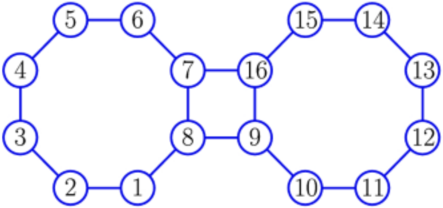
Universal quantum gates

- A generic unitary operator on n-qubit systems can be decomposed by means of C^k-U gates,
- any C^k-U gate ($k > 2$) can be decomposed using Toffoli gate and controlled-U gates,
- the Toffoli gate can be implemented using CNOT, controlled-U, and Hadamard gates
- any single-qubit rotation U, the controlled-U can be decomposed into single-qubit and CNOT gates

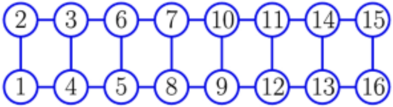
Equivalence A k tape QTM running for T steps can be simulated by a quantum circuit with accuracy ϵ, and size $O(T^2 \log^{O(1)} \epsilon)$.
Homework Circuit complexity, Query complexity

Challenge for NISQ computers?

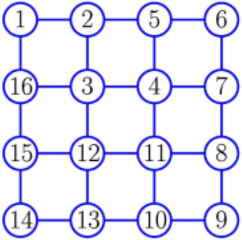
- ▷ limited connectivity between qubits: the coupling constraints



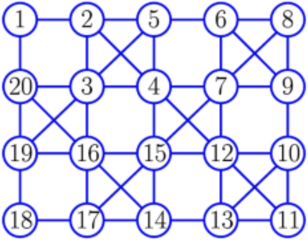
(a) Rigetti 16Q-Aspen



(b) IBM QX5



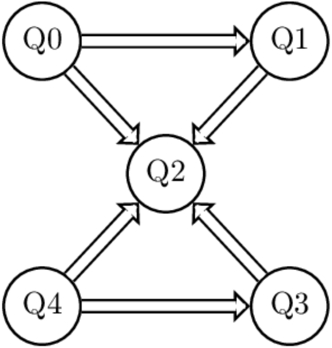
(c) 16q-Square



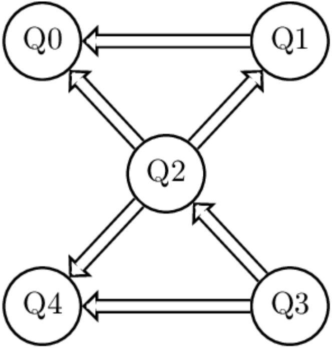
(d) IBM QX20 Tokyo

Challenge for NISQ computers?

- ▷ timespace coordinates



QX2



QX4

Information

What is information?

Information

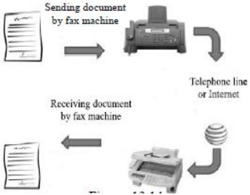
What is information?

- Informal way of thinking about information - listen to a song

Information

What is information?

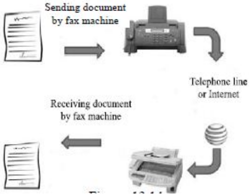
- ▷ Informal way of thinking about information - listen to a song
- ▷ Think of a fax machine - the font size of the words



Information

What is information?

- ▷ Informal way of thinking about information - listen to a song
- ▷ Think of a fax machine - the font size of the words

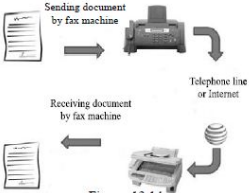


- ▷ Physics of information - how to store and process?

Information

What is information?

- ▷ Informal way of thinking about information - listen to a song
- ▷ Think of a fax machine - the font size of the words



- ▷ Physics of information - how to store and process?
  - ▽ Claude E Shannon (1916 - 2001) - father of information theory
  - ▽ A Mathematical Theory of Communication

Content of the course

Shannon's theory/model

- Information is uncertainty - information source is modeled as a random variable/process

Content of the course

Shannon's theory/model

- Information is uncertainty - information source is modeled as a random variable/process
- Information should be digital - ASCII

Content of the course

Shannon's theory/model

- Information is uncertainty - information source is modeled as a random variable/process
- Information should be digital - ASCII

Shannon's theorems

- Source coding theorem - *entropy* as a fundamental measure of information

Content of the course

Shannon's theory/model

- Information is uncertainty - information source is modeled as a random variable/process
- Information should be digital - ASCII

Shannon's theorems

- Source coding theorem - *entropy* as a fundamental measure of information
- Channel coding theorem - the *capacity* of a channel - reliable information with unreliable channel

Content of the course

Shannon's theory/model

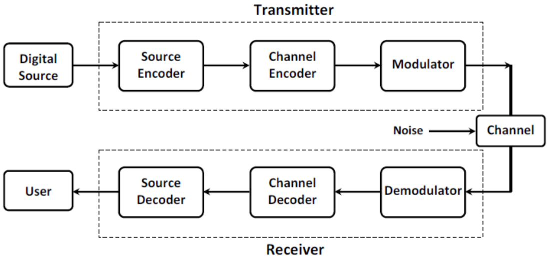
- Information is uncertainty - information source is modeled as a random variable/process
- Information should be digital - ASCII

Shannon's theorems

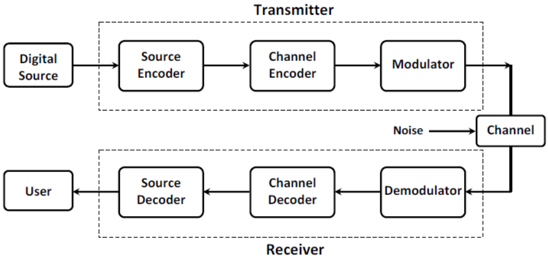
- Source coding theorem - *entropy* as a fundamental measure of information
- Channel coding theorem - the *capacity* of a channel - reliable information with unreliable channel

Quantum entropy existed before classical entropy!!

Model of a digital communication system

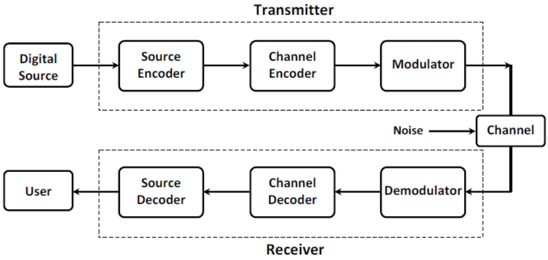


Model of a digital communication system



Example Let $S = \{a_1, \ldots, a_k\}$ denote the source i.e. a random variable X with sample space S and pmf p.

Model of a digital communication system



Example Let $S = \{a_1, \ldots, a_k\}$ denote the source i.e. a random variable X with sample space S and pmf p.

The encoding paradigm: Here

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

- H must be a continuous function of p_is

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

- H must be a continuous function of p_is
- H must be an increasing function of n when $p(a_i) = 1/n, 1 \leq i \leq n$

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

- H must be a continuous function of p_is
- H must be an increasing function of n when $p(a_i) = 1/n, 1 \leq i \leq n$
- Bundling/bucketing property

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

- ▷ H must be a continuous function of p_is
- ▷ H must be an increasing function of n when $p(a_i) = 1/n, 1 \leq i \leq n$
- ▷ Bundling/bucketing property

Question How much information is revealed when we know outcome of a random experiment?

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

- H must be a continuous function of p_is
- H must be an increasing function of n when $p(a_i) = 1/n, 1 \leq i \leq n$
- Bundling/bucketing property

Question How much information is revealed when we know outcome of a random experiment? How surprised are we?

Entropy

Shannon's assumptions - Let $\mathcal{X} = \{a_1, \ldots, a_n\}$ with pmf $p(a_i), 1 \leq i \leq n$. Denote the entropy by H

- H must be a continuous function of p_is
- H must be an increasing function of n when $p(a_i) = 1/n, 1 \leq i \leq n$
- Bundling/bucketing property

Question How much information is revealed when we know outcome of a random experiment? How surprised are we?

Question How much surprised you are if India wins in a football match against Argentina?

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x \in \mathcal{X}} p(x) \cdot \underbrace{\log_2 \left(\frac{1}{p(x)} \right)}_{\text{surprise}}$$

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x \in \mathcal{X}} p(x) \cdot \underbrace{\log_2\left(\frac{1}{p(x)}\right)}_{\text{surprise}} = -\sum_{x \in \mathcal{X}} p(x) \cdot \log_2(p(x)) \text{ bits}$$

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x \in \mathcal{X}} p(x) \cdot \underbrace{\log_2\left(\frac{1}{p(x)}\right)}_{\text{surprise}} = -\sum_{x \in \mathcal{X}} p(x) \cdot \log_2(p(x)) \text{ bits}$$

Example Consider the entropy of coin toss with p as probability of head.

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x \in \mathcal{X}} p(x) \cdot \underbrace{\log_2 \left(\frac{1}{p(x)} \right)}_{\text{surprise}} = - \sum_{x \in \mathcal{X}} p(x) \cdot \log_2(p(x)) \text{ bits}$$

Example Consider the entropy of coin toss with p as probability of head. What happens if $p_i = 1/n$?

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x\in\mathcal{X}} p(x) \cdot \underbrace{\log_2\left(\frac{1}{p(x)}\right)}_{\text{surprise}} = -\sum_{x\in\mathcal{X}} p(x) \cdot \log_2(p(x)) \text{ bits}$$

Example Consider the entropy of coin toss with p as probability of head. What happens if $p_i = 1/n$?

Proposition $0 \leq H(X) \leq \log(|\mathcal{X}|)$

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x\in\mathcal{X}} p(x) \cdot \underbrace{\log_2\left(\frac{1}{p(x)}\right)}_{\text{surprise}} = -\sum_{x\in\mathcal{X}} p(x) \cdot \log_2(p(x)) \text{ bits}$$

Example Consider the entropy of coin toss with p as probability of head. What happens if $p_i = 1/n$?

Proposition $0 \leq H(X) \leq \log(|\mathcal{X}|)$

Proof Let Y be a rv which takes the value $1/p(x)$ with probability $p(x)$. Then

$$\sum_{x\in\mathcal{X}} p(x) \cdot \log\left(\frac{1}{p(x)}\right) = \mathbb{E}[\log(Y)] \leq \log(\mathbb{E}[Y])$$

Entropy

Entropy Suppose X is a rv distributed over $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that each value $x \in \mathcal{X}$ occurs with probability $p(x)$. Then the entropy of X is

$$H(X) = \sum_{x\in\mathcal{X}} p(x) \cdot \underbrace{\log_2\left(\frac{1}{p(x)}\right)}_{\text{surprise}} = -\sum_{x\in\mathcal{X}} p(x) \cdot \log_2(p(x)) \text{ bits}$$

Example Consider the entropy of coin toss with p as probability of head. What happens if $p_i = 1/n$?

Proposition $0 \leq H(X) \leq \log(|\mathcal{X}|)$

Proof Let Y be a rv which takes the value $1/p(x)$ with probability $p(x)$. Then

$$\sum_{x\in\mathcal{X}} p(x) \cdot \log\left(\frac{1}{p(x)}\right) = \mathbb{E}[\log(Y)] \leq \log(\mathbb{E}[Y]) = \log\left(\sum_{x\in\mathcal{X}} p(x) \cdot \frac{1}{p(x)}\right)$$

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Question What is the operational meaning of entropy?

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Question What is the operational meaning of entropy?
Answer the rv X takes $H(X)$ bits to describe on average - is the fundamental limit for the compression rate in the iid setting

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Question What is the operational meaning of entropy?
Answer the rv X takes $H(X)$ bits to describe on average - is the fundamental limit for the compression rate in the iid setting

Code A code for a set $\mathcal{X}$ over an alphabet Σ is a map $C : \mathcal{X} \to \Sigma^*$ which maps each element of $\mathcal{X}$ to a finite string of elements of Σ.

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Question What is the operational meaning of entropy?
Answer the rv X takes $H(X)$ bits to describe on average - is the fundamental limit for the compression rate in the iid setting

Code A code for a set $\mathcal{X}$ over an alphabet Σ is a map $C : \mathcal{X} \to \Sigma^*$ which maps each element of $\mathcal{X}$ to a finite string of elements of Σ. $C(x)$ is called the codeword of x

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Question What is the operational meaning of entropy?
Answer the rv X takes $H(X)$ bits to describe on average - is the fundamental limit for the compression rate in the iid setting

Code A code for a set $\mathcal{X}$ over an alphabet Σ is a map $C : \mathcal{X} \to \Sigma^*$ which maps each element of $\mathcal{X}$ to a finite string of elements of Σ. $C(x)$ is called the codeword of x

Prefix-free code A code is prefix-free if for any $x, y \in \mathcal{X}$ such that $x \neq y, C(x)$ is not a prefix of $C(y)$ i.e. $C(y) \neq C(x) \circ \sigma$ for any $\sigma \in \Sigma^*$

Entropy

Source coding - How many bits are required to describe $n = 2^k$ outcomes of an experiment?

Question What is the operational meaning of entropy?
Answer the rv X takes $H(X)$ bits to describe on average - is the fundamental limit for the compression rate in the iid setting

Code A code for a set $\mathcal{X}$ over an alphabet Σ is a map $C : \mathcal{X} \to \Sigma^*$ which maps each element of $\mathcal{X}$ to a finite string of elements of Σ. $C(x)$ is called the codeword of x

Prefix-free code A code is prefix-free if for any $x, y \in \mathcal{X}$ such that $x \neq y, C(x)$ is not a prefix of $C(y)$ i.e. $C(y) \neq C(x) \circ \sigma$ for any $\sigma \in \Sigma^*$

Example $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Entropy

Source coding

Question What is the advantage of have a prefix-free code?

Entropy

Source coding

Question What is the advantage of have a prefix-free code?

Question Does a prefix-free code always exist for a given source, pmf and alphabet ?

Entropy

Source coding

Question What is the advantage of have a prefix-free code?

Question Does a prefix-free code always exist for a given source, pmf and alphabet ? If exists, how do we decide the length of the codewords?

Entropy

Source coding

Question What is the advantage of have a prefix-free code?

Question Does a prefix-free code always exist for a given source, pmf and alphabet ? If exists, how do we decide the length of the codewords?

Proposition (Kraft's inequality) Let $|\mathcal{X}| = n$. Then there exists a prefix-free code for $\mathcal{X}$ over $\Sigma = \{0, 1\}$ with codeword lengths $l_1, \ldots, l_n$ if and only if

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1.$$

Entropy

Source coding

Question What is the advantage of have a prefix-free code?

Question Does a prefix-free code always exist for a given source, pmf and alphabet ? If exists, how do we decide the length of the codewords?

Proposition (Kraft's inequality) Let $|\mathcal{X}| = n$. Then there exists a prefix-free code for $\mathcal{X}$ over $\Sigma = \{0, 1\}$ with codeword lengths $l_1, \ldots, l_n$ if and only if

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1.$$

For any alphabet Σ, replace 2^{l_i} by $|\Sigma|^{l_i}$.

Entropy

Source coding

Proposition Let X be a random variable taking values in $\mathcal{X}$, and let $C : \mathcal{X} \to \{0, 1\}$. Then the expected number of bits used by C to communicate the value of X is at least $H(X)$.

Entropy

Source coding

Proposition Let X be a random variable taking values in $\mathcal{X}$, and let $C : \mathcal{X} \to \{0,1\}$. Then the expected number of bits used by C to communicate the value of X is at least $H(X)$.

Proof the expected number of bits is $\sum_{x\in\mathcal{X}} p(x)\cdot|C(x)|$. Then

$$
\begin{aligned}
H(X) - \sum_{x\in\mathcal{X}} p(x)\cdot|C(x)| &= \sum_{x\in\mathcal{X}} p(x)\cdot\left(\log\left(\frac{1}{p(x)}\right) - |C(x)|\right) \\
&= \sum_{x\in\mathcal{X}} p(x)\cdot\log\left(\frac{1}{p(x)\cdot 2^{|C(x)|}}\right) \qquad (4)
\end{aligned}
$$

Entropy

Source coding

Proposition Let X be a random variable taking values in $\mathcal{X}$, and let $C : \mathcal{X} \to \{0,1\}$. Then the expected number of bits used by C to communicate the value of X is at least $H(X)$.

Proof the expected number of bits is $\sum_{x\in\mathcal{X}} p(x)\cdot|C(x)|$. Then

$$
\begin{aligned}
H(X) - \sum_{x\in\mathcal{X}} p(x)\cdot|C(x)| &= \sum_{x\in\mathcal{X}} p(x)\cdot\left(\log\left(\frac{1}{p(x)}\right) - |C(x)|\right) \\
&= \sum_{x\in\mathcal{X}} p(x)\cdot\log\left(\frac{1}{p(x)\cdot 2^{|C(x)|}}\right) \qquad (4)
\end{aligned}
$$

Now let Y be the rv which takes the value $\frac{1}{p(x)\cdot 2^{|C(x)|}}$ with probability $p(x)$.

Entropy

Source coding

Proposition Let X be a random variable taking values in $\mathcal{X}$, and let $C : \mathcal{X} \to \{0,1\}$. Then the expected number of bits used by C to communicate the value of X is at least $H(X)$.

Proof the expected number of bits is $\sum_{x\in\mathcal{X}} p(x)\cdot|C(x)|$. Then

$$\begin{aligned} H(X) - \sum_{x\in\mathcal{X}} p(x)\cdot|C(x)| &= \sum_{x\in\mathcal{X}} p(x)\cdot\left(\log\left(\frac{1}{p(x)}\right) - |C(x)|\right) \\ &= \sum_{x\in\mathcal{X}} p(x)\cdot\log\left(\frac{1}{p(x)\cdot 2^{|C(x)|}}\right) \end{aligned} \tag{4}$$

Now let Y be the rv which takes the value $\frac{1}{p(x)\cdot 2^{|C(x)|}}$ with probability $p(x)$. Then

$$\mathbb{E}[\log(Y)] \leq \log(\mathbb{E}[Y]) = \log\left(\sum_{x\in\mathcal{X}} p(x)\cdot\frac{1}{p(x)\cdot 2^{|C(x)|}}\right) = \log\left(\sum_{x\in\mathcal{X}} \frac{1}{2^{|C(x)|}}\right)$$

Entropy

Question revisited - $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Entropy

Question revisited - $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Answer $a = 0$, $b = 10$, $c = 110$, $d = 111$

Entropy

Question revisited - $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Answer $a = 0$, $b = 10$, $c = 110$, $d = 111$

The Shannon code A prefix-free code for a rv X with at most $H(X) + 1$ bits on average can be constructed, known as Shannon code.

Entropy

Question revisited - $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Answer $a = 0$, $b = 10$, $c = 110$, $d = 111$

The Shannon code A prefix-free code for a rv X with at most $H(X) + 1$ bits on average can be constructed, known as Shannon code.
For an element $x \in \mathcal{X}$, which occurs with probability $p(x)$, use a codeword of length $\lceil \log(1/p(x)) \rceil$.

Entropy

Question revisited - $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Answer $a = 0$, $b = 10$, $c = 110$, $d = 111$

The Shannon code A prefix-free code for a rv X with at most $H(X) + 1$ bits on average can be constructed, known as Shannon code.
For an element $x \in \mathcal{X}$, which occurs with probability $p(x)$, use a codeword of length $\lceil \log(1/p(x)) \rceil$. By Kraft's inequality, such a prefix-free code since

$$\sum_{x\in\mathcal{X}} \frac{1}{2^{|C(x)|}} = \sum_{x\in\mathcal{X}} \frac{1}{2^{\lceil \log(1/p(x)) \rceil}} \leq \sum_{x\in\mathcal{X}} \frac{1}{2^{\log(1/p(x))}} = \sum_{x\in\mathcal{X}} p(x) = 1$$

Entropy

Question revisited - $\Sigma = \{0, 1\}$. Let $\mathcal{X} = \{a, b, c, d\}$ with $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = 1/8$ and $p(d) = 1/8$. How do we design a code for $\mathcal{X}$ such that expected length of the code is minimized?

Answer $a = 0$, $b = 10$, $c = 110$, $d = 111$

The Shannon code A prefix-free code for a rv X with at most $H(X) + 1$ bits on average can be constructed, known as Shannon code.
For an element $x \in \mathcal{X}$, which occurs with probability $p(x)$, use a codeword of length $\lceil \log(1/p(x)) \rceil$. By Kraft's inequality, such a prefix-free code since

$$\sum_{x \in \mathcal{X}} \frac{1}{2^{|C(x)|}} = \sum_{x \in \mathcal{X}} \frac{1}{2^{\lceil \log(1/p(x)) \rceil}} \leq \sum_{x \in \mathcal{X}} \frac{1}{2^{\log(1/p(x))}} = \sum_{x \in \mathcal{X}} p(x) = 1$$

the expected number of bits used is

$$\sum_{x \in \mathcal{X}} p(x) \cdot \lceil \log(1/p(x)) \rceil \leq \sum_{x \in \mathcal{X}} p(x) \cdot (\log(1/p(x)) + 1) = H(X) + 1.$$

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect?

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect? What is a channel?

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect? What is a channel?

Joint entropy Let $Z = (X, Y)$ be a pair of random variables with joint distribution $p(x, y)$. Then

$$H(Z) = H(X, Y) = \sum_{x,y} p(x, y) \log(1/p(x, y))$$

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect? What is a channel?

Joint entropy Let $Z = (X, Y)$ be a pair of random variables with joint distribution $p(x, y)$. Then

$$\begin{aligned} H(Z) &= H(X,Y) = \sum_{x,y} p(x,y) \log(1/p(x,y)) \\ &= \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(x)} + \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(y|x)} \end{aligned}$$

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect? What is a channel?

Joint entropy Let $Z = (X, Y)$ be a pair of random variables with joint distribution $p(x, y)$. Then

$$\begin{aligned} H(Z) &= H(X, Y) = \sum_{x,y} p(x, y) \log(1/p(x, y)) \\ &= \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(x)} + \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(y|x)} \\ &= \sum_{x} p(x) \log \frac{1}{p(x)} \sum_{y} p(y|x) + \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(y|x)} \end{aligned}$$

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect? What is a channel?

Joint entropy Let $Z = (X, Y)$ be a pair of random variables with joint distribution $p(x, y)$. Then

$$
\begin{aligned}
H(Z) &= H(X,Y) = \sum_{x,y} p(x,y) \log(1/p(x,y)) \\
&= \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(x)} + \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(y|x)} \\
&= \sum_{x} p(x) \log \frac{1}{p(x)} \sum_{y} p(y|x) + \sum_{x,y} p(x)p(y|x) \log \frac{1}{p(y|x)} \\
&= H(X) + \sum_{x} p(x) H(Y|X=x)
\end{aligned}
$$

Entropy

Communication Suppose we have a source rv X and at the receiver end an output rv Y. The source letters are being transmitted through the channel. What do we expect? What is a channel?

Joint entropy Let $Z = (X, Y)$ be a pair of random variables with joint distribution $p(x, y)$. Then

$$\begin{aligned} H(Z) &= H(X,Y) = \sum_{x,y} p(x,y)\log(1/p(x,y)) \\ &= \sum_{x,y} p(x)p(y|x)\log\frac{1}{p(x)} + \sum_{x,y} p(x)p(y|x)\log\frac{1}{p(y|x)} \\ &= \sum_{x} p(x)\log\frac{1}{p(x)}\sum_{y} p(y|x) + \sum_{x,y} p(x)p(y|x)\log\frac{1}{p(y|x)} \\ &= H(X) + \sum_{x} p(x)H(Y|X=x) \\ &= H(X) + \mathbb{E}_x[H(Y|X=x)] \end{aligned}$$

Entropy

Chain rule of entropy

Set $H(Y|X) = \mathbb{E}_x[H(Y|X = x)]$. Then we have

$$H(X, Y) = H(X) + H(Y|X)$$

Entropy

Chain rule of entropy

Set $H(Y|X) = \mathbb{E}_x[H(Y|X = x)]$. Then we have

$$H(X, Y) = H(X) + H(Y|X)$$

Similarly, we can obtain

$$H(X, Y) = H(Y) + H(X|Y)$$

Entropy

Chain rule of entropy

Set $H(Y|X) = \mathbb{E}_x[H(Y|X = x)]$. Then we have

$$H(X, Y) = H(X) + H(Y|X)$$

Similarly, we can obtain

$$H(X, Y) = H(Y) + H(X|Y)$$

Homework Let (X, Y) be a joint random variable with $X \vee Y = 1$, $X \in \{0, 1\}$ and $Y \in \{0, 1\}$ such that $p(0, 1) = p(1, 0) = p(1, 1) = 1/3$. Then calculate $H(X)$, $H(Y)$, $H(Y|X = 0)$, $H(Y|X = 1)$, $H(Y|X)$, $H(X, Y)$

Entropy

Proposition $H(Y) \geq H(Y|X)$

Entropy

Proposition $H(Y) \geq H(Y|X)$

Proof

$$H(Y|X) - H(Y) = \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} - \sum_y p(y) \log \frac{1}{p(y)}$$

Entropy

Proposition $H(Y) \geq H(Y|X)$

Proof

$$\begin{aligned} H(Y|X) - H(Y) &= \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} - \sum_y p(y) \log \frac{1}{p(y)} \\ &= \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} \\ &\quad - \sum_y p(y) \log \frac{1}{p(y)} \sum_x p(x|y) \end{aligned}$$

Entropy

Proposition $H(Y) \geq H(Y|X)$

Proof

$$\begin{aligned}
H(Y|X) - H(Y) &= \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} - \sum_y p(y) \log \frac{1}{p(y)} \\
&= \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} \\
&\quad - \sum_y p(y) \log \frac{1}{p(y)} \sum_x p(x|y) \\
&= \sum_{x,y} p(x,y) \left(\log \frac{1}{p(y|x)} - \log \frac{1}{p(y)} \right)
\end{aligned}$$

Entropy

Proposition $H(Y) \geq H(Y|X)$

Proof

$$
\begin{aligned}
H(Y|X) - H(Y) &= \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} - \sum_y p(y) \log \frac{1}{p(y)} \\
&= \sum_x p(x) \sum_y p(y|x) \log \frac{1}{p(y|x)} \\
&\quad - \sum_y p(y) \log \frac{1}{p(y)} \sum_x p(x|y) \\
&= \sum_{x,y} p(x,y) \left(\log \frac{1}{p(y|x)} - \log \frac{1}{p(y)} \right) \\
&= \sum_{x,y} p(x,y) \left(\log \frac{p(x)p(y)}{p(x,y)} \right)
\end{aligned}
$$

Entropy

Now let W be a rv that takes the value $\frac{p(x)p(y)}{p(x,y)}$ with probability $p(x, y)$. Then using jensen's inequality

$$\sum_{x,y} p(x, y) \left(\log \frac{p(x)p(y)}{p(x, y)} \right) \leq \log \left(\sum_{x,y} \frac{p(x)p(y)}{p(x, y)} p(x, y) \right) = \log(1) = 0$$

Entropy

Now let W be a rv that takes the value $\frac{p(x)p(y)}{p(x,y)}$ with probability $p(x,y)$. Then using jensen's inequality

$$\sum_{x,y} p(x,y)\left(\log \frac{p(x)p(y)}{p(x,y)}\right) \leq \log\left(\sum_{x,y} \frac{p(x)p(y)}{p(x,y)} p(x,y)\right) = \log(1) = 0$$

Question What do you conclude ?

Entropy

Now let W be a rv that takes the value $\frac{p(x)p(y)}{p(x,y)}$ with probability $p(x,y)$. Then using jensen's inequality

$$\sum_{x,y} p(x,y)\left(\log \frac{p(x)p(y)}{p(x,y)}\right) \leq \log\left(\sum_{x,y} \frac{p(x)p(y)}{p(x,y)} p(x,y)\right) = \log(1) = 0$$

Question What do you conclude ?

Conditioning reduces entropy on average!!

Entropy

Now let W be a rv that takes the value $\frac{p(x)p(y)}{p(x,y)}$ with probability $p(x,y)$. Then using jensen's inequality

$$\sum_{x,y} p(x,y)\left(\log \frac{p(x)p(y)}{p(x,y)}\right) \leq \log\left(\sum_{x,y} \frac{p(x)p(y)}{p(x,y)} p(x,y)\right) = \log(1) = 0$$

Question What do you conclude ?

Conditioning reduces entropy on average!!
Homework $H(Y) = H(Y|X)$ if and only if X and Y are independent

Entropy

Now let W be a rv that takes the value $\frac{p(x)p(y)}{p(x,y)}$ with probability $p(x, y)$. Then using jensen's inequality

$$\sum_{x,y} p(x, y) \left(\log \frac{p(x)p(y)}{p(x, y)} \right) \leq \log \left(\sum_{x,y} \frac{p(x)p(y)}{p(x, y)} p(x, y) \right) = \log(1) = 0$$

Question What do you conclude ?

Conditioning reduces entropy on average!!
Homework $H(Y) = H(Y|X)$ if and only if X and Y are independent

Homework $H(Y|X, Z) \leq H(Y|Z)$

Entropy

General case Suppose $\overline{X} = (X_1, X_2, \ldots, X_m)$.

Entropy

General case Suppose $\overline{X} = (X_1, X_2, \ldots, X_m)$.

Homework Show (by induction) that

$$H(X_1, \ldots, X_m) = H(X_1) + H(X_2|X_1) + H(X_3|X_1, X_2) + \ldots$$

Entropy

General case Suppose $\overline{X} = (X_1, X_2, \ldots, X_m)$.

Homework Show (by induction) that

$$\begin{aligned} H(X_1, \ldots, X_m) &= H(X_1) + H(X_2|X_1) + H(X_3|X_1, X_2) + \ldots \\ &\quad + H(X_m|X_1, \ldots, X_{m-1}) \end{aligned}$$

Entropy

General case Suppose $\overline{X} = (X_1, X_2, \ldots, X_m)$.

Homework Show (by induction) that

$$\begin{aligned} H(X_1, \ldots, X_m) &= H(X_1) + H(X_2|X_1) + H(X_3|X_1, X_2) + \ldots \\ &\quad + H(X_m|X_1, \ldots, X_{m-1}) \end{aligned}$$

Sub-additive property of entropy

$$H(X_1, \ldots, X_m) \leq H(X_1) + H(X_2) + \ldots + H(X_m)$$

Entropy

General case Suppose $\overline{X} = (X_1, X_2, \ldots, X_m)$.

Homework Show (by induction) that

$$\begin{aligned} H(X_1, \ldots, X_m) &= H(X_1) + H(X_2|X_1) + H(X_3|X_1, X_2) + \ldots \\ &\quad + H(X_m|X_1, \ldots, X_{m-1}) \end{aligned}$$

Sub-additive property of entropy

$$H(X_1, \ldots, X_m) \leq H(X_1) + H(X_2) + \ldots + H(X_m)$$

Question Can the upper bound for expected code length of $H(X) + 1$ be improved?

Entropy

Recall

- ▷ Let X be a rv with range set $\{a_1, \ldots, a_n\}$ and $p(a_i) = p_i$

Entropy

Recall

- Let X be a rv with range set $\{a_1, \ldots, a_n\}$ and $p(a_i) = p_i$
- We want to encode a_is with expected code length small i.e. expected number of bits needed is small

Entropy

Recall

- ▷ Let X be a rv with range set $\{a_1, \ldots, a_n\}$ and $p(a_i) = p_i$
- ▷ We want to encode a_is with expected code length small i.e. expected number of bits needed is small
- ▷ If $l_1, l_2, \ldots, l_n$ are the codeword lengths for $a_1, \ldots, a_n$ respectively then

$$\sum_{i=1}^{n} 2^{l_i} \leq 1$$

Entropy

Recall

- Let X be a rv with range set $\{a_1, \dots, a_n\}$ and $p(a_i) = p_i$
- We want to encode a_is with expected code length small i.e. expected number of bits needed is small
- If $l_1, l_2, \dots, l_n$ are the codeword lengths for $a_1, \dots, a_n$ respectively then

$$\sum_{i=1}^{n} 2^{l_i} \leq 1$$

- We proved that the expected length is bounded below by $H(X)$ and bounded above by $H(X) + 1$ (Shannon code)

Entropy

Recall

- Let X be a rv with range set $\{a_1, \dots, a_n\}$ and $p(a_i) = p_i$
- We want to encode a_is with expected code length small i.e. expected number of bits needed is small
- If $l_1, l_2, \dots, l_n$ are the codeword lengths for $a_1, \dots, a_n$ respectively then

$$\sum_{i=1}^{n} 2^{l_i} \leq 1$$

- We proved that the expected length is bounded below by $H(X)$ and bounded above by $H(X) + 1$ (Shannon code)

Question Can we improve the upper bound?

Entropy

The idea - Source Coding Theorem

Entropy

The idea - Source Coding Theorem

- Consider m copies of the rv X, $X_1, \ldots, X_m$ and a code $C : \mathcal{X}^m \to \{0,1\}^*$

Entropy

The idea - Source Coding Theorem

- Consider m copies of the rv X, $X_1, \ldots, X_m$ and a code $C : \mathcal{X}^m \to \{0,1\}^*$
- Let $|\mathcal{X}|^m = N$

Entropy

The idea - Source Coding Theorem

- Consider m copies of the rv X, $X_1, \ldots, X_m$ and a code $C : \mathcal{X}^m \to \{0,1\}^*$
- Let $|\mathcal{X}|^m = N$
- We know that (Homework)

$$\mathbb{E}[|C(X_1, \ldots, X_m)|] \leq \sum_{i=1}^{N} p_i \lceil \log \frac{1}{p_i} \rceil \leq H(X_1, \ldots, X_m) + 1$$

Entropy

The idea - Source Coding Theorem

- Consider m copies of the rv X, $X_1, \ldots, X_m$ and a code $C : \mathcal{X}^m \to \{0,1\}^*$
- Let $|\mathcal{X}|^m = N$
- We know that (Homework)

$$\mathbb{E}[|C(X_1, \ldots, X_m)|] \leq \sum_{i=1}^{N} p_i \lceil \log \frac{1}{p_i} \rceil \leq H(X_1, \ldots, X_m) + 1$$

- Assume that m copies of X are iid

Entropy

The idea - Source Coding Theorem

- Consider m copies of the rv X, $X_1, \ldots, X_m$ and a code $C : \mathcal{X}^m \to \{0,1\}^*$
- Let $|\mathcal{X}|^m = N$
- We know that (Homework)

$$\mathbb{E}[|C(X_1, \ldots, X_m)|] \leq \sum_{i=1}^{N} p_i \lceil \log \frac{1}{p_i} \rceil \leq H(X_1, \ldots, X_m) + 1$$

- Assume that m copies of X are iid
- Then

$$\begin{aligned} H(X_1, \ldots, X_m) &= H(X_1) + H(X_2|X_1) + \ldots + H(X_m|X_1, \ldots, X_{m-1}) \\ &= H(X_1) + H(X_2) + \ldots + H(X_m) \\ &= m \cdot H(X) \end{aligned}$$

Entropy

Thus we have

$$\mathbb{E}[|C(X_1, \ldots, X_m)|] \leq m \cdot H(X) + 1$$

Entropy

Thus we have

$$\mathbb{E}[|C(X_1, \ldots, X_m)|] \leq m \cdot H(X) + 1$$

Thus we conclude that we can use $H(X) + \frac{1}{m}$ bits on average per copy of X

Entropy

Thus we have

$$\mathbb{E}[|C(X_1, \ldots, X_m)|] \leq m \cdot H(X) + 1$$

Thus we conclude that we can use $H(X) + \frac{1}{m}$ bits on average per copy of X

Theorem (Fundamental Source Coding Theorem (Shannon)). For any $\epsilon > 0$ there exists a n_0 such that for all $n \geq n_0$ and given n copies of X, $X_1, \ldots, X_n$ sampled i.i.d., it is possible to communicate $(X_1, \ldots, X_n)$ using at most $H(X) + \epsilon$ bits per copy on average.

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X; Y) = H(X) - H(X|Y)$$

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y)$$

Question What is the difference between correlation and MI?

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y)$$

Question What is the difference between correlation and MI?

Example X represents the roll of a fair 6-sided die, and Y represents whether the roll is even (0 if even, 1 if odd)

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y)$$

Question What is the difference between correlation and MI?

Example X represents the roll of a fair 6-sided die, and Y represents whether the roll is even (0 if even, 1 if odd)

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X,Y)$$

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y)$$

Question What is the difference between correlation and MI?
Example X represents the roll of a fair 6-sided die, and Y represents whether the roll is even (0 if even, 1 if odd)

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X,Y)$$

Then (expanding the formula of entropy)
$\rightarrow\ I(X;Y) \geq 0$

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y)$$

Question What is the difference between correlation and MI?
Example X represents the roll of a fair 6-sided die, and Y represents whether the roll is even (0 if even, 1 if odd)

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X,Y)$$

Then (expanding the formula of entropy)

$\rightarrow$ $I(X;Y) \geq 0$

$\rightarrow$ $I(X;Y) = I(Y;X)$

Mutual information

The mutual information (MI) between two random variables X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y)$$

Question What is the difference between correlation and MI?
Example X represents the roll of a fair 6-sided die, and Y represents whether the roll is even (0 if even, 1 if odd)

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X,Y)$$

Then (expanding the formula of entropy)

$\rightarrow$ $I(X;Y) \geq 0$

$\rightarrow$ $I(X;Y) = I(Y;X)$

Homework Let X, Y be two variables with $X \vee Y = 1$, $X \in \{0,1\}$, $Y \in \{0,1\}$ such that $(X,Y) = (1,0)$, $(X,Y) = (0,1)$ and $(X,Y) = (1,1)$ with probabilities $1/3$. Then calculate $I(X;Y)$

Mutual information

Conditional mutual information

$$I(X;Y|Z) \quad = \quad \mathbb{E}_Z[I(X|Z=z;Y|Z=z)]$$

Mutual information

Conditional mutual information

$$
\begin{aligned}
I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z;Y|Z=z)] \\
&= \mathbb{E}_Z[H(X|Z=z) - H(X|Y,Z=z)]
\end{aligned}
$$

Mutual information

Conditional mutual information

$$\begin{aligned} I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z;Y|Z=z)] \\ &= \mathbb{E}_Z[H(X|Z=z) - H(X|Y,Z=z)] \\ &= H(X|Z) - H(X|Y,Z) \end{aligned}$$

Mutual information

Conditional mutual information

$$\begin{aligned} I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z;Y|Z=z)] \\ &= \mathbb{E}_Z[H(X|Z=z) - H(X|Y,Z=z)] \\ &= H(X|Z) - H(X|Y,Z) \end{aligned}$$

Example Let (X, Y, Z) be a random variable with $Z = X \oplus Y$, $X \in \{0,1\}$, $Y \in \{0,1\}$ such that $(X, Y, Z) = (x, y, z)$ are equally likely. Then check that $I(X;Y) = 0$ and

$$I(X;Y|Z) = \mathbb{E}_Z[I(X|Z=z);Y|Z=z]$$

Mutual information

Conditional mutual information

$$\begin{aligned} I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z;Y|Z=z)] \\ &= \mathbb{E}_Z[H(X|Z=z) - H(X|Y,Z=z)] \\ &= H(X|Z) - H(X|Y,Z) \end{aligned}$$

Example Let (X,Y,Z) be a random variable with $Z = X \oplus Y$, $X \in \{0,1\}$, $Y \in \{0,1\}$ such that $(X,Y,Z) = (x,y,z)$ are equally likely. Then check that $I(X;Y) = 0$ and

$$\begin{aligned} I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z);Y|Z=z] \\ &= \frac{1}{2}I(X|Z=0;Y|Z=0) + \frac{1}{2}I(X|Z=1;Y|Z=1) \end{aligned}$$

Mutual information

Conditional mutual information

$$
\begin{aligned}
I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z;Y|Z=z)] \\
&= \mathbb{E}_Z[H(X|Z=z) - H(X|Y,Z=z)] \\
&= H(X|Z) - H(X|Y,Z)
\end{aligned}
$$

Example Let (X, Y, Z) be a random variable with $Z = X \oplus Y$, $X \in \{0,1\}$, $Y \in \{0,1\}$ such that $(X, Y, Z) = (x, y, z)$ are equally likely. Then check that $I(X;Y) = 0$ and

$$
\begin{aligned}
I(X;Y|Z) &= \mathbb{E}_Z[I(X|Z=z);Y|Z=z] \\
&= \frac{1}{2}I(X|Z=0;Y|Z=0) + \frac{1}{2}I(X|Z=1;Y|Z=1) \\
&= \frac{1}{2}\log 2 + \frac{1}{2}\log 2 = 1
\end{aligned}
$$

Mutual information

Question What is the conclusion from the above example?

Mutual information

Question What is the conclusion from the above example?

Chain rule of MI: $I((X_1, \ldots, X_m); Y) = \sum_{i=1}^{m} I(X_i; Y | X_1, \ldots, X_{i-1})$

Mutual information

Question What is the conclusion from the above example?

Chain rule of MI: $I((X_1, \dots, X_m); Y) = \sum_{i=1}^{m} I(X_i; Y | X_1, \dots, X_{i-1})$

Proof

$$\begin{aligned} & I((X_1, \dots, X_m); Y) \\ = \; & H(X_1, \dots, X_m) - H(X_1, \dots, X_m | Y) \end{aligned}$$

Mutual information

Question What is the conclusion from the above example?

Chain rule of MI: $I((X_1,\ldots,X_m);Y) = \sum_{i=1}^m I(X_i;Y|X_1,\ldots,X_{i-1})$

Proof

$$
\begin{aligned}
& I((X_1,\ldots,X_m);Y) \\
= \; & H(X_1,\ldots,X_m) - H(X_1,\ldots,X_m|Y) \\
= \; & \sum_{i=1}^m H(X_i|X_1,\ldots,X_{i-1}) - \sum_{i=1}^m H(X_i|Y,X_1,\ldots,X_{i-1})
\end{aligned}
$$

Mutual information

Question What is the conclusion from the above example?

Chain rule of MI: $I((X_1,\ldots,X_m);Y) = \sum_{i=1}^m I(X_i;Y|X_1,\ldots,X_{i-1})$

Proof

$$\begin{aligned}
& I((X_1,\ldots,X_m);Y) \\
= \; & H(X_1,\ldots,X_m) - H(X_1,\ldots,X_m|Y) \\
= \; & \sum_{i=1}^m H(X_i|X_1,\ldots,X_{i-1}) - \sum_{i=1}^m H(X_i|Y,X_1,\ldots,X_{i-1}) \\
= \; & \sum_{i=1}^m [H(X_i|X_1,\ldots,X_{i-1}) - H(X_i|Y,X_1,\ldots,X_{i-1})]
\end{aligned}$$

Mutual information

Question What is the conclusion from the above example?

Chain rule of MI: $I((X_1, \ldots, X_m); Y) = \sum_{i=1}^m I(X_i; Y|X_1, \ldots, X_{i-1})$

Proof

$$
\begin{aligned}
& I((X_1, \ldots, X_m); Y) \\
= \; & H(X_1, \ldots, X_m) - H(X_1, \ldots, X_m|Y) \\
= \; & \sum_{i=1}^m H(X_i|X_1, \ldots, X_{i-1}) - \sum_{i=1}^m H(X_i|Y, X_1, \ldots, X_{i-1}) \\
= \; & \sum_{i=1}^m [H(X_i|X_1, \ldots, X_{i-1}) - H(X_i|Y, X_1, \ldots, X_{i-1})] \\
= \; & \sum_{i=1}^m I(X_i; Y|X_1, \ldots, X_{i-1})
\end{aligned}
$$

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X; Y) \geq I(X; Z)$.

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X; Y) \geq I(X; Z)$.

Proof Let $Z = g(Y)$ for some g then obviously $X \to Y \to g(Y)$.

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X;Y) \geq I(X;Z)$.

Proof Let $Z = g(Y)$ for some g then obviously $X \to Y \to g(Y)$.

$$I(X;Y) \;=\; H(X) - H(X|Y) = H(X) - H(X|Y, g(Y))$$

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X;Y) \geq I(X;Z)$.

Proof Let $Z = g(Y)$ for some g then obviously $X \to Y \to g(Y)$.

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) = H(X) - H(X|Y, g(Y)) \\ &\geq H(X) - H(X|g(Y)) = I(X;g(Y)) \end{aligned}$$

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X;Y) \geq I(X;Z)$.

Proof Let $Z = g(Y)$ for some g then obviously $X \to Y \to g(Y)$.

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) = H(X) - H(X|Y, g(Y)) \\ &\geq H(X) - H(X|g(Y)) = I(X; g(Y)) \end{aligned}$$

From the first line, $I(X;Y) = I(X;(Y, g(Y))) = I(X;(Y,Z))$

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X; Y) \geq I(X; Z)$.

Proof Let $Z = g(Y)$ for some g then obviously $X \to Y \to g(Y)$.

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) = H(X) - H(X|Y, g(Y)) \\ &\geq H(X) - H(X|g(Y)) = I(X; g(Y)) \end{aligned}$$

From the first line, $I(X;Y) = I(X;(Y, g(Y))) = I(X;(Y,Z))$ However, in general,

$$I(X;(Y,Z)) = I(X;Y) + I(X;Z|Y) = I(X;Y)$$

Mutual information

Markov chain (a memoryless process) An ordered tuple of random variables (X, Y, Z) is said to form a Markov chain if X and Z are independent conditioned on Y. In that case we write as $X \to Y \to Z$.

Question If $X \to Y \to Z$ then $Z \to Y \to X$?

Lemma Data Processing Inequality: Let $X \to Y \to Z$ be a Markov chain. Then $I(X;Y) \geq I(X;Z)$.

Proof Let $Z = g(Y)$ for some g then obviously $X \to Y \to g(Y)$.

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) = H(X) - H(X|Y, g(Y)) \\ &\geq H(X) - H(X|g(Y)) = I(X; g(Y)) \end{aligned}$$

From the first line, $I(X;Y) = I(X;(Y, g(Y))) = I(X;(Y,Z))$ However, in general,

$$I(X;(Y,Z)) = I(X;Y) + I(X;Z|Y) = I(X;Y)$$

Thus,

$$I(X;Y) = I(X;(Y,Z)) = H(X) - H(X|Y,Z) \geq H(X) - H(X|Z) = I(X;Z)$$

Kullback Leibler (KL) divergence

Also known as *relative entropy* is a measure of how different two distributions are.

Kullback Leibler (KL) divergence

Also known as *relative entropy* is a measure of how different two distributions are.

Definition Let P and and Q be be two distributions on a sample space $\mathcal{X}$. The KL-divergence between P and Q is defined as:

$$D(P\|Q) = \sum_{x\in\mathcal{X}} p(x)\log\left(\frac{p(x)}{q(x)}\right)$$

Kullback Leibler (KL) divergence

Also known as *relative entropy* is a measure of how different two distributions are.

Definition Let P and and Q be be two distributions on a sample space $\mathcal{X}$. The KL-divergence between P and Q is defined as:

$$D(P\|Q) = \sum_{x\in\mathcal{X}} p(x)\log\left(\frac{p(x)}{q(x)}\right)$$

Example Suppose $\mathcal{X} = \{a, b, c\}$ with $p(x) = \frac{1}{3}$, $x \in \mathcal{X}$ and $q(a) = \frac{1}{2}$, $q(b) = \frac{1}{2}$, $q(c) = 0$. Then

Kullback Leibler (KL) divergence

Also known as *relative entropy* is a measure of how different two distributions are.

Definition Let P and and Q be be two distributions on a sample space $\mathcal{X}$. The KL-divergence between P and Q is defined as:

$$D(P\|Q) = \sum_{x\in\mathcal{X}} p(x)\log\left(\frac{p(x)}{q(x)}\right)$$

Example Suppose $\mathcal{X} = \{a, b, c\}$ with $p(x) = \frac{1}{3}$, $x \in \mathcal{X}$ and $q(a) = \frac{1}{2}$, $q(b) = \frac{1}{2}$, $q(c) = 0$. Then

$$\begin{aligned} D(P\|Q) &= \frac{2}{3}\log\frac{2}{3} + \infty = \infty \\ D(Q\|P) &= \log\frac{3}{2} + 0 = \log\frac{3}{2} \end{aligned}$$

KL divergence

$\rightarrow$ $D(P\|Q)$ and $D(Q\|P)$ are not necessarily equal

KL divergence

$\rightarrow$ $D(P\|Q)$ and $D(Q\|P)$ are not necessarily equal

$\rightarrow$ $D(P\|Q)$ may be infinite

KL divergence

- $\rightarrow$ $D(P\|Q)$ and $D(Q\|P)$ are not necessarily equal
- $\rightarrow$ $D(P\|Q)$ may be infinite
- $\rightarrow$ Let $\mathfrak{Supp}(P) = \{x : p(x) > 0\}$. Then we must have $\mathfrak{Supp}(P) \subseteq \mathfrak{Supp}(Q)$ if $D(P\|Q) < \infty$

KL divergence

- $\rightarrow$ $D(P\|Q)$ and $D(Q\|P)$ are not necessarily equal
- $\rightarrow$ $D(P\|Q)$ may be infinite
- $\rightarrow$ Let $\mathfrak{Supp}(P) = \{x : p(x) > 0\}$. Then we must have $\mathfrak{Supp}(P) \subseteq \mathfrak{Supp}(Q)$ if $D(P\|Q) < \infty$

Even though the KL-divergence is not symmetric, it is often used as a measure of "dissimilarity" between two distributions

KL divergence

- $\rightarrow$ $D(P\|Q)$ and $D(Q\|P)$ are not necessarily equal
- $\rightarrow$ $D(P\|Q)$ may be infinite
- $\rightarrow$ Let $\mathfrak{Supp}(P) = \{x : p(x) > 0\}$. Then we must have $\mathfrak{Supp}(P) \subseteq \mathfrak{Supp}(Q)$ if $D(P\|Q) < \infty$

Even though the KL-divergence is not symmetric, it is often used as a measure of "dissimilarity" between two distributions

Lemma Let P and Q be distributions on a finite space $\mathcal{X}$. Then $D(P\|Q) \geq 0$ with equality if and only if $P = Q$.

KL divergence

- $\rightarrow$ $D(P\|Q)$ and $D(Q\|P)$ are not necessarily equal
- $\rightarrow$ $D(P\|Q)$ may be infinite
- $\rightarrow$ Let $\mathfrak{Supp}(P) = \{x : p(x) > 0\}$. Then we must have $\mathfrak{Supp}(P) \subseteq \mathfrak{Supp}(Q)$ if $D(P\|Q) < \infty$

Even though the KL-divergence is not symmetric, it is often used as a measure of "dissimilarity" between two distributions

Lemma Let P and Q be distributions on a finite space $\mathcal{X}$. Then $D(P\|Q) \geq 0$ with equality if and only if $P = Q$.

$$\begin{aligned} D(P\|Q) = \sum_x p(x) \log \frac{p(x)}{q(x)} &= \sum_{x \in \mathfrak{Supp}(P)} p(x) \log \frac{p(x)}{q(x)} \\ &\geq -\log \left(\sum_{x \in \mathfrak{Supp}(P)} p(x) \cdot \frac{q(x)}{p(x)} \right) \\ &= -\log \left(\sum_{x \in \mathfrak{Supp}(P)} q(x) \right) \geq -\log 1 = 0 \end{aligned}$$

Kl divergence

Interpretation of KL divergence in terms of source coding

$$D(P\|Q) = \sum_x p(x)\log\frac{p(x)}{q(x)} = \sum_x p(x)\log\frac{1}{q(x)} - \sum_x p(x)\log\frac{1}{p(x)}$$

Kl divergence

Interpretation of KL divergence in terms of source coding

$$D(P\|Q) = \sum_x p(x) \log \frac{p(x)}{q(x)} = \sum_x p(x) \log \frac{1}{q(x)} - \sum_x p(x) \log \frac{1}{p(x)}$$

→ This can be interpreted as the number of extra bits we use (on average) if we designed a code according to the distribution P, but used it to communicate outcomes of a random variable X distributed according to Q

Kl divergence

Interpretation of KL divergence in terms of source coding

$$D(P\|Q) = \sum_x p(x) \log \frac{p(x)}{q(x)} = \sum_x p(x) \log \frac{1}{q(x)} - \sum_x p(x) \log \frac{1}{p(x)}$$

- $\rightarrow$ This can be interpreted as the number of extra bits we use (on average) if we designed a code according to the distribution P, but used it to communicate outcomes of a random variable X distributed according to Q
- $\rightarrow$ The first term in the RHS, which corresponds to the average number of bits used by the "wrong" encoding, is also referred to as *cross entropy*

Code

Nonsingular code - if every element of $\mathcal{X}$ maps into a different string of the alphabet set i.e. $x \neq y \Rightarrow c(x) \neq c(y)$

Code

Nonsingular code - if every element of $\mathcal{X}$ maps into a different string of the alphabet set i.e. $x \neq y \Rightarrow c(x) \neq c(y)$

Extension of a code The extension C^* of a code C is the mapping from the finite strings of $\mathcal{X}$ to finite strings of the alphabet set i.e.

$$C \quad \underbrace{(x_1 x_2 \ldots x_n)}_{\text{discrete memoryless source}} = \underbrace{C(x_1)C(x_2)\ldots C(x_n)}_{\text{concatenation}}$$

Code

Nonsingular code - if every element of $\mathcal{X}$ maps into a different string of the alphabet set i.e. $x \neq y \Rightarrow c(x) \neq c(y)$

Extension of a code The extension C^* of a code C is the mapping from the finite strings of $\mathcal{X}$ to finite strings of the alphabet set i.e.

$$C \quad \underbrace{(x_1 x_2 \ldots x_n)}_{\text{discrete memoryless source}} \quad = \underbrace{C(x_1)C(x_2)\ldots C(x_n)}_{\text{concatenation}}$$

Uniquely decodable code A code is uniquely decodable if its extension is nonsingular i.e. any encoded string is a uniquely decodable code has only one possible source string

Code

Note

- Prefix free code is uniquely decodable

Code

Note

- ▷ Prefix free code is uniquely decodable
- ▷ Using the Shannon's idea the expected codeword length is $H(X) + 1$

Code

Note

- Prefix free code is uniquely decodable
- Using the Shannon's idea the expected codeword length is $H(X) + 1$

Question Can we construct a uniquely decobale code with expected codeword length $H(X)$? - optimal codeword length (Huffman code)

Channel capacity

$$\underbrace{(x_1, x_2, \ldots, x_n)}_{\text{input}} \rightarrow \underbrace{(y_1, y_2, \ldots, y_n)}_{\text{output}}$$

Channel capacity

$$\underbrace{(x_1, x_2, \ldots, x_n)}_{\text{input}} \to \underbrace{(y_1, y_2, \ldots, y_n)}_{\text{output}}$$

memoryless: y_j depends only on x_j

Channel capacity

$$\underbrace{(x_1, x_2, \ldots, x_n)}_{\text{input}} \to \underbrace{(y_1, y_2, \ldots, y_n)}_{\text{output}}$$

memoryless: y_j depends only on x_j

$$X \xrightarrow[Channel]{p(y_j|x_k)} Y$$

Channel capacity

$$\underbrace{(x_1, x_2, \ldots, x_n)}_{\text{input}} \to \underbrace{(y_1, y_2, \ldots, y_n)}_{\text{output}}$$

memoryless: y_j depends only on x_j

$$X \xrightarrow[Channel]{p(y_j|x_k)} Y$$

$$\begin{bmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_K) \end{bmatrix} \to \begin{bmatrix} p(y_1) \\ p(y_2) \\ \vdots \\ p(y_J) \end{bmatrix}, \ p(y_j) = \sum_{k=1}^{K} p(y_j|x_k)p(x_k)$$

Channel capacity

$$\begin{bmatrix} p(y_1) \\ p(y_2) \\ \vdots \\ p(y_J) \end{bmatrix} = \underbrace{\begin{bmatrix} p(y_1|x_1) & p(y_1|x_2) & \dots & p(y_1|x_K) \\ \vdots & \vdots & \dots & \vdots \\ p(y_J|x_1) & p(y_J|x_2) & \dots & p(y_J|x_K) \end{bmatrix}_{J\times K}}_{\text{Channel matrix}} \begin{bmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_K) \end{bmatrix}$$

Channel capacity

$$\begin{bmatrix} p(y_1) \\ p(y_2) \\ \vdots \\ p(y_J) \end{bmatrix} = \underbrace{\begin{bmatrix} p(y_1|x_1) & p(y_1|x_2) & \dots & p(y_1|x_K) \\ \vdots & \vdots & \dots & \vdots \\ p(y_J|x_1) & p(y_J|x_2) & \dots & p(y_J|x_K) \end{bmatrix}_{J\times K}}_{\text{Channel matrix}} \begin{bmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_K) \end{bmatrix}$$

Observation

1. the channel matrix is nonnegative

Channel capacity

$$\begin{bmatrix} p(y_1) \\ p(y_2) \\ \vdots \\ p(y_J) \end{bmatrix} = \underbrace{\begin{bmatrix} p(y_1|x_1) & p(y_1|x_2) & \dots & p(y_1|x_K) \\ \vdots & \vdots & \dots & \vdots \\ p(y_J|x_1) & p(y_J|x_2) & \dots & p(y_J|x_K) \end{bmatrix}_{J\times K}}_{\text{Channel matrix}} \begin{bmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_K) \end{bmatrix}$$

Observation

1. the channel matrix is nonnegative
2. sum of entries in each column is 1

Channel capacity

$$\begin{bmatrix} p(y_1) \\ p(y_2) \\ \vdots \\ p(y_J) \end{bmatrix} = \underbrace{\begin{bmatrix} p(y_1|x_1) & p(y_1|x_2) & \dots & p(y_1|x_K) \\ \vdots & \vdots & \dots & \vdots \\ p(y_J|x_1) & p(y_J|x_2) & \dots & p(y_J|x_K) \end{bmatrix}_{J\times K}}_{\text{Channel matrix}} \begin{bmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_K) \end{bmatrix}$$

Observation

1. the channel matrix is nonnegative
2. sum of entries in each column is 1

Channel capacity

$$\begin{bmatrix} p(y_1) \\ p(y_2) \\ \vdots \\ p(y_J) \end{bmatrix} = \underbrace{\begin{bmatrix} p(y_1|x_1) & p(y_1|x_2) & \dots & p(y_1|x_K) \\ \vdots & \vdots & \dots & \vdots \\ p(y_J|x_1) & p(y_J|x_2) & \dots & p(y_J|x_K) \end{bmatrix}_{J\times K}}_{\text{Channel matrix}} \begin{bmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_K) \end{bmatrix}$$

Observation

1. the channel matrix is nonnegative
2. sum of entries in each column is 1

$p(y_j|x_k)$ are called transition probabilities

Channel capacity

Memoryless channel if each output letter in the output sequence depends only on the corresponding in put i.e.

$$p_N(\overline{y}|\overline{x}) = p_N((y_1 \dots y_N)|(x_1 \dots x_N)) = \prod_{n=1}^{N} p(y_n|x_n)$$

for all $n, N, \overline{x}, \overline{y}$

Channel capacity

Memoryless channel if each output letter in the output sequence depends only on the corresponding in put i.e.

$$p_N(\overline{y}|\overline{x}) = p_N((y_1 \dots y_N)|(x_1 \dots x_N)) = \prod_{n=1}^{N} p(y_n|x_n)$$

for all $n, N, \overline{x}, \overline{y}$

Example Binary symmetric channel

Channel capacity

Alternative interpretation of mutual information Suppose

$$I(x; y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} = \log \frac{p(x, y)}{p(x)p(y)} = I(y; x)$$

Channel capacity

Alternative interpretation of mutual information Suppose

$$I(x;y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} = \log \frac{p(x,y)}{p(x)p(y)} = I(y;x)$$

Set

$$I(X;Y) = \sum_{x,y} p(x,y) I(x;y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)},$$

the 'average' mutual information

Channel capacity

Alternative interpretation of mutual information Suppose

$$I(x;y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} = \log \frac{p(x,y)}{p(x)p(y)} = I(y;x)$$

Set

$$I(X;Y) = \sum_{x,y} p(x,y) I(x;y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)},$$

the 'average' mutual information

Homework $I(X;Y) = H(X) - H(X|Y)$!!

Channel capacity

Alternative interpretation of mutual information Suppose

$$I(x;y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} = \log \frac{p(x,y)}{p(x)p(y)} = I(y;x)$$

Set

$$I(X;Y) = \sum_{x,y} p(x,y) I(x;y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)},$$

the 'average' mutual information

Homework $I(X;Y) = H(X) - H(X|Y)$!!
Question Does it have any connection with the KL-divergence?

Channel capacity

The largest 'average' mutual information that can be obtained over the channel

$$C = \max_{p(X)} I(X;Y)$$

i.e.

$$\max I(X;Y) \text{ wrt } \sum_{k=1}^{K} p_k = 1, p_k \geq 0$$

Channel capacity

The largest 'average' mutual information that can be obtained over the channel

$$C = \max_{p(X)} I(X;Y)$$

i.e.

$$\max I(X;Y) \text{ wrt } \sum_{k=1}^{K} p_k = 1, p_k \geq 0$$

Question Does it exist?

Channel capacity

Theorem (DMC) Let $\overline{X}^N, \overline{Y}^N$ denote the random variables corresponding to the sequences of N-length input and output sequences respectively:

$$\overline{X}^N = (X_1, \ldots, X_N),\ \overline{Y}^N = (Y_1, \ldots, Y_N),$$

where X_i, Y_i are iid. Then

$$I(\overline{X}^N; \overline{Y}^N) \leq \sum_{n=1}^{N} I(X_n; Y_n)$$

and

$$I(\overline{X}^N; \overline{Y}^N) \leq NC.$$

Channel capacity

Theorem (DMC) Let $\overline{X}^N, \overline{Y}^N$ denote the random variables corresponding to the sequences of N-length input and output sequences respectively:

$$\overline{X}^N = (X_1, \ldots, X_N),\ \overline{Y}^N = (Y_1, \ldots, Y_N),$$

where X_i, Y_i are iid. Then

$$I(\overline{X}^N; \overline{Y}^N) \leq \sum_{n=1}^{N} I(X_n; Y_n)$$

and

$$I(\overline{X}^N; \overline{Y}^N) \leq NC.$$

Question What is the conclusion of this theorem?

Review

$\rightarrow$ Shannon demonstrated that with a proper encoding of the information, the errors induced by a noisy channel or storage medium can be reduced to any desired level as long as the information rate is less than the capacity of the channel

Review

→ Shannon demonstrated that with a proper encoding of the information, the errors induced by a noisy channel or storage medium can be reduced to any desired level as long as the information rate is less than the capacity of the channel

→ The source encoder transform the source output into a string of bits, called the information sequence

- △ The number of bits per unit time required to represent the source output is minimized
- △ The source output can be perfectly reconstructed from the information sequence $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$

Review

- → Shannon demonstrated that with a proper encoding of the information, the errors induced by a noisy channel or storage medium can be reduced to any desired level as long as the information rate is less than the capacity of the channel
- → The source encoder transform the source output into a string of bits, called the information sequence
  - △ The number of bits per unit time required to represent the source output is minimized
  - △ The source output can be perfectly reconstructed from the information sequence $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$
- → The channel encoder transforms the information sequence $\mathbf{u}$ into a string of bits $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ called a *codeword*

Review

→ The modulator transforms each output symbol of the channel encoder into a waveform of duration, say T seconds which is suitable for transmission

- △ This waveform enters the channel and get corrupted by noise

Review

- → The modulator transforms each output symbol of the channel encoder into a waveform of duration, say T seconds which is suitable for transmission
  - △ This waveform enters the channel and get corrupted by noise
  - △ Examples of transmission channels - telephone lines, mobile cellular technology, high-frequency (HF) radio, microwave and satellite links, optical fiber cables

Review

- → The modulator transforms each output symbol of the channel encoder into a waveform of duration, say T seconds which is suitable for transmission
  - △ This waveform enters the channel and get corrupted by noise
  - △ Examples of transmission channels - telephone lines, mobile cellular technology, high-frequency (HF) radio, microwave and satellite links, optical fiber cables
  - △ Examples of storage media - semiconductor memories, magnetic tapes, compact discs

Review

- → The modulator transforms each output symbol of the channel encoder into a waveform of duration, say T seconds which is suitable for transmission
  - △ This waveform enters the channel and get corrupted by noise
  - △ Examples of transmission channels - telephone lines, mobile cellular technology, high-frequency (HF) radio, microwave and satellite links, optical fiber cables
  - △ Examples of storage media - semiconductor memories, magnetic tapes, compact discs
  - △ Examples of noise - On a telephone line, disturbances may come from: switching impulse noise, crosstalk from other lines. On compact discs: dust particles

Review

- → The modulator transforms each output symbol of the channel encoder into a waveform of duration, say T seconds which is suitable for transmission
  - △ This waveform enters the channel and get corrupted by noise
  - △ Examples of transmission channels - telephone lines, mobile cellular technology, high-frequency (HF) radio, microwave and satellite links, optical fiber cables
  - △ Examples of storage media - semiconductor memories, magnetic tapes, compact discs
  - △ Examples of noise - On a telephone line, disturbances may come from: switching impulse noise, crosstalk from other lines. On compact discs: dust particles
- → The demodulator processes each received waveform of duration T and produces either a discrete or continuous output

Review

- → The modulator transforms each output symbol of the channel encoder into a waveform of duration, say T seconds which is suitable for transmission
  - △ This waveform enters the channel and get corrupted by noise
  - △ Examples of transmission channels - telephone lines, mobile cellular technology, high-frequency (HF) radio, microwave and satellite links, optical fiber cables
  - △ Examples of storage media - semiconductor memories, magnetic tapes, compact discs
  - △ Examples of noise - On a telephone line, disturbances may come from: switching impulse noise, crosstalk from other lines. On compact discs: dust particles
- → The demodulator processes each received waveform of duration T and produces either a discrete or continuous output
- → The sequence of demodulator outputs corresponding to the encoded sequence $\mathbf{v}$, called the received sequence $\mathbf{r}$

Review

→ The channel decoder transforms the received sequence $\mathbf{r}$ into a binary sequence $\widehat{\mathbf{u}}$, called the estimated information sequence

Review

→ The channel decoder transforms the received sequence $\mathbf{r}$ into a binary sequence $\widehat{\mathbf{u}}$, called the estimated information sequence

- △ The decoding strategy is based on the rules of channel encoding and the noise characteristics of the channel or the storage medium
- △ Ideally, $\widehat{\mathbf{u}} = \mathbf{u}$, although noise may cause decoding errors

Review

→ The channel decoder transforms the received sequence $\mathbf{r}$ into a binary sequence $\widehat{\mathbf{u}}$, called the estimated information sequence

- △ The decoding strategy is based on the rules of channel encoding and the noise characteristics of the channel or the storage medium
- △ Ideally, $\widehat{\mathbf{u}} = \mathbf{u}$, although noise may cause decoding errors

The big picture

$$\mathbf{u} \to \mathbf{v} \to \mathbf{r} \to \widehat{\mathbf{u}}$$

Review

- $\rightarrow$ The channel decoder transforms the received sequence $\mathbf{r}$ into a binary sequence $\widehat{\mathbf{u}}$, called the estimated information sequence
  - $\triangle$ The decoding strategy is based on the rules of channel encoding and the noise characteristics of the channel or the storage medium
  - $\triangle$ Ideally, $\widehat{\mathbf{u}} = \mathbf{u}$, although noise may cause decoding errors

The big picture

$$\mathbf{u} \rightarrow \mathbf{v} \rightarrow \mathbf{r} \rightarrow \widehat{\mathbf{u}}$$

Problem Design and implementation of encoder/decoder pair such that - information can be transmitted in noisy environment, and the information can be reliably reproduced at the output of the channel decoder

Codes

Observation

→ The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)

Codes

Observation

- → The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)
- → There are 2^k different possible messages

Codes

Observation

- $\rightarrow$ The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)
- $\rightarrow$ There are 2^k different possible messages
- $\rightarrow$ The encoder transform each message $\mathbf{u}$ into an n-tuple $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$, called a codeword (sometimes $\mathbf{v}$ is used to denote an n-symbol block rather than the entire encoded sequence)

Codes

Observation

$\rightarrow$ The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)

$\rightarrow$ There are 2^k different possible messages

$\rightarrow$ The encoder transform each message $\mathbf{u}$ into an n-tuple $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$, called a codeword (sometimes $\mathbf{v}$ is used to denote an n-symbol block rather than the entire encoded sequence)

$\rightarrow$ Therefore, corresponding to 2^k different possible messages, there are 2^k different possible codewords at the endoder output

Codes

Observation

- $\rightarrow$ The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)
- $\rightarrow$ There are 2^k different possible messages
- $\rightarrow$ The encoder transform each message $\mathbf{u}$ into an n-tuple $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$, called a codeword (sometimes $\mathbf{v}$ is used to denote an n-symbol block rather than the entire encoded sequence)
- $\rightarrow$ Therefore, corresponding to 2^k different possible messages, there are 2^k different possible codewords at the endoder output
- $\rightarrow$ This set of 2^k codewords of length n is called an (n, k) block code

Codes

Observation

- → The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)
- → There are 2^k different possible messages
- → The encoder transform each message $\mathbf{u}$ into an n-tuple $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$, called a codeword (sometimes $\mathbf{v}$ is used to denote an n-symbol block rather than the entire encoded sequence)
- → Therefore, corresponding to 2^k different possible messages, there are 2^k different possible codewords at the endoder output
- → This set of 2^k codewords of length n is called an (n, k) block code
- → The ratio $R = k/n$ is called the *code rate*, and it can be interpreted as the number of information bits entering the encoder per transmitted symbol

Codes

Observation

- → The k-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$, called a message (sometimes $\mathbf{u}$ is used to denote a k-bit message rather than the entire information sequence)
- → There are 2^k different possible messages
- → The encoder transform each message $\mathbf{u}$ into an n-tuple $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$, called a codeword (sometimes $\mathbf{v}$ is used to denote an n-symbol block rather than the entire encoded sequence)
- → Therefore, corresponding to 2^k different possible messages, there are 2^k different possible codewords at the endoder output
- → This set of 2^k codewords of length n is called an (n, k) block code
- → The ratio $R = k/n$ is called the *code rate*, and it can be interpreted as the number of information bits entering the encoder per transmitted symbol
- → Each message is encoded independently, so the encoder is memoryless and can be implemented with a *combinatorial logic circuit*

Linear block codes

Definition A block code of length n and 2^k codewords is called a linear (n, k)-code if and only if its 2^k codewords form a k-dimensional subspace of the vector space of all n-tuples over the field $GF(2)$, the Galois Field of order 2

Linear block codes

Definition A block code of length n and 2^k codewords is called a linear (n, k)-code if and only if its 2^k codewords form a k-dimensional subspace of the vector space of all n-tuples over the field $GF(2)$, the Galois Field of order 2

Conclusion

- △ A binary block code is linear if and only if the modulo-2 sum of two codewords is also a codeword
- △ Since (n, k) linear block code C is a k-dimension subspace of V_n, the vector space of all binary n-tuples, it is possible to find k linearly independent codewords $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}$ in C such that any codeword $\mathbf{v}$ in C can be written as

$$v = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots u_{k-1}\mathbf{g}_{k-1}$$

where $u_i \in \{0, 1\}, 0 \leq i \leq k-1$

Linear block codes

Write

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix}_{k \times n}$$

where

$$\mathbf{g}_i = (g_{i0}, g_{i1}, \ldots, g_{i,n-1}), 0 \leq i \leq k-1.$$

Linear block codes

Write

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix}_{k \times n}$$

where

$$\mathbf{g}_i = (g_{i0}, g_{i1}, \ldots, g_{i,n-1}), 0 \leq i \leq k-1.$$

Then

$$\begin{aligned} \mathbf{v} &= \mathbf{u} \cdot \mathbf{G} \\ &= u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots + \ldots, u_{k-1}\mathbf{g}_{k-1} \end{aligned}$$

Linear block codes

Since $\mathbf{G}$ generate the (n, k) linear code C, the matrix $\mathbf{G}$ is called a generator matrix for C.

Linear block codes

Since $\mathbf{G}$ generate the (n, k) linear code C, the matrix $\mathbf{G}$ is called a generator matrix for C.

Example

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

generates a $(7, 4)$ linear code

Linear block codes

Since $\mathbf{G}$ generate the (n, k) linear code C, the matrix $\mathbf{G}$ is called a generator matrix for C.

Example

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

generates a $(7, 4)$ linear code

Question Verify that $\mathbf{v} = (0\,0\,0\,1\,1\,0\,1)$ is a codeword for the above generator matrix

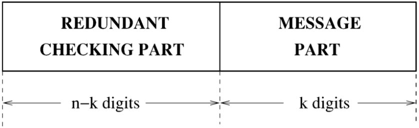
Linear block codes

Systematic format of a codeword A codeword is divided into two parts - the message part and the redundant checking part

Linear block codes

Systematic format of a codeword A codeword is divided into two parts - the message part and the redundant checking part

The message part consists of k unaltered information digits, and the redundant checking part consists of $n - k$ *parity-check* digits



A linear block with this structure is referred to as *linear systematic block code*

Linear block code

Thus a linear systematic (n, k) code is completely described by a $k \times n$ matrix $\mathbf{G}$ of the following form

$$\mathbf{G} = \begin{bmatrix} \mathbf{P} & I_k \end{bmatrix}, \ \mathbf{P} = [p_{ij}] \in \{0, 1\}^{k \times (n-k)}$$

Linear block code

Thus a linear systematic (n, k) code is completely described by a $k \times n$ matrix $\mathbf{G}$ of the following form

$$\mathbf{G} = \begin{bmatrix} \mathbf{P} & I_k \end{bmatrix}, \ \mathbf{P} = [p_{ij}] \in \{0, 1\}^{k \times (n-k)}$$

Let $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ be the message to be encoded. Then the corresponding codeword is

$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G}$$

which gives two equations

$$\begin{aligned} v_{n-k+i} &= u_i, \ 0 \leq i \leq k-1 && (5) \\ v_j &= u_0 p_{0j} + u_1 p_{1j} + \ldots + u_{k-1} p_{k-1,j}, \ 0 \leq j \leq n-k-1. && (6) \end{aligned}$$

Linear block code

Thus a linear systematic (n, k) code is completely described by a $k \times n$ matrix $\mathbf{G}$ of the following form

$$\mathbf{G} = \begin{bmatrix} \mathbf{P} & I_k \end{bmatrix}, \; \mathbf{P} = [p_{ij}] \in \{0, 1\}^{k \times (n-k)}$$

Let $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ be the message to be encoded. Then the corresponding codeword is

$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G}$$

which gives two equations

$$\begin{aligned} v_{n-k+i} &= u_i, \; 0 \leq i \leq k-1 && (5) \\ v_j &= u_0 p_{0j} + u_1 p_{1j} + \ldots + u_{k-1} p_{k-1,j}, \; 0 \leq j \leq n-k-1. && (6) \end{aligned}$$

The $(n-k)$ equations given by equation (6) are called parity-check equations.

Linear block code

Parity-check matrix

- △ The generator matrix $\mathbf{G}$ has k linearly independent rows from $\{0,1\}^n$

Linear block code

Parity-check matrix

- △ The generator matrix $\mathbf{G}$ has k linearly independent rows from $\{0, 1\}^n$
- △ Then there can be $n - k$ linearly independent rows from $\{0, 1\}^n$, say $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-k}$ such that any vector in the row space of $\mathbf{G}$ is orthogonal to $\mathbf{h}_j$, $0 \le j \le n - 1$

Linear block code

Parity-check matrix

- $\triangle$ The generator matrix $\mathbf{G}$ has k linearly independent rows from $\{0,1\}^n$
- $\triangle$ Then there can be $n-k$ linearly independent rows from $\{0,1\}^n$, say $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-k}$ such that any vector in the row space of $\mathbf{G}$ is orthogonal to $\mathbf{h}_j$, $0 \leq j \leq n-1$
- $\triangle$ Define

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{n-k} \end{bmatrix}$$

Linear block code

Parity-check matrix

- The generator matrix $\mathbf{G}$ has k linearly independent rows from $\{0,1\}^n$
- Then there can be $n-k$ linearly independent rows from $\{0,1\}^n$, say $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-k}$ such that any vector in the row space of $\mathbf{G}$ is orthogonal to $\mathbf{h}_j$, $0 \leq j \leq n-1$
- Define

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{n-k} \end{bmatrix}$$

Then an n-tuple $\mathbf{v}$ is a codeword in the code C generated by $\mathbf{G}$ if and only if $\mathbf{v} \cdot \mathbf{H}^T = \mathbf{0}$

Linear block code

Then the code C is just the null-space of $\mathbf{H}$, which is called a parity-check matrix of the code.

Linear block code

Then the code C is just the null-space of $\mathbf{H}$, which is called a parity-check matrix of the code.

Note The rows of $\mathbf{H}$ also generate a $(n, n-k)$ linear code C_d, which is called the dual code of C.

Linear block code

Then the code C is just the null-space of $\mathbf{H}$, which is called a parity-check matrix of the code.

Note The rows of $\mathbf{H}$ also generate a $(n, n-k)$ linear code C_d, which is called the dual code of C.

Problem The code C_d is the null space $\mathbf{G}$.

Linear block code

Then the code C is just the null-space of $\mathbf{H}$, which is called a parity-check matrix of the code.

Note The rows of $\mathbf{H}$ also generate a $(n, n-k)$ linear code C_d, which is called the dual code of C.

Problem The code C_d is the null space $\mathbf{G}$.

If the generator matrix of an (n, k) linear code is in the systematic form then the parity-check matrix can be in the following form:

$$\mathbf{H} = \begin{bmatrix} I_{n-k} & \mathbf{P}^T \end{bmatrix}.$$

Linear block code

Then the code C is just the null-space of $\mathbf{H}$, which is called a parity-check matrix of the code.

Note The rows of $\mathbf{H}$ also generate a $(n, n-k)$ linear code C_d, which is called the dual code of C.

Problem The code C_d is the null space $\mathbf{G}$.

If the generator matrix of an (n, k) linear code is in the systematic form then the parity-check matrix can be in the following form:

$$\mathbf{H} = \begin{bmatrix} I_{n-k} & \mathbf{P}^T \end{bmatrix}.$$

Then see that

$$\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}.$$

Linear block code

Syndrome decoding Consider an (n, k) linear code corresponding to generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$. Let $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})$ be the received vector at the output of a noisy channel corresponding to a codeword $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$.

Linear block code

Syndrome decoding Consider an (n, k) linear code corresponding to generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$. Let $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})$ be the received vector at the output of a noisy channel corresponding to a codeword $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$.

Then

$$\mathbf{r} = \mathbf{v} + \mathbf{e} \Rightarrow \mathbf{e} = \mathbf{r} + \mathbf{v} = (e_0, e_1, \ldots, e_{n-1})$$

is the *error vector*, where $e_i = 1$ for $r_i \neq v_i$, and $e_i = 0$ for $r_i = v_i$.

Linear block code

Syndrome decoding Consider an (n, k) linear code corresponding to generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$. Let $\mathbf{r} = (r_0, r_1, \dots, r_{n-1})$ be the received vector at the output of a noisy channel corresponding to a codeword $\mathbf{v} = (v_0, v_1, \dots, v_{n-1})$.

Then

$$\mathbf{r} = \mathbf{v} + \mathbf{e} \Rightarrow \mathbf{e} = \mathbf{r} + \mathbf{v} = (e_0, e_1, \dots, e_{n-1})$$

is the *error vector*, where $e_i = 1$ for $r_i \neq v_i$, and $e_i = 0$ for $r_i = v_i$. Thus the 1's in $\mathbf{e}$ are the transmission errors caused by the channel noise.

Linear block code

Syndrome decoding Consider an (n, k) linear code corresponding to generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$. Let $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})$ be the received vector at the output of a noisy channel corresponding to a codeword $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$.

Then

$$\mathbf{r} = \mathbf{v} + \mathbf{e} \Rightarrow \mathbf{e} = \mathbf{r} + \mathbf{v} = (e_0, e_1, \ldots, e_{n-1})$$

is the *error vector*, where $e_i = 1$ for $r_i \neq v_i$, and $e_i = 0$ for $r_i = v_i$. Thus the 1's in $\mathbf{e}$ are the transmission errors caused by the channel noise.

Note The receiver does not know both $\mathbf{v}$ and $\mathbf{e}$

Linear block code

Syndrome decoding Consider an (n, k) linear code corresponding to generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$. Let $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})$ be the received vector at the output of a noisy channel corresponding to a codeword $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$.

Then

$$\mathbf{r} = \mathbf{v} + \mathbf{e} \Rightarrow \mathbf{e} = \mathbf{r} + \mathbf{v} = (e_0, e_1, \ldots, e_{n-1})$$

is the *error vector*, where $e_i = 1$ for $r_i \neq v_i$, and $e_i = 0$ for $r_i = v_i$. Thus the 1's in $\mathbf{e}$ are the transmission errors caused by the channel noise.

Note The receiver does not know both $\mathbf{v}$ and $\mathbf{e}$

Question How does the receiver detect, locate and correct the error?

Linear block code

On receiving $\mathbf{r}$, the decoder must first determine whether $\mathbf{r}$ contains transmission errors. Thus the decoder computes

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (s_0, s_1, \ldots, s_{n-k-1})$$

which is called the *syndrome* of $\mathbf{r}$.

Linear block code

On receiving $\mathbf{r}$, the decoder must first determine whether $\mathbf{r}$ contains transmission errors. Thus the decoder computes

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (s_0, s_1, \ldots, s_{n-k-1})$$

which is called the *syndrome* of $\mathbf{r}$.

Then $\mathbf{s} = \mathbf{0}$ if and only if $\mathbf{r}$ is a codeword, and $\mathbf{s} \neq \mathbf{0}$ if and only if $\mathbf{r}$ is not a codeword. Thus when $\mathbf{s} = 0$, $\mathbf{r}$ is a codeword, and the receiver accepts $\mathbf{r}$ as the transmitted codeword.

Linear block code

On receiving $\mathbf{r}$, the decoder must first determine whether $\mathbf{r}$ contains transmission errors. Thus the decoder computes

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (s_0, s_1, \ldots, s_{n-k-1})$$

which is called the *syndrome* of $\mathbf{r}$.

Then $\mathbf{s} = \mathbf{0}$ if and only if $\mathbf{r}$ is a codeword, and $\mathbf{s} \neq \mathbf{0}$ if and only if $\mathbf{r}$ is not a codeword. Thus when $\mathbf{s} = 0$, $\mathbf{r}$ is a codeword, and the receiver accepts $\mathbf{r}$ as the transmitted codeword.

Caution It is possible that the errors in certain error vectors are not detectable. For instance, if $\mathbf{e}$ is identical to a nonzero codeword. This kind of error patterns are called *undetectable* error patterns. There are $2^k - 1$ undetectable errors (Homework)

Linear block code

However, note that

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (\mathbf{v} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{v} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T$$

Linear block code

However, note that

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (\mathbf{v} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{v} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T$$

Thus the syndrome bits give information about error bits.

Linear block code

However, note that

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (\mathbf{v} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{v} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T$$

Thus the syndrome bits give information about error bits.

Question Can we solve the linear system and obtain $\mathbf{e}$?

Linear block code

However, note that

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (\mathbf{v} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{v} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T$$

Thus the syndrome bits give information about error bits.

Question Can we solve the linear system and obtain $\mathbf{e}$?

Note that there are $n - k$ linear equations and the system does not have a unique solution but can have 2^k solutions!!

Linear block code

Minimum distance of a block code Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be an n-tuple. Then the *Hamming weight* of $\mathbf{v}$, denotes as $w(\mathbf{v})$ is the number of nonzero entries of $\mathbf{v}$.

Linear block code

Minimum distance of a block code Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be an n-tuple. Then the *Hamming weight* of $\mathbf{v}$, denotes as $w(\mathbf{v})$ is the number of nonzero entries of $\mathbf{v}$.

The *Hamming distance* between two vectors $\mathbf{v}$ and $\mathbf{w}$, denotes as $d_h(\mathbf{v}, \mathbf{w})$ is the number of places where $\mathbf{v}$ and $\mathbf{w}$ differ.

Linear block code

Minimum distance of a block code Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be an n-tuple. Then the *Hamming weight* of $\mathbf{v}$, denotes as $w(\mathbf{v})$ is the number of nonzero entries of $\mathbf{v}$.

The *Hamming distance* between two vectors $\mathbf{v}$ and $\mathbf{w}$, denotes as $d_h(\mathbf{v}, \mathbf{w})$ is the number of places where $\mathbf{v}$ and $\mathbf{w}$ differ.

Question Show that Hamming distance is a metric.

Linear block code

Minimum distance of a block code Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be an n-tuple. Then the *Hamming weight* of $\mathbf{v}$, denotes as $w(\mathbf{v})$ is the number of nonzero entries of $\mathbf{v}$.

The *Hamming distance* between two vectors $\mathbf{v}$ and $\mathbf{w}$, denotes as $d_h(\mathbf{v}, \mathbf{w})$ is the number of places where $\mathbf{v}$ and $\mathbf{w}$ differ.

Question Show that Hamming distance is a metric.

The minimum distance of a code C is defined by

$$d_{\min} = \min\{d_h(\mathbf{v}, \mathbf{w}) : \mathbf{v}, \mathbf{w} \in C, \mathbf{v} \neq \mathbf{w}\}$$

Linear block code

Note that

$$\begin{aligned} d_{\min} &= \min\{w(\mathbf{v}+\mathbf{w}) : \mathbf{v}, \mathbf{w} \in C, \mathbf{v} \neq \mathbf{w}\} \\ &= \min\{w(\mathbf{x}) : \mathbf{x} \in C, \mathbf{x} \neq 0\} \end{aligned}$$

Thus minimum distance of a linear code is the minimum weight of the code.

Linear block code

Theorem Let C be an (n, k) linear code with parity-check matrix $\mathbf{H}$. Then for each codeword of Hamming weight l, there exists l columns of $\mathbf{H}$ such that the sum of these l columns is equal to the zero vector. Conversely, if there exist l columns of $\mathbf{H}$ whose sum is the zero vector then there exists a codeword of Hamming weight l in C.

Linear block code

Theorem Let C be an (n, k) linear code with parity-check matrix $\mathbf{H}$. Then for each codeword of Hamming weight l, there exists l columns of $\mathbf{H}$ such that the sum of these l columns is equal to the zero vector. Conversely, if there exist l columns of $\mathbf{H}$ whose sum is the zero vector then there exists a codeword of Hamming weight l in C.

Corollary Let C be a linear block code with parity-check matrix $\mathbf{H}$. Then

(a) If no $d - 1$ or fewer columns of $\mathbf{H}$ add to $\mathbf{0}$, the code has minimum weight at least d

Linear block code

Theorem Let C be an (n, k) linear code with parity-check matrix $\mathbf{H}$. Then for each codeword of Hamming weight l, there exists l columns of $\mathbf{H}$ such that the sum of these l columns is equal to the zero vector. Conversely, if there exist l columns of $\mathbf{H}$ whose sum is the zero vector then there exists a codeword of Hamming weight l in C.

Corollary Let C be a linear block code with parity-check matrix $\mathbf{H}$. Then

(a) If no $d - 1$ or fewer columns of $\mathbf{H}$ add to $\mathbf{0}$, the code has minimum weight at least d

(b) The minimum distance of C is equal to the smallest number of columns of $\mathbf{H}$ that sum to $\mathbf{0}$.

Linear block code

Error detection and error correction Suppose a codeword $\mathbf{v}$ is transmitted over a noisy channel. Then a block code with minimum distance $d_{\min}$ is capable of detecting all the error patterns of $d_{\min} - 1$ or fewer errors:

$\rightarrow$ If there are l errors in the corresponding received vector $\mathbf{r}$, then $d(\mathbf{v}, \mathbf{r}) = l$

Linear block code

Error detection and error correction Suppose a codeword $\mathbf{v}$ is transmitted over a noisy channel. Then a block code with minimum distance $d_{\min}$ is capable of detecting all the error patterns of $d_{\min} - 1$ or fewer errors:

- $\rightarrow$ If there are l errors in the corresponding received vector $\mathbf{r}$, then $d(\mathbf{v}, \mathbf{r}) = l$
- $\rightarrow$ If the minimum distance of a block code C is $d_{\min}$, then any two distinct codewords in C differ at least in $d_{\min}$ places

Linear block code

Error detection and error correction Suppose a codeword $\mathbf{v}$ is transmitted over a noisy channel. Then a block code with minimum distance $d_{\min}$ is capable of detecting all the error patterns of $d_{\min} - 1$ or fewer errors:

- $\rightarrow$ If there are l errors in the corresponding received vector $\mathbf{r}$, then $d(\mathbf{v}, \mathbf{r}) = l$
- $\rightarrow$ If the minimum distance of a block code C is $d_{\min}$, then any two distinct codewords in C differ at least in $d_{\min}$ places
- $\rightarrow$ Then for this code, no error pattern of $d_{\min} - 1$ or fewer errors can change one codeword into another, hence any error pattern of $d_{\min} - 1$ or few errors will result in a received vector $\mathbf{r}$ that is not a codeword in C

Linear block code

Error detection and error correction Suppose a codeword $\mathbf{v}$ is transmitted over a noisy channel. Then a block code with minimum distance $d_{\min}$ is capable of detecting all the error patterns of $d_{\min} - 1$ or fewer errors:

- $\rightarrow$ If there are l errors in the corresponding received vector $\mathbf{r}$, then $d(\mathbf{v}, \mathbf{r}) = l$
- $\rightarrow$ If the minimum distance of a block code C is $d_{\min}$, then any two distinct codewords in C differ at least in $d_{\min}$ places
- $\rightarrow$ Then for this code, no error pattern of $d_{\min} - 1$ or fewer errors can change one codeword into another, hence any error pattern of $d_{\min} - 1$ or few errors will result in a received vector $\mathbf{r}$ that is not a codeword in C

Question Can it detect all the error patterns of $d_{\min}$ errors?

Linear block code

Observation (n, k) linear block code can detect $2^n - 2^k$ error patterns of length n

$\rightarrow$ The number of nonzero error patterns is equal to $2^n - 1$, among which $2^k - 1$ error patterns are the $2^k - 1$ nonzero codewords.

Linear block code

Observation (n, k) linear block code can detect $2^n - 2^k$ error patterns of length n

- $\rightarrow$ The number of nonzero error patterns is equal to $2^n - 1$, among which $2^k - 1$ error patterns are the $2^k - 1$ nonzero codewords.
- $\rightarrow$ If any of these $2^k - 1$ error patterns occurs, it alters $\mathbf{v}$ into another codeword $\mathbf{w}$, and its syndrome is zero. Thus the decoder performs an incorrect decoding. Therefore there are $2^k - 1$ undetectable error patterns

Linear block code

Observation (n, k) linear block code can detect $2^n - 2^k$ error patterns of length n

- $\rightarrow$ The number of nonzero error patterns is equal to $2^n - 1$, among which $2^k - 1$ error patterns are the $2^k - 1$ nonzero codewords.
- $\rightarrow$ If any of these $2^k - 1$ error patterns occurs, it alters $\mathbf{v}$ into another codeword $\mathbf{w}$, and its syndrome is zero. Thus the decoder performs an incorrect decoding. Therefore there are $2^k - 1$ undetectable error patterns
- $\rightarrow$ Note that there are exactly $2^n - 2^k$ error patterns that are not identical to the codewords of the (n, k) block code, which are detectable

Linear block code

Observation (n, k) linear block code can detect $2^n - 2^k$ error patterns of length n

- $\rightarrow$ The number of nonzero error patterns is equal to $2^n - 1$, among which $2^k - 1$ error patterns are the $2^k - 1$ nonzero codewords.
- $\rightarrow$ If any of these $2^k - 1$ error patterns occurs, it alters $\mathbf{v}$ into another codeword $\mathbf{w}$, and its syndrome is zero. Thus the decoder performs an incorrect decoding. Therefore there are $2^k - 1$ undetectable error patterns
- $\rightarrow$ Note that there are exactly $2^n - 2^k$ error patterns that are not identical to the codewords of the (n, k) block code, which are detectable
- $\rightarrow$ For large n, $2^k - 1 \ll 2^n$ in general, hence only a small fraction of error patterns pass through the decoder without being detected

Linear block code

Maximum-Likelihood (ML) decoding

→ A decoder must determine **w** to minimize

$$P(E|\,\mathbf{r}) = P(\mathbf{w} \neq \mathbf{v}|\,\mathbf{r})$$

→ The probability of error is

$$P(E) = \sum_{\mathbf{r}} P(E|\,\mathbf{r})P(\mathbf{r})$$

Linear block code

Maximum-Likelihood (ML) decoding

→ A decoder must determine $\mathbf{w}$ to minimize

$$P(E|\,\mathbf{r}) = P(\mathbf{w} \neq \mathbf{v}|\,\mathbf{r})$$

→ The probability of error is

$$P(E) = \sum_{\mathbf{r}} P(E|\,\mathbf{r})P(\mathbf{r})$$

→ Memoryless channel: ML decoder

△ Maximize $P(\mathbf{r}|\,\mathbf{v}) = \prod_j P(r_j|\,v_j)$

Linear block code

Maximum-Likelihood (ML) decoding

→ A decoder must determine $\mathbf{w}$ to minimize

$$P(E|\,\mathbf{r}) = P(\mathbf{w} \neq \mathbf{v}|\,\mathbf{r})$$

→ The probability of error is

$$P(E) = \sum_{\mathbf{r}} P(E|\,\mathbf{r})P(\mathbf{r})$$

→ Memoryless channel: ML decoder

- △ Maximize $P(\mathbf{r}|\,\mathbf{v}) = \prod_j P(r_j|\,v_j)$
- △ Alternatively, choose $\mathbf{v}$ to maximize $\log P(\mathbf{r}|\,\mathbf{v}) = \sum_j \log P(r_j|\,v_j)$

Linear block code

Maximum-Likelihood (ML) decoding

→ A decoder must determine $\mathbf{w}$ to minimize

$$P(E|\,\mathbf{r}) = P(\mathbf{w} \neq \mathbf{v}|\,\mathbf{r})$$

→ The probability of error is

$$P(E) = \sum_{\mathbf{r}} P(E|\,\mathbf{r})P(\mathbf{r})$$

→ Memoryless channel: ML decoder

- △ Maximize $P(\mathbf{r}|\,\mathbf{v}) = \prod_j P(r_j|\,v_j)$
- △ Alternatively, choose $\mathbf{v}$ to maximize $\log P(\mathbf{r}|\,\mathbf{v}) = \sum_j \log P(r_j|\,v_j)$
- △ The ML decoder is optimal if and only if all $\mathbf{v}$ are equally likely as input vectors, otherwise $P(\mathbf{r}|\,\mathbf{v})$ must be weighted by the codeword probabilities $P(\mathbf{v})$

Linear block code

ML decoding on the BSC Suppose the noisy channel is BSC with bit-flip probability ϵ. Then

$\rightarrow$ $P(r_j | v_j) = 1 - \epsilon$ if $r_j = v_j$ and ϵ otherwise

Linear block code

ML decoding on the BSC Suppose the noisy channel is BSC with bit-flip probability ϵ. Then

- $\rightarrow$ $P(r_j | v_j) = 1 - \epsilon$ if $r_j = v_j$ and ϵ otherwise
- $\rightarrow$

$$
\begin{aligned}
\log P(\mathbf{r}|\mathbf{v}) &= \sum_j \log P(r_j | v_j) \\
&= d(\mathbf{r},\mathbf{v}) \log \epsilon + (n - d(\mathbf{r},\mathbf{v})) \log(1-\epsilon) \\
&= d(\mathbf{r},\mathbf{v}) \log \frac{\epsilon}{1-\epsilon} + n \log(1-\epsilon)
\end{aligned}
$$

Linear block code

ML decoding on the BSC Suppose the noisy channel is BSC with bit-flip probability ϵ. Then

→ $P(r_j | v_j) = 1 - \epsilon$ if $r_j = v_j$ and ϵ otherwise

→

$$\begin{aligned} \log P(\mathbf{r}|\mathbf{v}) &= \sum_j \log P(r_j|v_j) \\ &= d(\mathbf{r},\mathbf{v}) \log \epsilon + (n - d(\mathbf{r},\mathbf{v})) \log(1-\epsilon) \\ &= d(\mathbf{r},\mathbf{v}) \log \frac{\epsilon}{1-\epsilon} + n \log(1-\epsilon) \end{aligned}$$

→ $\log \frac{\epsilon}{1-\epsilon} < 0$ for $\epsilon < 0.5$, so an ML decoder for a BSC must choose $\mathbf{v}$ to minimize $d(\mathbf{r},\mathbf{v})$

Linear block code

Then for a linear block code, an ML decoder takes n received bits as input and returns the most likely k-bit message among the 2^k possible messages.

Linear block code

Then for a linear block code, an ML decoder takes n received bits as input and returns the most likely k-bit message among the 2^k possible messages. Implementing ML decoder

$\rightarrow$ Enumerate all 2^k valid codewords, each n bit in length

Linear block code

Then for a linear block code, an ML decoder takes n received bits as input and returns the most likely k-bit message among the 2^k possible messages. Implementing ML decoder

$\rightarrow$ Enumerate all 2^k valid codewords, each n bit in length

$\rightarrow$ Compare the received word $\mathbf{r}$ to each of these valid codewords and find the one with smallest Hamming distance to $\mathbf{r}$

Linear block code

Then for a linear block code, an ML decoder takes n received bits as input and returns the most likely k-bit message among the 2^k possible messages. Implementing ML decoder

→ Enumerate all 2^k valid codewords, each n bit in length

→ Compare the received word $\mathbf{r}$ to each of these valid codewords and find the one with smallest Hamming distance to $\mathbf{r}$

→ However, it has exponential time complexity. What we would like is something a lot faster. Note that this comparing to all valid codewords method does not take advantage of the linearity of the code.

Linear block code

Correction of error Let C be an (n, k) linear code with minimum distance $d_{\min}$. Then

$$2t + 1 \leq d_{\min} \leq 2t + 2$$

for some positive integer t.

Linear block code

Correction of error Let C be an (n, k) linear code with minimum distance $d_{\min}$. Then

$$2t + 1 \leq d_{\min} \leq 2t + 2$$

for some positive integer t.

Claim C is capable of correcting all the error patterns of t or fewer errors.

$\rightarrow$ Let $\mathbf{v}$ and $\mathbf{r}$ denote the transmitted codeword and the received vector respectively.

Linear block code

Correction of error Let C be an (n, k) linear code with minimum distance $d_{\min}$. Then

$$2t + 1 \leq d_{\min} \leq 2t + 2$$

for some positive integer t.

Claim C is capable of correcting all the error patterns of t or fewer errors.

→ Let $\mathbf{v}$ and $\mathbf{r}$ denote the transmitted codeword and the received vector respectively.

→ Let $\mathbf{w}$ be any other codeword of C. Then

$$d(\mathbf{v}, \mathbf{w}) \leq d(\mathbf{v}, \mathbf{r}) + d(\mathbf{w}, \mathbf{r})$$

Linear block code

Correction of error Let C be an (n, k) linear code with minimum distance d_{min}. Then

$$2t + 1 \leq d_{\text{min}} \leq 2t + 2$$

for some positive integer t.

Claim C is capable of correcting all the error patterns of t or fewer errors.

- → Let $\mathbf{v}$ and $\mathbf{r}$ denote the transmitted codeword and the received vector respectively.
- → Let $\mathbf{w}$ be any other codeword of C. Then

$$d(\mathbf{v}, \mathbf{w}) \leq d(\mathbf{v}, \mathbf{r}) + d(\mathbf{w}, \mathbf{r})$$

- → Suppose an error pattern of t' errors occurs i.e. $d(\mathbf{v}, \mathbf{r}) = t'$

Linear block code

Correction of error Let C be an (n, k) linear code with minimum distance $d_{\min}$. Then

$$2t + 1 \leq d_{\min} \leq 2t + 2$$

for some positive integer t.

Claim C is capable of correcting all the error patterns of t or fewer errors.

→ Let $\mathbf{v}$ and $\mathbf{r}$ denote the transmitted codeword and the received vector respectively.

→ Let $\mathbf{w}$ be any other codeword of C. Then

$$d(\mathbf{v}, \mathbf{w}) \leq d(\mathbf{v}, \mathbf{r}) + d(\mathbf{w}, \mathbf{r})$$

→ Suppose an error pattern of t' errors occurs i.e. $d(\mathbf{v}, \mathbf{r}) = t'$

→ Obviously, $d(\mathbf{v}, \mathbf{w}) \geq d_{\min} \geq 2t + 1$, and hence $d(\mathbf{w}, \mathbf{r}) \geq 2t + 1 - t'$

Linear block code

$\rightarrow$ If $t' < t$ then $d(\mathbf{w}, \mathbf{r}) > t$

$\rightarrow$ Thus if an error pattern of t or fewer errors occurs, the received vector $\mathbf{r}$ is closer in Hamming distance to the transmitted codeword $\mathbf{v}$ than any other codeword $\mathbf{w}$ in C

Linear block code

$\rightarrow$ If $t' < t$ then $d(\mathbf{w}, \mathbf{r}) > t$

$\rightarrow$ Thus if an error pattern of t or fewer errors occurs, the received vector $\mathbf{r}$ is closer in Hamming distance to the transmitted codeword $\mathbf{v}$ than any other codeword $\mathbf{w}$ in C

$\rightarrow$ According to ML decoding scheme, it is a correct transmitted codeword, thus the errors are corrected.

Quantum information theory

- Classical information is carried by systems with a definite state, and it can be replicated and measured without being altered

Quantum information theory

- Classical information is carried by systems with a definite state, and it can be replicated and measured without being altered
- Quantum information is encoded as a property of quantum systems (e.g., photon polarization or particle spin) and has special properties such as superposition and entanglement with no classical counterpart; quantum information cannot be cloned, and it is altered as a result of a measurement

Quantum information theory

- Classical information is carried by systems with a definite state, and it can be replicated and measured without being altered
- Quantum information is encoded as a property of quantum systems (e.g., photon polarization or particle spin) and has special properties such as superposition and entanglement with no classical counterpart; quantum information cannot be cloned, and it is altered as a result of a measurement

transmission of classical information over quantum channels

Quantum information theory

- Classical information is carried by systems with a definite state, and it can be replicated and measured without being altered
- Quantum information is encoded as a property of quantum systems (e.g., photon polarization or particle spin) and has special properties such as superposition and entanglement with no classical counterpart; quantum information cannot be cloned, and it is altered as a result of a measurement

    transmission of classical information over quantum channels
    transmission of quantum information over quantum channels

Quantum information theory

- Classical information is carried by systems with a definite state, and it can be replicated and measured without being altered
- Quantum information is encoded as a property of quantum systems (e.g., photon polarization or particle spin) and has special properties such as superposition and entanglement with no classical counterpart; quantum information cannot be cloned, and it is altered as a result of a measurement

    transmission of classical information over quantum channels
    transmission of quantum information over quantum channels
    effect of quantum entanglement on information transmission

Quantum information theory

- Classical information is carried by systems with a definite state, and it can be replicated and measured without being altered
- Quantum information is encoded as a property of quantum systems (e.g., photon polarization or particle spin) and has special properties such as superposition and entanglement with no classical counterpart; quantum information cannot be cloned, and it is altered as a result of a measurement

    transmission of classical information over quantum channels
    transmission of quantum information over quantum channels
    effect of quantum entanglement on information transmission
    informational aspect of the quantum measurement process, the trade offs between the disturbance of the quantum state and the accuracy of the measurement

von Neumann entropy

For a density matrix ρ, of an n-qubit system

$$S(\rho) = -tr[\rho \log \rho]$$

von Neumann entropy

For a density matrix ρ, of an n-qubit system

$$S(\rho) = -tr[\rho \log \rho]$$

Setting, $\rho = \sum_{j=1}^{2^n} p_i \left|e_j\right\rangle \left\langle e_j\right|$, (spectral decomposition)

$$\log \rho = \sum_{j=1}^{2^n} (\log p_j) \left|e_j\right\rangle \left\langle e_j\right|$$

and hence

$$S(\rho) = -tr\left(\sum_{j=1}^{2^n} p_j \left|e_j\right\rangle \left\langle e_j\right| \sum_{i=1}^{2^n} \log p_i \left|e_i\right\rangle \left\langle e_i\right|\right) = -\sum_{j=1}^{2^n} p_j \log p_j = H(p_1, \ldots$$

von Neumann entropy

For a density matrix ρ, of an n-qubit system

$$S(\rho) = -tr[\rho \log \rho]$$

Setting, $\rho = \sum_{j=1}^{2^n} p_i \left|e_j\right\rangle \left\langle e_j\right|$, (spectral decomposition)

$$\log \rho = \sum_{j=1}^{2^n} (\log p_j) \left|e_j\right\rangle \left\langle e_j\right|$$

and hence

$$S(\rho) = -tr \left(\sum_{j=1}^{2^n} p_j \left|e_j\right\rangle \left\langle e_j\right| \sum_{i=1}^{2^n} \log p_i \left|e_i\right\rangle \left\langle e_i\right| \right) = -\sum_{j=1}^{2^n} p_j \log p_j = H(p_1, \ldots$$

Question What does this mean?

von Neumann entropy

Observations

- If $\rho = |\psi\rangle \langle\psi|$, is a pure state then it has only one eigenvalue 1, hence trace is 1 and $S(\rho) = 0$

von Neumann entropy

Observations

- If $\rho = |\psi\rangle \langle\psi|$, is a pure state then it has only one eigenvalue 1, hence trace is 1 and $S(\rho) = 0$
- Consider an ensemble of pure states $|e_j\rangle$, $1 \leq j \leq N$, and prepare a mixed state with $|e_j\rangle$ probability p_j

von Neumann entropy

Observations

- If $\rho = |\psi\rangle \langle\psi|$, is a pure state then it has only one eigenvalue 1, hence trace is 1 and $S(\rho) = 0$
- Consider an ensemble of pure states $|e_j\rangle$, $1 \leq j \leq N$, and prepare a mixed state with $|e_j\rangle$ probability p_j
- We can safely say that von Neumann entropy is the least amount of information to be used to create ρ, and equivalently we can say that it is the minimum amount of classical information that we can access from ρ

von Neumann entropy

Observations

- If $\rho = |\psi\rangle \langle\psi|$, is a pure state then it has only one eigenvalue 1, hence trace is 1 and $S(\rho) = 0$
- Consider an ensemble of pure states $|e_j\rangle$, $1 \leq j \leq N$, and prepare a mixed state with $|e_j\rangle$ probability p_j
- We can safely say that von Neumann entropy is the least amount of information to be used to create ρ, and equivalently we can say that it is the minimum amount of classical information that we can access from ρ
- Consider evolution of a system described by ρ : $\rho(t) = e^{-iHt}\rho e^{iH(t)}$, then $S(\rho(t)) = S(\rho)$ - second law of thermodynamics, the entropy of a closed system never decreases

von Neumann entropy

Let ρ_{AB} denote a 'joint' density matrix corresponding to a bipartite/composite system. Then

$$\rho_A = tr_B(\rho_{AB}),\ \rho_B = tr_A(\rho_{AB})$$

are partial traces of ρ_{AB}

von Neumann entropy

Let ρ_{AB} denote a 'joint' density matrix corresponding to a bipartite/composite system. Then

$$\rho_A = tr_B(\rho_{AB}),\ \rho_B = tr_A(\rho_{AB})$$

are partial traces of ρ_{AB}

joint von Neumann entropy: $S(A, B) = S(\rho_{AB})$

von Neumann entropy

Let ρ_{AB} denote a 'joint' density matrix corresponding to a bipartite/composite system. Then

$$\rho_A = tr_B(\rho_{AB}),\ \rho_B = tr_A(\rho_{AB})$$

are partial traces of ρ_{AB}

joint von Neumann entropy: $S(A, B) = S(\rho_{AB})$

conditional von Neumann entropy of system, A, conditioned by system, B:

$$S(A|B) = S(\rho_{AB}) - S(\rho_B)$$

von Neumann entropy

Let ρ_{AB} denote a 'joint' density matrix corresponding to a bipartite/composite system. Then

$$\rho_A = tr_B(\rho_{AB}),\ \rho_B = tr_A(\rho_{AB})$$

are partial traces of ρ_{AB}

joint von Neumann entropy: $S(A, B) = S(\rho_{AB})$

conditional von Neumann entropy of system, A, conditioned by system, B:

$$S(A|B) = S(\rho_{AB}) - S(\rho_B)$$

Note Conditioning cannot increase entropy

von Neumann entropy

mutual information: for a pair of systems A, B

$$I(A;B) = S(\rho_A) + S(\rho_B) - S(\rho_{AB}) = S(\rho_A) - S(A|B)$$

von Neumann entropy

mutual information: for a pair of systems A, B

$$I(A;B) = S(\rho_A) + S(\rho_B) - S(\rho_{AB}) = S(\rho_A) - S(A|B)$$

relative entropy: $S(\rho_1 \| \rho_2) = tr(\rho_1(\log \rho_1) - \log \rho_2)$

von Neumann entropy

mutual information: for a pair of systems A, B

$$I(A;B) = S(\rho_A) + S(\rho_B) - S(\rho_{AB}) = S(\rho_A) - S(A|B)$$

relative entropy: $S(\rho_1 \| \rho_2) = tr(\rho_1(\log \rho_1) - \log \rho_2)$

$$S'(\rho_1 \| \rho_2) = tr(\rho_1 \log\{\rho_1^{1/2} \rho_2^{-1} \rho_1^{1/2}\})$$

Question Are these generalizations of classical relative entropy? Which one to choose?

von Neumann entropy

mutual information: for a pair of systems A, B

$$I(A; B) = S(\rho_A) + S(\rho_B) - S(\rho_{AB}) = S(\rho_A) - S(A|B)$$

relative entropy: $S(\rho_1 \| \rho_2) = tr(\rho_1(\log \rho_1) - \log \rho_2)$

$$S'(\rho_1 \| \rho_2) = tr(\rho_1 \log\{\rho_1^{1/2} \rho_2^{-1} \rho_1^{1/2}\})$$

Question Are these generalizations of classical relative entropy? Which one to choose?
Justification:

$$S(\rho_1 \| \rho_2) = \lim_{\epsilon \to 0} S(\rho_1 \| \rho_2 + \epsilon I)$$

Quantum information processing

von Neumann entropy - is it related to the fundamental limit of compression?

[5] Wilde, M.M., 2013. Quantum information theory. Cambridge university press

Quantum information processing

von Neumann entropy - is it related to the fundamental limit of compression?

- A simple model of quantum information source[5] is an ensemble of quantum states $\{p_X(x), |\psi_x\rangle\}$ - the source outputs the state $|\psi_x\rangle$ with probability $p_X(x)$

[5] Wilde, M.M., 2013. Quantum information theory. Cambridge university press

Quantum information processing

von Neumann entropy - is it related to the fundamental limit of compression?

- A simple model of quantum information source[5] is an ensemble of quantum states $\{p_X(x), |\psi_x\rangle\}$ - the source outputs the state $|\psi_x\rangle$ with probability $p_X(x)$
- The states $\{|\psi_x\rangle\}$ do not necessarily have to form an ONB

[5] Wilde, M.M., 2013. Quantum information theory. Cambridge university press

Quantum information processing

von Neumann entropy - is it related to the fundamental limit of compression?

- A simple model of quantum information source[5] is an ensemble of quantum states $\{p_X(x), |\psi_x\rangle\}$ - the source outputs the state $|\psi_x\rangle$ with probability $p_X(x)$
- The states $\{|\psi_x\rangle\}$ do not necessarily have to form an ONB

An obvious strategy - ignoring the quantum input and treating x as the corresponding classical input

[5] Wilde, M.M., 2013. Quantum information theory. Cambridge university press

Quantum information processing

von Neumann entropy - is it related to the fundamental limit of compression?

- A simple model of quantum information source[5] is an ensemble of quantum states $\{p_X(x), |\psi_x\rangle\}$ - the source outputs the state $|\psi_x\rangle$ with probability $p_X(x)$
- The states $\{|\psi_x\rangle\}$ do not necessarily have to form an ONB

An obvious strategy - ignoring the quantum input and treating x as the corresponding classical input

Question How can we use a quantum channel? What is a noiseless quantum channel?

[5]Wilde, M.M., 2013. Quantum information theory. Cambridge university press

Quantum information processing

Alice's State preparation the information source outputs a sequence $|\psi_{x^n}\rangle_{A^n}$ of quantum states according to the ensemble $\{p_X(x), |\psi_x\rangle\}$, where

$$|\psi_{x^n}\rangle_{A^n} = |\psi_{x_1}\rangle_{A_1} \otimes \ldots \otimes |\psi_{x_n}\rangle_{A_n}$$

Quantum information processing

Alice's State preparation the information source outputs a sequence $|\psi_{x^n}\rangle_{A^n}$ of quantum states according to the ensemble $\{p_X(x), |\psi_x\rangle\}$, where

$$|\psi_{x^n}\rangle_{A^n} = |\psi_{x_1}\rangle_{A_1} \otimes \ldots \otimes |\psi_{x_n}\rangle_{A_n}$$

The density operator is $\rho^{\otimes n}$ where

$$\rho = \sum_x p_X(x) |\psi_x\rangle \langle\psi_x|$$

Quantum information processing

Alice's State preparation the information source outputs a sequence $|\psi_{x^n}\rangle_{A^n}$ of quantum states according to the ensemble $\{p_X(x), |\psi_x\rangle\}$, where

$$|\psi_{x^n}\rangle_{A^n} = |\psi_{x_1}\rangle_{A_1} \otimes \ldots \otimes |\psi_{x_n}\rangle_{A_n}$$

The density operator is $\rho^{\otimes n}$ where

$$\rho = \sum_x p_X(x) |\psi_x\rangle \langle\psi_x|$$

Alice can think about purification of the density operator as

$$|\phi_\rho\rangle_{RA} = \sum_x \sqrt{p_X(x)}\, |x\rangle_R |\psi_x\rangle_A ,$$

where R is the lebel for the inaccessible reference system, hence the resulting iid state is $|\psi_\rho\rangle_{RA}^{\otimes n}$

Quantum information processing

Encoding Alice encodes the systems A^n according to a compression channel $\mathcal{E}_{A^n \to W}$, where W is a quantum system of dimension 2^{nR}, where R is the rate of the compression

Quantum information processing

Encoding Alice encodes the systems A^n according to a compression channel $\mathcal{E}_{A^n \to W}$, where W is a quantum system of dimension 2^{nR}, where R is the rate of the compression

Note
$$R = \frac{1}{n} \log \dim(H_W)$$

Quantum information processing

Encoding Alice encodes the systems A^n according to a compression channel $\mathcal{E}_{A^n \to W}$, where W is a quantum system of dimension 2^{nR}, where R is the rate of the compression

Note

$$R = \frac{1}{n} \log \dim(H_W)$$

Transmission Alice transmits the system W to Bob using nR noiseless qubit channels

Quantum information processing

Encoding Alice encodes the systems A^n according to a compression channel $\mathcal{E}_{A^n \to W}$, where W is a quantum system of dimension 2^{nR}, where R is the rate of the compression

Note

$$R = \frac{1}{n} \log \dim(H_W)$$

Transmission Alice transmits the system W to Bob using nR noiseless qubit channels

Decoding Bob sends the system W through a decompression channel $\mathcal{D}_{W \to \widehat{A^n}}$

Quantum information processing

The protocol has ϵ-error if

$$\frac{1}{2}\left\|(|\phi_\rho\rangle_{RA})^{\otimes n} - (\mathcal{D}_{W\to\widehat{A^n}} \circ \mathcal{E}_{A^n\to W})(|\phi_\rho\rangle)_{RA}^{\otimes n}\right\|_1 \leq \epsilon$$

Quantum information processing

The protocol has ϵ-error if

$$\frac{1}{2}\left\|(|\phi_\rho\rangle_{RA})^{\otimes n} - (\mathcal{D}_{W\to\widehat{A^n}} \circ \mathcal{E}_{A^n\to W})(|\phi_\rho\rangle)_{RA}^{\otimes n}\right\|_1 \leq \epsilon$$

- ▷ a quantum compression rate is achivable is there exists an $(n, R+\delta, \epsilon)$ quantum compression code for all $\delta > 0$, $\epsilon \in (0,1)$, for sufficiently large n

Quantum information processing

The protocol has ϵ-error if

$$\frac{1}{2}\left\|(|\phi_\rho\rangle_{RA})^{\otimes n} - (\mathcal{D}_{W\to\widehat{A^n}} \circ \mathcal{E}_{A^n\to W})(|\phi_\rho\rangle)^{\otimes n}_{RA}\right\|_1 \leq \epsilon$$

- a quantum compression rate is achivable is there exists an $(n, R+\delta, \epsilon)$ quantum compression code for all $\delta > 0$, $\epsilon \in (0,1)$, for sufficiently large n
- The quantum data compression limit of ρ is equal to the infimum of all achievable quantum compression rates

Schumacher compression

Data compression theorem Suppose ρ is the density matrix corresponding to a quantum information source. then the von Neumann entropy is equal to the quantum data compression limit of ρ

Schumacher compression

Data compression theorem Suppose ρ is the density matrix corresponding to a quantum information source. then the von Neumann entropy is equal to the quantum data compression limit of ρ

Quantum channel A quantum channel is a completely positive map

Schumacher compression

Data compression theorem Suppose ρ is the density matrix corresponding to a quantum information source. then the von Neumann entropy is equal to the quantum data compression limit of ρ

Quantum channel A quantum channel is a completely positive map
Positive map A linear map $\mathcal{M} : \mathcal{L}(H_A) \to \mathcal{L}(H_B)$ is positive if $\mathcal{M}(X_A)$ is positive semi-definite for all positive semi-definite $X_A \in \mathcal{L}(H_A)$

Schumacher compression

Data compression theorem Suppose ρ is the density matrix corresponding to a quantum information source. then the von Neumann entropy is equal to the quantum data compression limit of ρ

Quantum channel A quantum channel is a completely positive map
Positive map A linear map $\mathcal{M} : \mathcal{L}(H_A) \to \mathcal{L}(H_B)$ is positive if $\mathcal{M}(X_A)$ is positive semi-definite for all positive semi-definite $X_A \in \mathcal{L}(H_A)$
Complete positivity A linear map $\mathcal{M} : \mathcal{L}(H_A) \to \mathcal{L}(H_B)$ is completely positive if $ID_m \otimes \mathcal{M}$ is a positive map

Quantum channel

Example Unitary evolution is a special kind of quantum channel. Under the action of a unitary channel $\mathcal{U}$, the state evolves as

$$\mathcal{U}(\rho) = U\rho U^\dagger$$

Quantum channel

Example Unitary evolution is a special kind of quantum channel. Under the action of a unitary channel $\mathcal{U}$, the state evolves as

$$\mathcal{U}(\rho) = U\rho U^\dagger$$

- ▷ Classical-to-classical channels
- ▷ Classical-to-quantum-channels
- ▷ Quantum-to-classical channels (measurement channels)

Quantum channel

Example Unitary evolution is a special kind of quantum channel. Under the action of a unitary channel $\mathcal{U}$, the state evolves as

$$\mathcal{U}(\rho) = U\rho U^{\dagger}$$

- ▷ Classical-to-classical channels
- ▷ Classical-to-quantum-channels
- ▷ Quantum-to-classical channels (measurement channels)

The Holevo bound an upper bound of the accessible information in a quantum measurement

Thanks for your attention!!

For questions or comments: bibhas.adhikari AT gmail DOT com