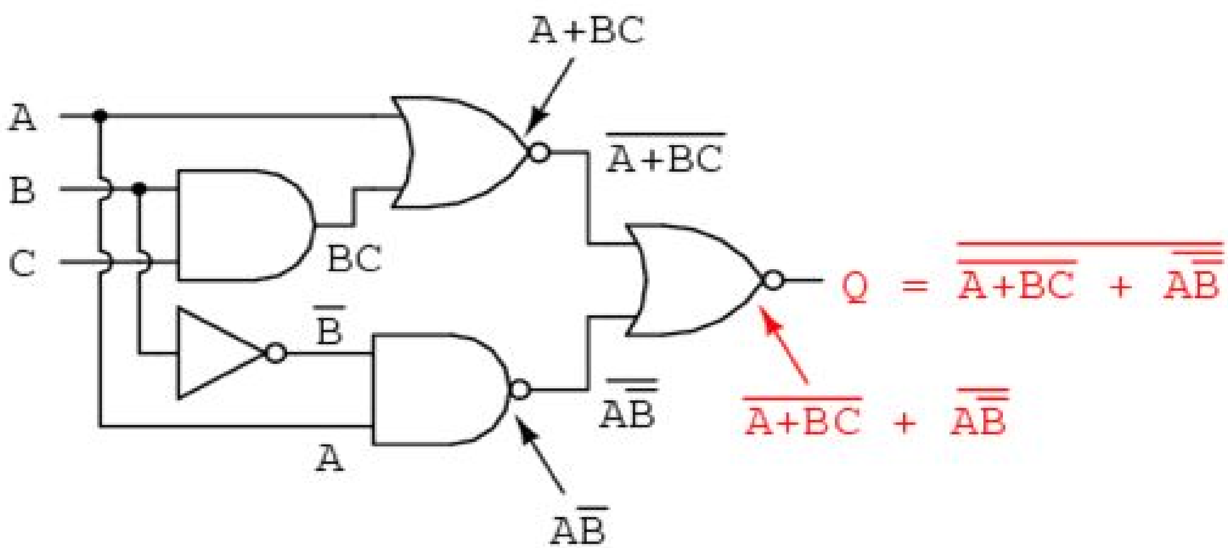
De Morgan’s Law and its History

Jeffrey Lansford, Ramon Ramirez, Fidel Garcia,
Brandon Ihlein, Antonio Maldonado, Raymond Fierro
Computer Science Department Undergraduate Students

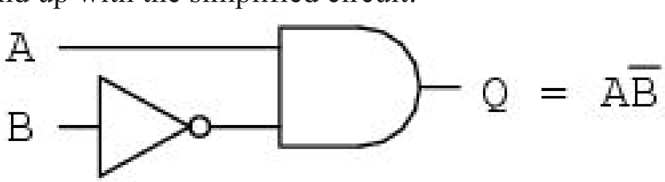
December 4, 2018

Description of De Morgan’s Law:

Although discovered centuries ago, De Morgan’s Law is still heavily used today in the fields of engineering, set theory, propositional logic, and boolean algebra. De Morgan’s Laws describe how mathematical statements and concepts are related through their opposites. In set theory, the laws relate the intersection and union of sets through complements. The laws are then used to simplify or represent the sets in different ways to show that one expression is equivalent to another. For instance, say the set is: $(A \cup B)^c$ (the complement of the union of A & B), the equivalent set notation, derived using De Morgan’s Law is: $A^c \cap B^c$ (The complement of A intersect complement of B). This means you could apply De Morgan’s Law and other properties and rules to simplify long set expressions. In propositional logic, the laws relate the conjunctions and disjunctions of propositions using negation. In computer/electrical engineering, De Morgan’s law is applied to develop logic gates for circuits. Using the laws you can redesign any logic gate with only NAND, NOR and NOT gates. These gates are the best for hardware and at times can be significantly faster, and cheaper to produce (Kesavan, 2014). An example from Kuphaldt (2013) uses De Morgan’s Law to simplify this circuit:



By applying De Morgan’s law and other properties of Boolean logic to the boolean equation you end up with the simplified circuit:



In that equation specifically uses De Morgan's Law to simplify it at the beginning by make using of the negation of the entire equation and the "or" of A+BC and AB. It then becomes double negation of (A +BC) added with double negation of (A negate(B)). After that main step and the use of other properties your left with the simplified logic gate. These two gates are equivalent but the latter is much better as it is simpler and in turn would use up very little resources and money to produce. Another interesting application is that NAND gates can make up any gate and is universal (Hayes, 2018). By applying De Morgan's Law to a logical equation, you can make an equivalent equation using only NAND gates. It may or not be shorter depending on the equation its applied to, but is universal thanks to the application of De Morgan's Law. Though De Morgan's Law can also be applied to logic and sets, it is most widely used in building circuits. DeMorgan's law simplifies our circuits which creates more efficient and cost effective hardware.

History:

Aristotle made the first observation of negation in recorded history. He explains that some expressions are equivalent to each other like "possible", "not impossible", and "not necessary not" and concluded that they can be interchangeable.

Possible	Not Impossible	Not Necessary Not
Not Possible	Impossible	Necessary Not
Possible Not	Not Impossible Not	Not Necessary
Not Possible Not	Impossible Not	Necessary

What this shows is that Aristotle made the first observations of equivalence among negation of statements. Then Aristotle stated that each affirmative can be converted into a negative by opposition. What this means is that we can convert the statement "it is possible to belong" into "it is possible to not to belong" or "it is possible to belong to some" into "to some it is possible not to belong". Bocheński expands on Aristotle's observations with in his book *History of Formal Logic* using modern logic to explain his theorems. An example of the conversion is that Bocheński took three sentences: a) 'A possibly belongs to B', b) 'A does not possibly belong to B', and c) 'A possibly does not belong to B'. Bocheński concludes, "(b) is the proper denial of (a), (c) is no denial of (a) but it is 'negative in form'. Then it is stated that sentences such as (a) imply those such as (c) and are even equivalent to them." (Bocheński, 1965, pg. 85). What Bocheński is stating that the statements (a) and (c) are equivalent and you can convert statement (a) into statement (c). So, these statements are like how De Morgan's Law work.

Augustus De Morgan was born on June 27, 1806 in Madura, India and later died on March 18, 1871 in London England. In 1828 he became a professor of mathematics at the University in London. In 1838 he introduced and defined the term "mathematical induction" which was one of various methods of proofs of mathematical propositions. Now with the history of logic that began in ancient times and with the help of Aristotle, De Morgan investigated these laws and was able to express them mathematically. Later on De Morgan's findings were also influenced by a british mathematician known as George Boole. Boole was one who cemented modern symbolic logic with logic of algebra which is called boolean algebra. In 1847 Boole published *Mathematical Analysis of Logic* which he argued that logic should be based with mathematics and not philosophy.

De Morgan did not formally introduce De Morgan’s Law. The reason why he is credited in the name of the law is that he set up the foundation of the law to be founded. With the influence of Boole, De Morgan went back into Aristotle’s work and make his theorems into modern logic. In one of De Morgan’s work *Syllabus of a Proposed System of Logic,* He goes into great detail to express Aristotle’s syllogism into mathematical notations. An example of this conversion of statements into mathematical statements comes from a table in his book :

Symbol	Metaphysical Reading	Arithmetical reading in intension
X))Y	X dependent of Y	All qualities of Y are some qualities of X
X(•(Y	X independent of Y	Some qualities of Y not any qualities of X
X)•(Y	X repugnant of Y	All things want either some qualities of X or some qualities of Y
X( )Y	X irrepugnant of Y	Some things want neither any quality of X, nor any quality of Y
X(•)Y	X alternative of Y	All things have either all the qualities of X or all the qualities of Y
X)(Y	X inalternative of Y	Some things want either some of the qualities of X, or some of the qualities of Y
X((Y	X essential of Y	Some qualities of Y are all the qualities of X
X)•)Y	X inessential of Y	Any qualities of Y are not some qualities of X

(De Morgan, 59)

When looking through De Morgan’s work, We found an interesting equation in his book *Trigonometry and Double Algebra.* In this work, De Morgan wants current algebra to be in double algebra since current algebra can express certain terms that are illogical like -1 . Later on in the book, we see a equation dealing with logarithm be defined as A + B = (AB)(De Morgan, 129). This is like our equation with De Morgan’s Law: p q = (p q)and could be an influence for this law. It is uncertain who exactly came up with the laws we use today, but De Morgan and others past him have influenced the work to achieve these laws.

<u>Practical Use:</u>

Using De Morgan’s Law in programming development can prove to be useful. If-then statements are used widely throughout software development languages. In the language Java, for example, “!” represents *NOT,* “&&” represent *AND*, “||” *represents OR*. Then you have your arithmetic comparisons such as greater than (>), greater than or equal to (>=), less than (<), and less than or equal to (<=) (Kesavan V., 2018). We can use the example directly from Kesavan’s article *De Morgan’s law and its applications*, where the author converts the if-statement code segment:

```
if ( !(x>5 && x<8) ){
```

```
    //Do something
}
```

Currently the if-statement considers all real numbers that are not 6 or 7 in order to execute the lines of code inside the if-statement. Using De Morgan's law we can write the equivalent statement of the condition of the if-statement to:

```
if ( (x<=5 || x>=8) ){
  //Do something
}
```

In order for the if-statement to be true, the variable x is to be a real number that is not 6 or 7 which is the same the previous condition.

Another example, using direct boolean variables (true or false values) in if-then statements can be simplified using De Morgan's law and other logical identities. If we take two boolean variables, x and y. We can apply these variables to an if-then statement:

```
if( !(x && y) ){
  //Do something
}
```

Using De Morgan's law, the statement !(x && y), "NOT (x AND y)", is logically equivalent to !x || !y, "NOT x OR NOT y." A truth table is shown to test the equivalences of the two if-then statements.

x	y	!(x && y)	!x || !y
T	T	F	F
F	T	T	T
T	F	T	T
F	F	T	T

<u>Conclusions:</u>

While Augustus De Morgan was not the first person to observe the properties of what is called the De Morgan's he was the first person to express the relations between negation and not negation, in today's propositional logic. With the help from previous logical thinkers such as Aristotle, William Ockham, and George Boole, De Morgan was able to concisely represent this this relationship in a manner that has held through almost two centuries. The De Morgan's law have been used in different aspect of Computer Engineering alone, whether it comes to making logical circuits less complex and more efficient, or making a particular program more legible. With different facets considered, the De Morgan's law has been a key property in making our current technological advancements possible by simplifying more complex logical expressions.

References

Bochenski, I. (1961). *A History of Formal Logic*. Retrieved from University of Florida Libraries: https://ia600205.us.archive.org/1/items/historyofformall00boch/historyofformall00boch.pdf

Britannica, T. E. (2018). *Augustus De Morgan*. Retrieved from https://www.britannica.com/biography/Augustus-De-Morgan

Hayes, A. (n.d.). *De Morgan's Laws*. Retrieved from https://brilliant.org/wiki/de-morgans-laws/

Kesavan, V. (2014, August 18). *De Morgan's law and its applications.* Retrieved November 7, 2018, from https://mathdeconstructed.wordpress.com/2014/08/18/de-morgans-laws-and-its-applications-2/

Morgan, S. E. (2014). *Augustus De Morgan*. Retrieved from http://www.nndb.com/people/437/000097146/

De Morgan, A. (1860). *Syllabus of a Proposed System of Logic*. Retrieved from https://books.google.com/books/about/Syllabus_of_a_Proposed_System_of_Logic.html?id=Od3jgF5rZtgC&printsec=frontcover&source=kp_read_button#v=onepage&q&f=false

De Morgan, A. (1849). *Trigonometry and Double Algebra*. Retrieved from https://books.google.com/books/about/Trigonometry_and_Double_Algebra.html?id=699Vt2hyoNwC&printsec=frontcover&source=kp_read_button#v=onepage&q&f=false

Taylor, C. (2017, May 15). *What are De Morgan's Laws*. Retrieved from https://www.thoughtco.com/what-are-de-morgans-laws-3953524

Kuphaldt, T. (2013). *Vol.IV Lessons in Circuts* Retrieved from https://www.allaboutcircuits.com/textbook/digital/chpt-7/demorgans-theorems/