Exclusive or

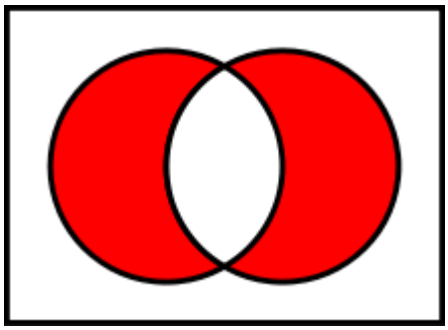
From Wikipedia, the free encyclopedia

Exclusive or or **exclusive disjunction** is a logical operation that outputs true only when inputs differ (one is true, the other is false).[1]

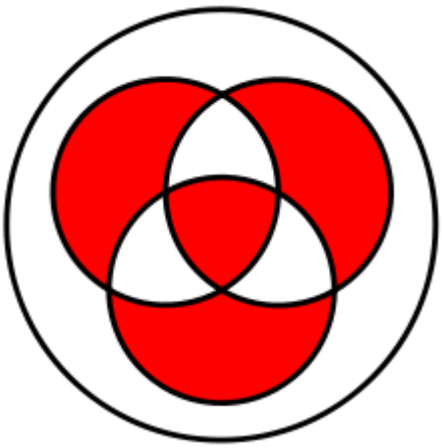
It is symbolized by the prefix operator **J**[2] and by the infix operators **XOR** (/ˌɛks ˈɔːr/), **EOR**, **EXOR**, ⊻, ⊕, ↮, and ≢. The negation of XOR is logical biconditional, which outputs true only when both inputs are the same.

It gains the name "exclusive or" because the meaning of "or" is ambiguous when both operands are true; the exclusive or operator *excludes* that case. This is sometimes thought of as "one or the other but not both". This could be written as "A or B, but not, A and B".

More generally, XOR is true only when an odd number of inputs are true. A chain of XORs—*a* XOR *b* XOR *c* XOR *d* (and so on)—is true whenever an odd number of the inputs are true and is false whenever an even number of inputs are true.



Venn diagram of $A \oplus B$



Venn diagram of $A \oplus B \oplus C$

Contents

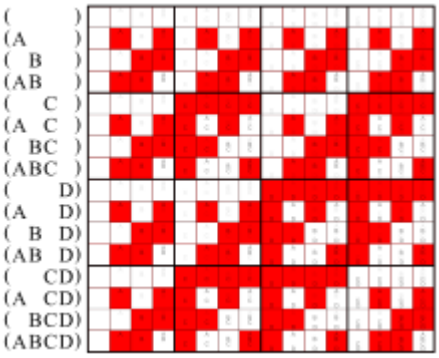
Truth table

The truth table of A XOR B shows that it outputs true whenever the inputs differ:

XOR truth table

Input		Output
A	B	
0	0	0
0	1	1
1	0	1
1	1	0

- 0, false
- 1, true



Arguments on the left combined by XOR. This is a binary Walsh matrix (cf. Hadamard code).

Equivalences, elimination, and introduction

Exclusive disjunction essentially means 'either one, but not both nor none'. In other words, the statement is true if and only if one is true and the other is false. For example, if two horses are racing, then one of the two will win the race, but not both of them. The exclusive disjunction $p \oplus q$, or J*pq*, can be expressed in terms of the logical conjunction ("logical and", $\wedge$), the disjunction ("logical or", $\vee$), and the negation ($\neg$) as follows:

$$p \oplus q \quad = \quad (p \vee q) \wedge \neg(p \wedge q)$$

The exclusive disjunction $p \oplus q$ can also be expressed in the following way:

$$p \oplus q \quad = \quad (p \wedge \neg q) \vee (\neg p \wedge q)$$

This representation of XOR may be found useful when constructing a circuit or network, because it has only one $\neg$ operation and small number of $\wedge$ and $\vee$ operations. A proof of this identity is given below:

$$\begin{aligned}
p \oplus q &= (p \wedge \neg q) \vee (\neg p \wedge q) \\
&= ((p \wedge \neg q) \vee \neg p) \wedge ((p \wedge \neg q) \vee q) \\
&= ((p \vee \neg p) \wedge (\neg q \vee \neg p)) \wedge ((p \vee q) \wedge (\neg q \vee q)) \\
&= (\neg p \vee \neg q) \wedge (p \vee q) \\
&= \neg(p \wedge q) \wedge (p \vee q)
\end{aligned}$$

It is sometimes useful to write $p \oplus q$ in the following way:

$$p \oplus q \quad = \quad \neg((p \wedge q) \vee (\neg p \wedge \neg q))$$

or:

$$p \oplus q \quad = \quad (p \vee q) \wedge (\neg p \vee \neg q)$$

This equivalence can be established by applying De Morgan's laws twice to the fourth line of the above proof.

The exclusive or is also equivalent to the negation of a logical biconditional, by the rules of material implication (a material conditional is equivalent to the disjunction of the negation of its antecedent and its consequence) and material equivalence.

In summary, we have, in mathematical and in engineering notation:

$$\begin{aligned}
p \oplus q &= (p \wedge \neg q) \vee (\neg p \wedge q) &&= p\overline{q} + \overline{p}q \\
&= (p \vee q) \wedge (\neg p \vee \neg q) &&= (p + q)(\overline{p} + \overline{q}) \\
&= (p \vee q) \wedge \neg(p \wedge q) &&= (p + q)(\overline{pq})
\end{aligned}$$

Relation to modern algebra

Although the operators $\wedge$ (conjunction) and $\vee$ (disjunction) are very useful in logic systems, they fail a more generalizable structure in the following way:

The systems $(\{T, F\}, \wedge)$ and $(\{T, F\}, \vee)$ are monoids, but neither is a group. This unfortunately prevents the combination of these two systems into larger structures, such as a mathematical ring.

However, the system using exclusive or $(\{T, F\}, \oplus)$ **is** an abelian group. The combination of operators $\wedge$ and $\oplus$ over elements $\{T, F\}$ produce the well-known field F_2. This field can represent any logic obtainable with the system $(\wedge, \vee)$ and has the added benefit of the arsenal of algebraic analysis tools for fields.

More specifically, if one associates F with 0 and T with 1, one can interpret the logical "AND" operation as multiplication on F_2 and the "XOR" operation as addition on F_2:

$$r = p \wedge q \quad \Leftrightarrow \quad r = p \cdot q \pmod 2$$
$$r = p \oplus q \quad \Leftrightarrow \quad r = p + q \pmod 2$$

Using this basis to describe a boolean system is referred to as algebraic normal form.

Exclusive "or" in English

The Oxford English Dictionary explains "either ... or" as follows:

> The primary function of *either*, etc., is to emphasize the perfect indifference of the two (or more) things or courses ... ; but a secondary function is to emphasize the mutual exclusiveness, = either of the two, but not both.[3]

The exclusive-or explicitly states "one or the other, but not neither nor both." However, the mapping correspondence between formal Boolean operators and natural language conjunctions is far from simple or one-to-one, and has been studied for decades in linguistics and analytic philosophy.

Following this kind of common-sense intuition about "or", it is sometimes argued that in many natural languages, English included, the word "or" has an "exclusive" sense.[4] The **exclusive disjunction** of a pair of propositions, (*p*, *q*), is supposed to mean that *p* is true or *q* is true, but not both. For example, it might be argued that the normal intention of a statement like "You may have coffee, or you may have tea" is to stipulate that exactly one of the conditions can be true. Certainly under some circumstances a sentence like this example should be taken as forbidding the possibility of one's accepting both options. Even so, there is good reason to suppose that this sort of sentence is not disjunctive at all. If all we know about some disjunction is that it is true overall, we cannot be sure which of its disjuncts is true. For example, if a woman has been told that her friend is either at the snack bar or on the tennis court, she cannot validly infer that he is on the tennis court. But if her waiter tells her that she may have coffee or she may have tea, she can validly infer that she may have tea. Nothing classically thought of as a disjunction has this property. This is so even given that she might reasonably take her waiter as having denied her the possibility of having both coffee and tea.

In English, the construct "either ... or" is usually used to indicate exclusive or and "or" generally used for inclusive. But in Spanish, the word "*o*" (or) can be used in the form "*p o q*" (exclusive) or the form "*o p o q*" (inclusive). Some may contend that any binary or other n-ary exclusive "or" is true if and only if it has an odd number of true inputs (this is not, however, the only reasonable definition; for example, digital xor gates with multiple inputs typically do not use that definition), and that there is no conjunction in English that has this general property. For example, Barrett and Stenner contend in the 1971 article "The Myth of the Exclusive 'Or' " (Mind, 80 (317), 116–121) that no author has produced an example of an English or-sentence that appears to be false because both of its inputs are true, and brush off or-sentences such as "The light bulb is either on or off" as reflecting particular facts about the world rather than the nature of the word "or". However, the "barber paradox"—Everybody in town shaves himself or is shaved by the barber, who shaves the barber? -- would not be paradoxical if "or" could not be exclusive (although a purist could say that "either" is required in the statement of the paradox).

Whether these examples can be considered "natural language" is another question. Certainly when one sees a menu stating "Lunch special: sandwich and soup or salad" (parsed as "sandwich and (soup or salad)" according to common usage in the restaurant trade), one would not expect to be permitted to order both soup and salad. Nor would one expect to order neither soup nor salad, because that belies the nature of the "special", that ordering the two items together is cheaper than ordering them a la carte. Similarly, a lunch special consisting of one meat, French fries or mashed potatoes and vegetable would consist of three items, only one of which would be a form of potato. If one wanted to have meat and both kinds of potatoes, one would ask if it were possible to substitute a second order of potatoes for the vegetable. And, one would not expect to be permitted to have both types of potato and vegetable, because the result would be a vegetable plate rather than a meat plate.

Alternative symbols

The symbol used for exclusive disjunction varies from one field of application to the next, and even depends on the properties being emphasized in a given context of discussion. In addition to the abbreviation "XOR", any of the following symbols may also be seen:

- A plus sign (+). This makes sense mathematically because exclusive disjunction corresponds to addition modulo 2, which has the following addition table, clearly isomorphic to the one above:

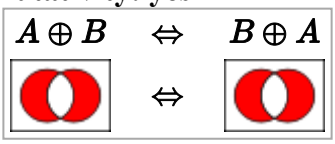
Addition modulo 2

p	q	$p+q$
0	0	0
0	1	1
1	0	1
1	1	0

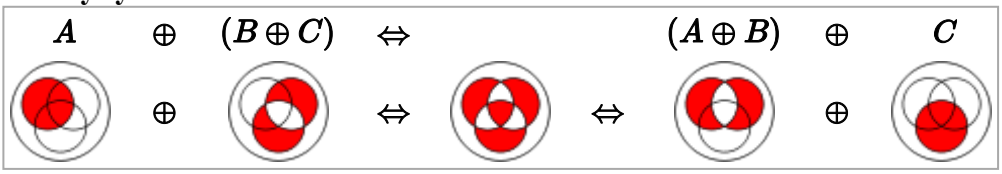
- The use of the plus sign has the added advantage that all of the ordinary algebraic properties of mathematical rings and fields can be used without further ado. However, the plus sign is also used for inclusive disjunction in some notation systems.
- A plus sign that is modified in some way, such as being encircled ($\oplus$). This usage faces the objection that this same symbol is already used in mathematics for the *direct sum* of algebraic structures.
- A prefixed J, as in Jpq.
- An inclusive disjunction symbol ($\vee$) that is modified in some way, such as being underlined ($\underline{\vee}$) or with dot above ($\dot{\vee}$).
- In several programming languages, such as C, C++, C#, D, Java, Perl, Ruby, PHP and Python, a caret (^) is used to denote the bitwise XOR operator. This is not used outside of programming contexts because it is too easily confused with other uses of the caret.
- The symbol ⋋, sometimes written as >< or as >-<.

Properties

Commutativity: yes



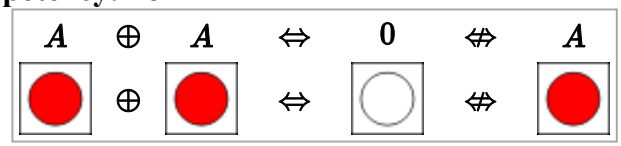
Associativity: yes



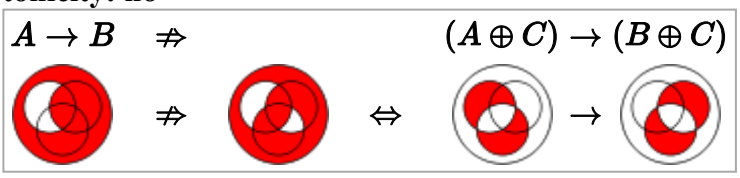
Distributivity:

The exclusive or doesn't distribute over any binary function (not even itself), but logical conjunction distributes over exclusive or. $C \wedge (A \oplus B) = C \wedge A \oplus C \wedge B$ (Conjunction and exclusive or form the multiplication and addition operations of a field GF(2), and as in any field they obey the distributive law.)

Idempotency: no

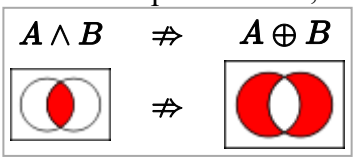


Monotonicity: no



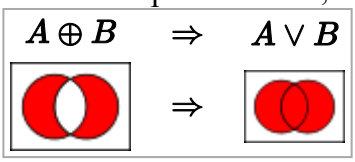
Truth-preserving: no

When all inputs are true, the output is not true.



Falsehood-preserving: yes

When all inputs are false, the output is false.



Walsh spectrum: (2,0,0,−2)

Non-linearity: 0

The function is linear.

If using binary values for true (1) and false (0), then *exclusive or* works exactly like addition modulo 2.

Computer science

Bitwise operation

Traditional symbolic representation of an XOR logic gate

Exclusive disjunction is often used for bitwise operations. Examples:

- 1 XOR 1 = 0
- 1 XOR 0 = 1
- 0 XOR 1 = 1
- 0 XOR 0 = 0
- 1110_2 XOR $1001_2 = 0111_2$ (this is equivalent to addition without carry)

As noted above, since exclusive disjunction is identical to addition modulo 2, the bitwise exclusive disjunction of two *n*-bit strings is identical to the standard vector of addition in the vector space $(\mathbb{Z}/2\mathbb{Z})^n$.

In computer science, exclusive disjunction has several uses:

- It tells whether two bits are unequal.
- It is an optional bit-flipper (the deciding input chooses whether to invert the data input).
- It tells whether there is an odd number of 1 bits ($A \oplus B \oplus C \oplus D \oplus E$ is true iff an odd number of the variables are true).

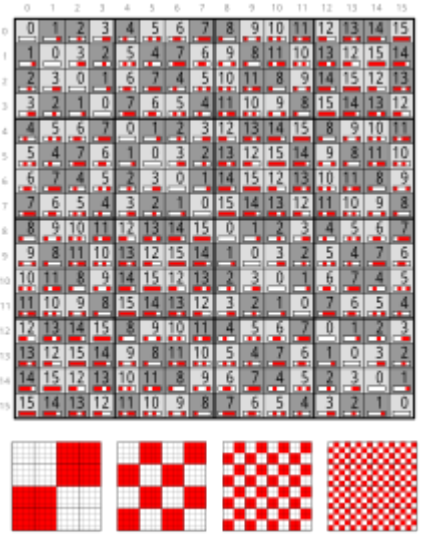
In logical circuits, a simple adder can be made with an XOR gate to add the numbers, and a series of AND, OR and NOT gates to create the carry output.

On some computer architectures, it is more efficient to store a zero in a register by XOR-ing the register with itself (bits XOR-ed with themselves are always zero) instead of loading and storing the value zero.

In simple threshold activated neural networks, modeling the XOR function requires a second layer because XOR is not a linearly separable function.

Exclusive-or is sometimes used as a simple mixing function in cryptography, for example, with one-time pad or Feistel network systems.

Exclusive-or is also heavily used in block ciphers such as AES (Rijndael) or Serpent and in block cipher implementation (CBC, CFB, OFB or CTR).



Nimber addition is the *exclusive or* of nonnegative integers in binary representation. This is also the vector addition in $(\mathbb{Z}/2\mathbb{Z})^4$.

Similarly, XOR can be used in generating entropy pools for hardware random number generators. The XOR operation preserves randomness, meaning that a random bit XORed with a non-random bit will result in a random bit. Multiple sources of potentially random data can be combined using XOR, and the unpredictability of the output is guaranteed to be at least as good as the best individual source.[5]

XOR is used in RAID 3–6 for creating parity information. For example, RAID can "back up" bytes 10011100_2 and 01101100_2 from two (or more) hard drives by XORing the just mentioned bytes, resulting in (11110000_2) and writing it to another drive. Under this method, if any one of the three hard drives are lost, the lost byte can be re-created by XORing bytes from the remaining drives. For instance, if the drive containing 01101100_2 is lost, 10011100_2 and 11110000_2 can be XORed to recover the lost byte.[6]

XOR is also used to detect an overflow in the result of a signed binary arithmetic operation. If the leftmost retained bit of the result is not the same as the infinite number of digits to the left, then that means overflow occurred. XORing those two bits will give a "1" if there is an overflow.

XOR can be used to swap two numeric variables in computers, using the XOR swap algorithm; however this is regarded as more of a curiosity and not encouraged in practice.

XOR linked lists leverage XOR properties in order to save space to represent doubly linked list data structures.

In computer graphics, XOR-based drawing methods are often used to manage such items as bounding boxes and cursors on systems without alpha channels or overlay planes.

Encodings

Apart from the ASCII codes, the operator is encoded at `U+22BB` ⊻ XOR (HTML `&#8891;`) and `U+2295` ⊕ CIRCLED PLUS (HTML `&#8853;` · `&oplus;`), both in block Mathematical Operators.

See also

- Material conditional • (Paradox)
- Affirming a disjunct
- Ampheck
- Boolean algebra (logic)

- Boolean domain
- Boolean function
- Boolean-valued function
- Controlled NOT gate
- Disjunctive syllogism
- First-order logic
- Inclusive or
- Involution
- List of Boolean algebra topics
- Logical graph
- Logical value
- Operation
- Parity bit
- Propositional calculus
- Rule 90
- Symmetric difference
- XOR cipher
- XOR gate
- XOR linked list

Notes

1. Germundsson, Roger; Weisstein, Eric. "XOR" (http://mathworld.wolfram.com/XOR.html). *MathWorld*. Wolfram Research. Retrieved 17 June 2015.
2. Craig, Edward, ed. (1998), *Routledge Encyclopedia of Philosophy* (https://books.google.com/books?id=HP9O6OM4iOgC&pg=PA496), **10**, Taylor & Francis, p. 496, ISBN 9780415073103
3. or, conj.2 (adv.3) 2a *Oxford English Dictionary*, second edition (1989). OED Online.
4. Jennings quotes numerous authors saying that the word "or" has an exclusive sense. See Chapter 3, "The First Myth of 'Or'":
   Jennings, R. E. (1994). *The Genealogy of Disjunction*. New York: Oxford University Press.
5. Davies, Robert B (28 February 2002). "Exclusive OR (XOR) and hardware random number generators" (http://www.robertnz.net/pdf/xor2.pdf) (PDF). Retrieved 28 August 2013.
6. Nobel, Rickard (26 July 2011). "How RAID 5 actually works" (http://rickardnobel.se/how-raid5-works). Retrieved 23 March 2017.

External links

- An example of XOR being used in cryptography (http://www.codeplex.com/rexor)