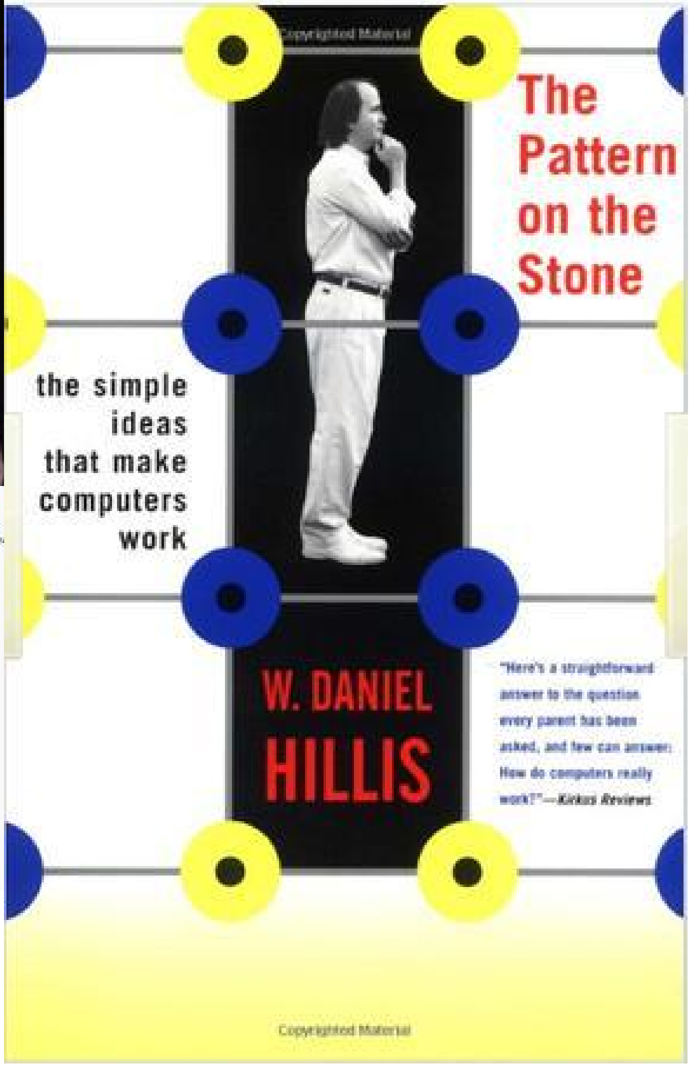
离散数学
Discrete Mathematics

第12讲 格与布尔代数 Lattice and Boolean Algebra

Boole is the inventor of Boolean logic, which is the basis of modern digital computer logic, thus Boole is regarded in hindsight as a founder of the field of computer science.

William Daniel Hillis

Born	September 25, 1956 (age 5 Baltimore, Maryland
Residence	Los Angeles
Nationality	🇺🇸 American
Fields	Computer Science Computer Engineering
Institutions	Thinking Machines Walt Disney Imagineering Applied Minds


Copyrighted Material

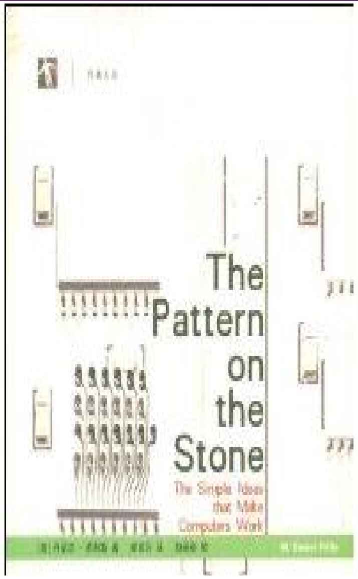
The Pattern on the Stone

the simple ideas that make computers work

W. DANIEL HILLIS

"Here's a straightforward answer to the question every parent has been asked, and few can answer: How do computers really work?" —Kirkus Reviews

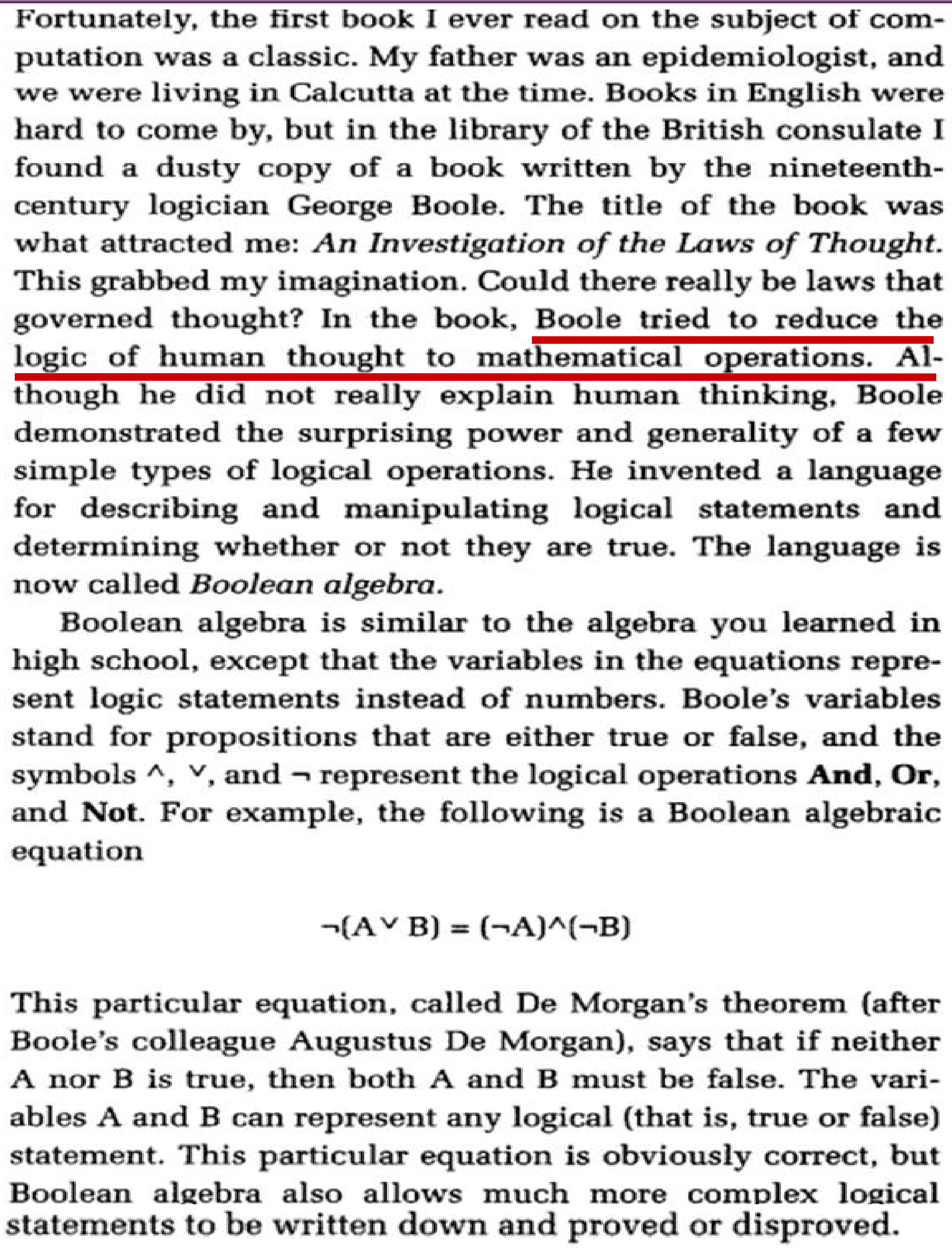
Copyrighted Material


The Pattern on the Stone

The Simple Ideas that Make Computers Work

"The work performed by the computer is specified by a program, which is written in a programming language. This language is converted to sequences of machine-language instructions by interpreters or compilers, via a predefined set of subroutines called the operating system. The instructions, which are stored in the memory of the computer, define the operations to be performed on data, which are also stored in the computer's memory. A finite-state machine fetches and executes these instructions. The instructions as well as the data are represented by patterns of bits. Both the finite-state machine and the memory are built of storage registers and Boolean logic blocks, and the latter are based on simple logical functions, such as And, Or, and Invert. These logical functions are implemented by switches, which are set up either in series or in parallel, and these switches control a physical substance, such as water or electricity, which is used to send one of two possible signals from one switch to another: 1 or 0. This is the hierarchy of abstraction that makes computers work."

——The Pattern on the Stone by Daniel Hillis

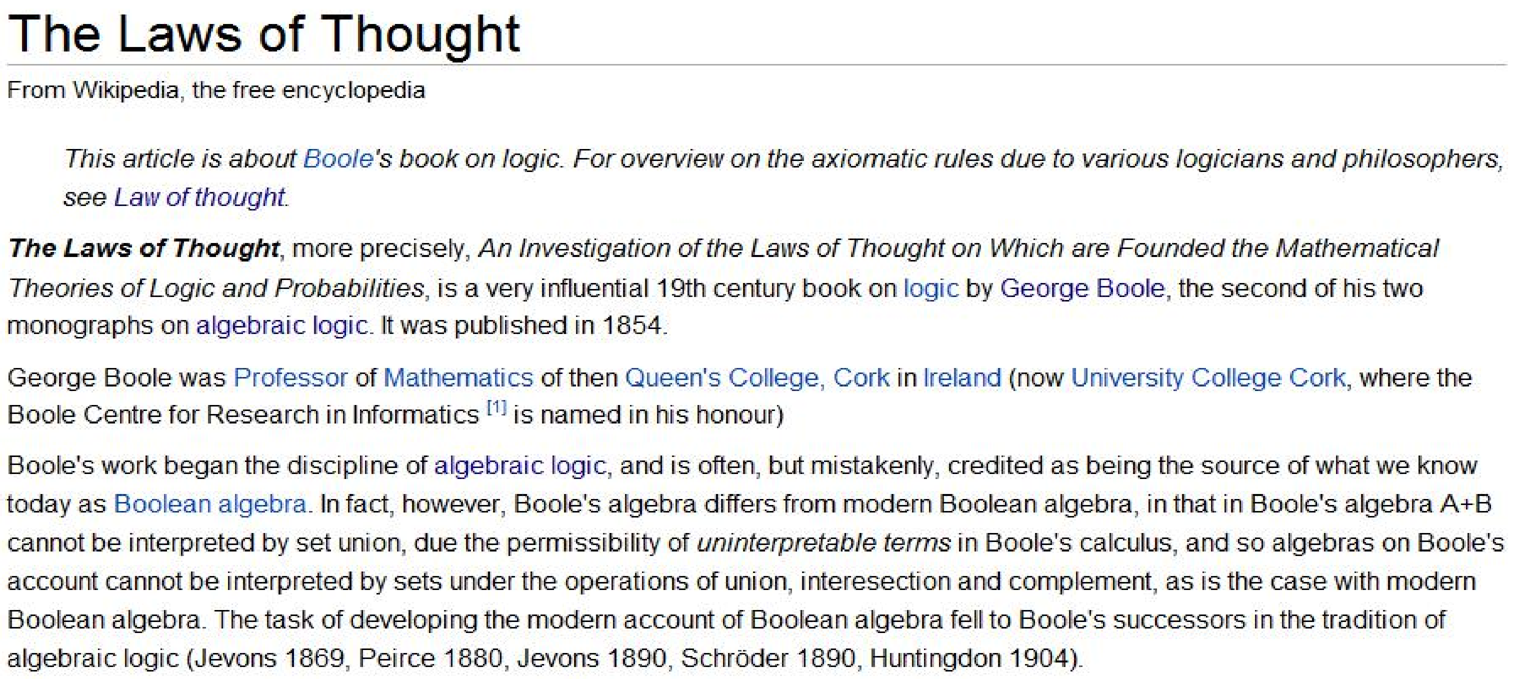


Fortunately, the first book I ever read on the subject of computation was a classic. My father was an epidemiologist, and we were living in Calcutta at the time. Books in English were hard to come by, but in the library of the British consulate I found a dusty copy of a book written by the nineteenth-century logician George Boole. The title of the book was what attracted me: *An Investigation of the Laws of Thought*. This grabbed my imagination. Could there really be laws that governed thought? In the book, Boole tried to reduce the logic of human thought to mathematical operations. Although he did not really explain human thinking, Boole demonstrated the surprising power and generality of a few simple types of logical operations. He invented a language for describing and manipulating logical statements and determining whether or not they are true. The language is now called *Boolean algebra*.

Boolean algebra is similar to the algebra you learned in high school, except that the variables in the equations represent logic statements instead of numbers. Boole's variables stand for propositions that are either true or false, and the symbols $\wedge$, $\vee$, and $\neg$ represent the logical operations **And**, **Or**, and **Not**. For example, the following is a Boolean algebraic equation

$$\neg(A \vee B) = (\neg A)\wedge(\neg B)$$

This particular equation, called De Morgan's theorem (after Boole's colleague Augustus De Morgan), says that if neither A nor B is true, then both A and B must be false. The variables A and B can represent any logical (that is, true or false) statement. This particular equation is obviously correct, but Boolean algebra also allows much more complex logical statements to be written down and proved or disproved.



The Laws of Thought

From Wikipedia, the free encyclopedia

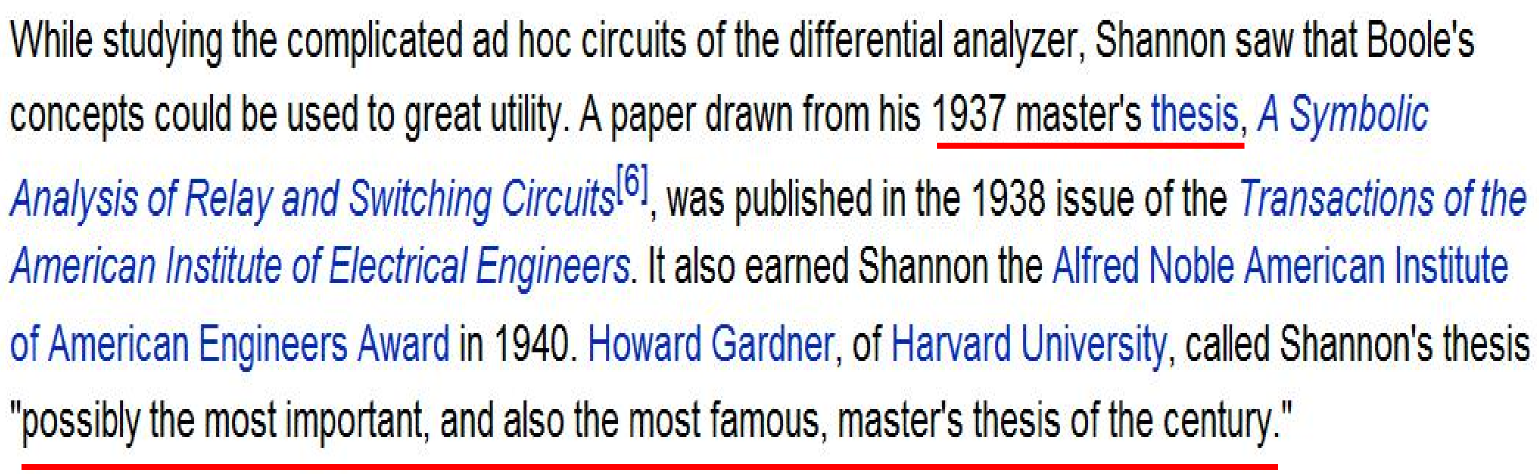
This article is about Boole's book on logic. For overview on the axiomatic rules due to various logicians and philosophers, see Law of thought.

The Laws of Thought, more precisely, *An Investigation of the Laws of Thought on Which are Founded the Mathematical Theories of Logic and Probabilities*, is a very influential 19th century book on logic by George Boole, the second of his two monographs on algebraic logic. It was published in 1854.

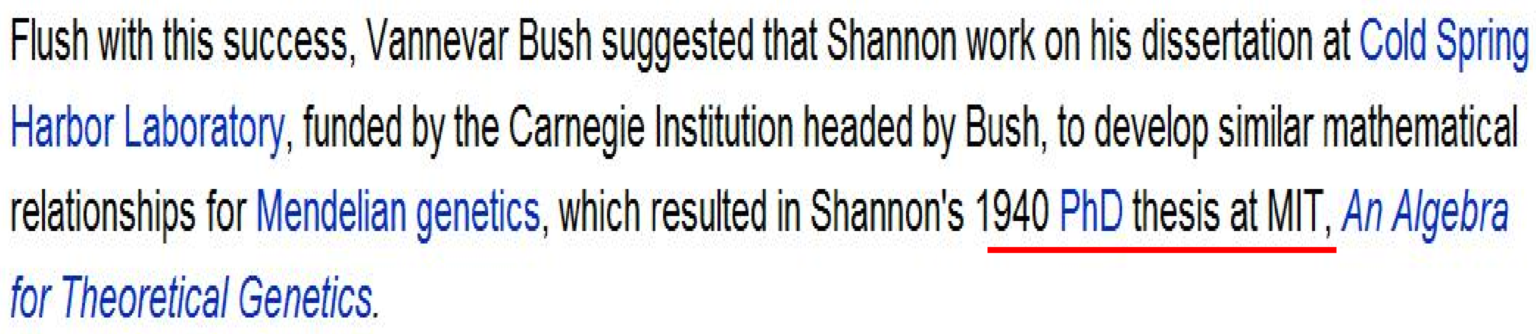
George Boole was Professor of Mathematics of then Queen's College, Cork in Ireland (now University College Cork, where the Boole Centre for Research in Informatics [1] is named in his honour)

Boole's work began the discipline of algebraic logic, and is often, but mistakenly, credited as being the source of what we know today as Boolean algebra. In fact, however, Boole's algebra differs from modern Boolean algebra, in that in Boole's algebra A+B cannot be interpreted by set union, due the permissibility of *uninterpretable terms* in Boole's calculus, and so algebras on Boole's account cannot be interpreted by sets under the operations of union, interesection and complement, as is the case with modern Boolean algebra. The task of developing the modern account of Boolean algebra fell to Boole's successors in the tradition of algebraic logic (Jevons 1869, Peirce 1880, Jevons 1890, Schröder 1890, Huntingdon 1904).

Claude Shannon

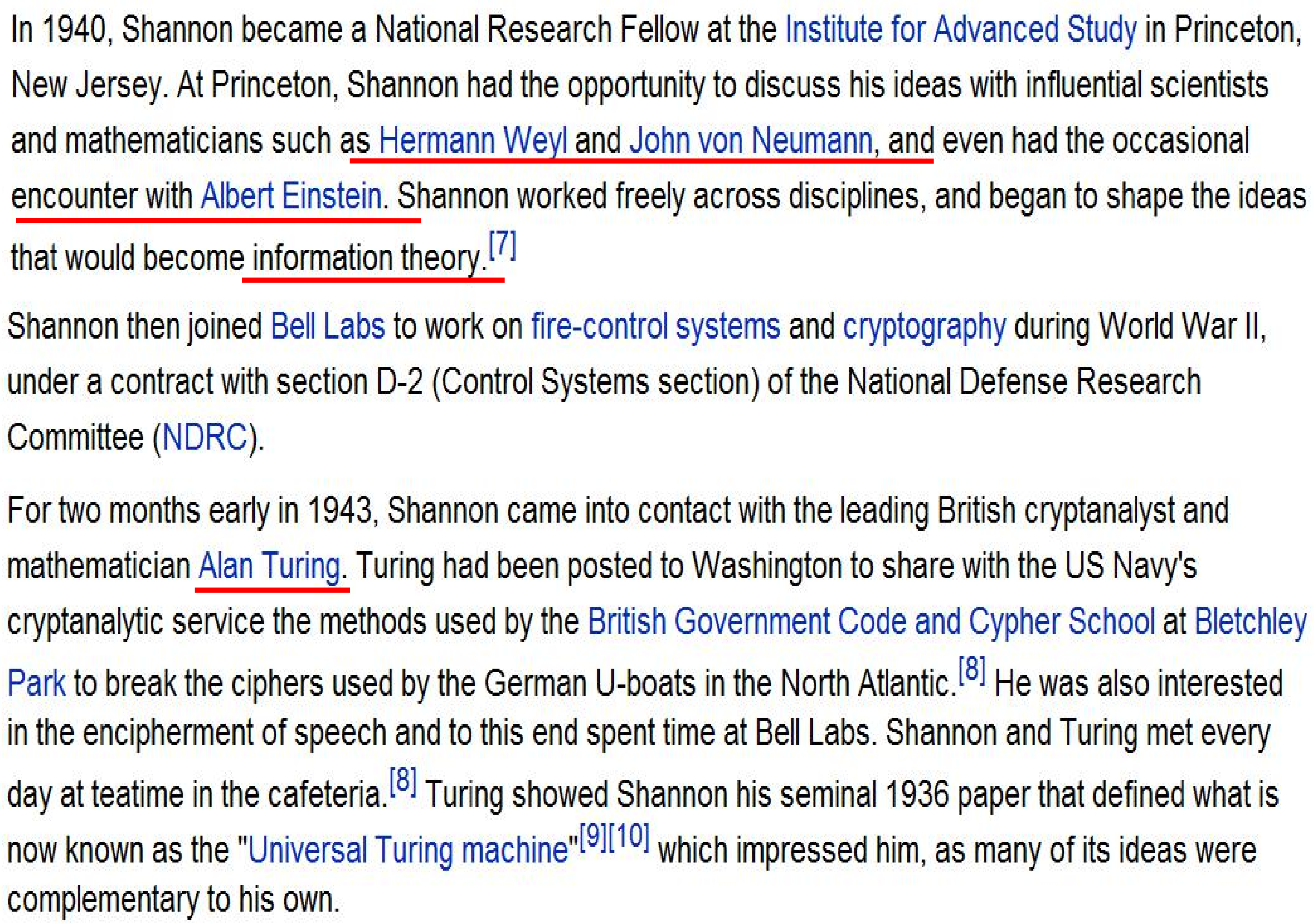


While studying the complicated ad hoc circuits of the differential analyzer, Shannon saw that Boole's concepts could be used to great utility. A paper drawn from his 1937 master's thesis, *A Symbolic Analysis of Relay and Switching Circuits*[6], was published in the 1938 issue of the *Transactions of the American Institute of Electrical Engineers*. It also earned Shannon the Alfred Noble American Institute of American Engineers Award in 1940. Howard Gardner, of Harvard University, called Shannon's thesis "possibly the most important, and also the most famous, master's thesis of the century."



Flush with this success, Vannevar Bush suggested that Shannon work on his dissertation at Cold Spring Harbor Laboratory, funded by the Carnegie Institution headed by Bush, to develop similar mathematical relationships for Mendelian genetics, which resulted in Shannon's 1940 PhD thesis at MIT, *An Algebra for Theoretical Genetics*.

Claude Shannon

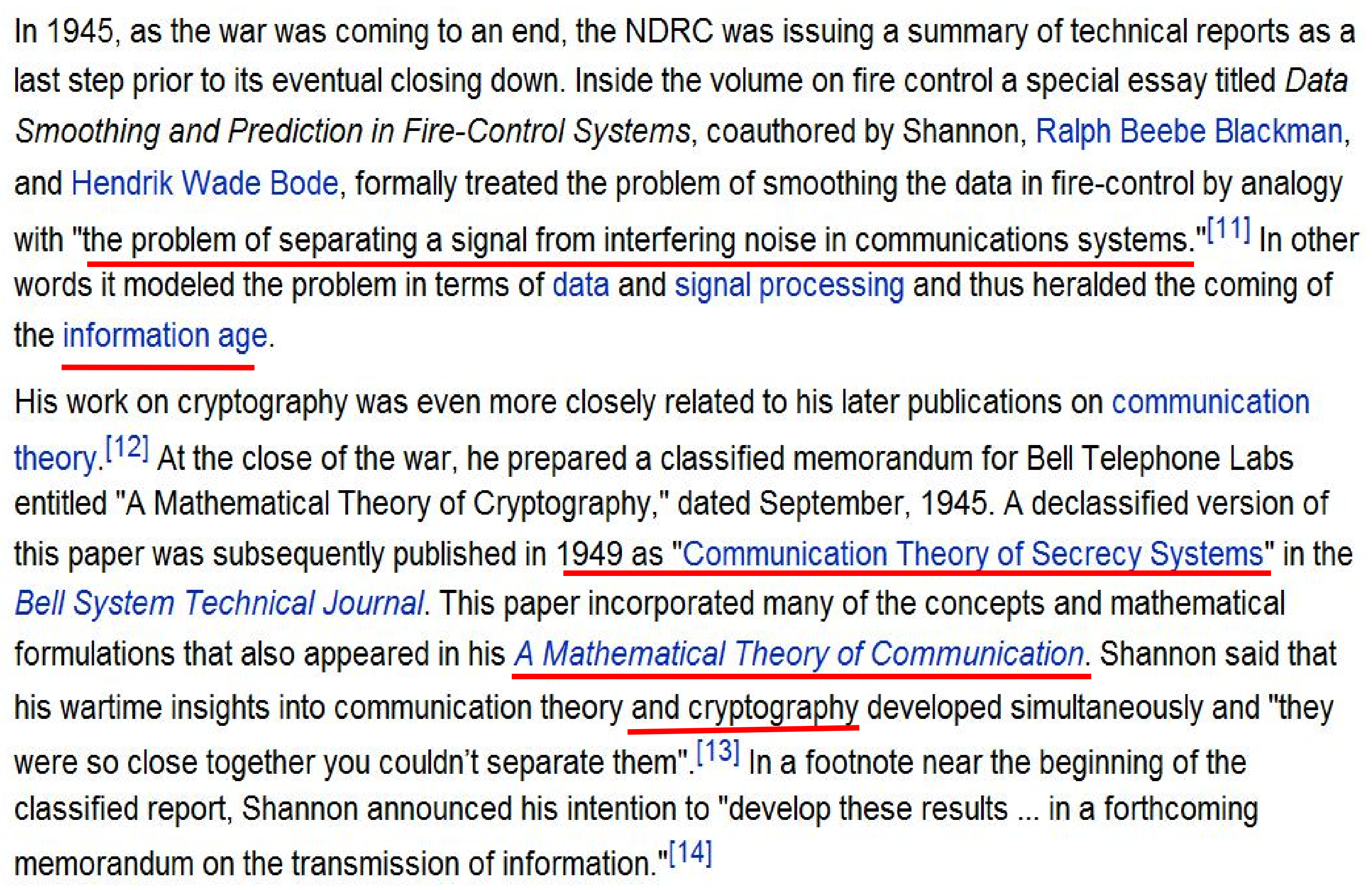


In 1940, Shannon became a National Research Fellow at the Institute for Advanced Study in Princeton, New Jersey. At Princeton, Shannon had the opportunity to discuss his ideas with influential scientists and mathematicians such as Hermann Weyl and John von Neumann, and even had the occasional encounter with Albert Einstein. Shannon worked freely across disciplines, and began to shape the ideas that would become information theory.[7]

Shannon then joined Bell Labs to work on fire-control systems and cryptography during World War II, under a contract with section D-2 (Control Systems section) of the National Defense Research Committee (NDRC).

For two months early in 1943, Shannon came into contact with the leading British cryptanalyst and mathematician Alan Turing. Turing had been posted to Washington to share with the US Navy's cryptanalytic service the methods used by the British Government Code and Cypher School at Bletchley Park to break the ciphers used by the German U-boats in the North Atlantic.[8] He was also interested in the encipherment of speech and to this end spent time at Bell Labs. Shannon and Turing met every day at teatime in the cafeteria.[8] Turing showed Shannon his seminal 1936 paper that defined what is now known as the "Universal Turing machine"[9][10] which impressed him, as many of its ideas were complementary to his own.

Claude Shannon



In 1945, as the war was coming to an end, the NDRC was issuing a summary of technical reports as a last step prior to its eventual closing down. Inside the volume on fire control a special essay titled *Data Smoothing and Prediction in Fire-Control Systems*, coauthored by Shannon, Ralph Beebe Blackman, and Hendrik Wade Bode, formally treated the problem of smoothing the data in fire-control by analogy with "the problem of separating a signal from interfering noise in communications systems."[11] In other words it modeled the problem in terms of data and signal processing and thus heralded the coming of the information age.

His work on cryptography was even more closely related to his later publications on communication theory.[12] At the close of the war, he prepared a classified memorandum for Bell Telephone Labs entitled "A Mathematical Theory of Cryptography," dated September, 1945. A declassified version of this paper was subsequently published in 1949 as "Communication Theory of Secrecy Systems" in the *Bell System Technical Journal*. This paper incorporated many of the concepts and mathematical formulations that also appeared in his *A Mathematical Theory of Communication*. Shannon said that his wartime insights into communication theory and cryptography developed simultaneously and "they were so close together you couldn't separate them".[13] In a footnote near the beginning of the classified report, Shannon announced his intention to "develop these results ... in a forthcoming memorandum on the transmission of information."[14]

——wikipedia

布尔演算（布尔代数）——数理逻辑

Huntington公理→布尔代数

格代数→布尔代数

集合代数，命题代数，开关代数均为布尔代数.

关于布尔代数(Boolean algebra)

布尔在《逻辑的数学分析》中建立了“布尔代数”原型的简化表示

含有二个运算$\vee$, $\wedge$以及一个一元运算$'$的集合, 并且有两个特殊的元素0与1, 该集合中任意元素x, y, z, 满足如下定律:

1) 结合律: $x\vee(y\vee z)=x\vee(y\vee z)$; $x\wedge(y\wedge z)=x\wedge(y\wedge z)$

2) 交换律: $x\vee y=y\vee x$; $x\wedge y=y\wedge x$

3) 分配律: $x\vee(y\wedge z)=(x\vee y)\wedge(x\vee z)$; $x\wedge(y\vee z)=(x\wedge y)\vee(x\wedge z)$

4) 同一律: $x\vee 0=x$; $x\wedge 1=x$

5) 互补律: $x\vee x'=1$; $x\wedge x'=0$

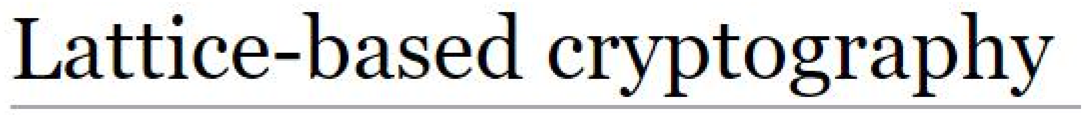
事实上, 上述代数结构还满足包括De.Morgan律在内的很多性质, 但它们(包括结合律)都可以从2)-5)四条定律推导出来, 而且可以证明2)-5)这四条定律是相互独立的。

布尔代数

一个代数结构含有二个运算$\wedge$, $\vee$以及一个一元运算$'$的集合, 并且有两个特殊的元素0与1, 如果该代数结构满足上述2)-5)四条定律。以这四条定律为公理定义布尔代数的方法称Huntington 公理方法。

其中, 运算“$\wedge$”、“$\vee$”也可以用+, *来表示, 可以读作“加”或“合取”、“乘”或“析取”, 运算“$'$”读作“补”。

关于格(Lattice)



Lattice-based cryptography

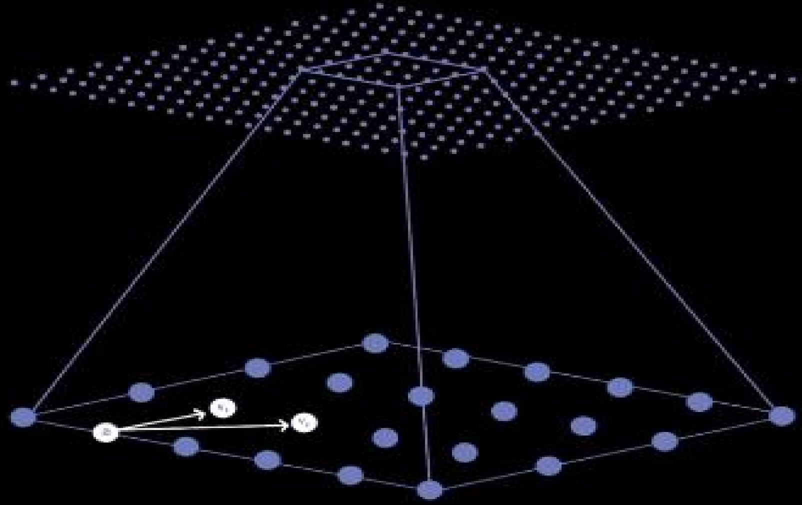
Lattice Cryptography

Lattice cryptography uses high-dimensional geometric structures to hide information, creating problems that are considered impossible to solve without the key – even by universal fault-tolerant quantum computers.

The Inner Workings of a Potentially Unhackable Cryptographic System

Why are Lattices Interesting to Cryptographers?

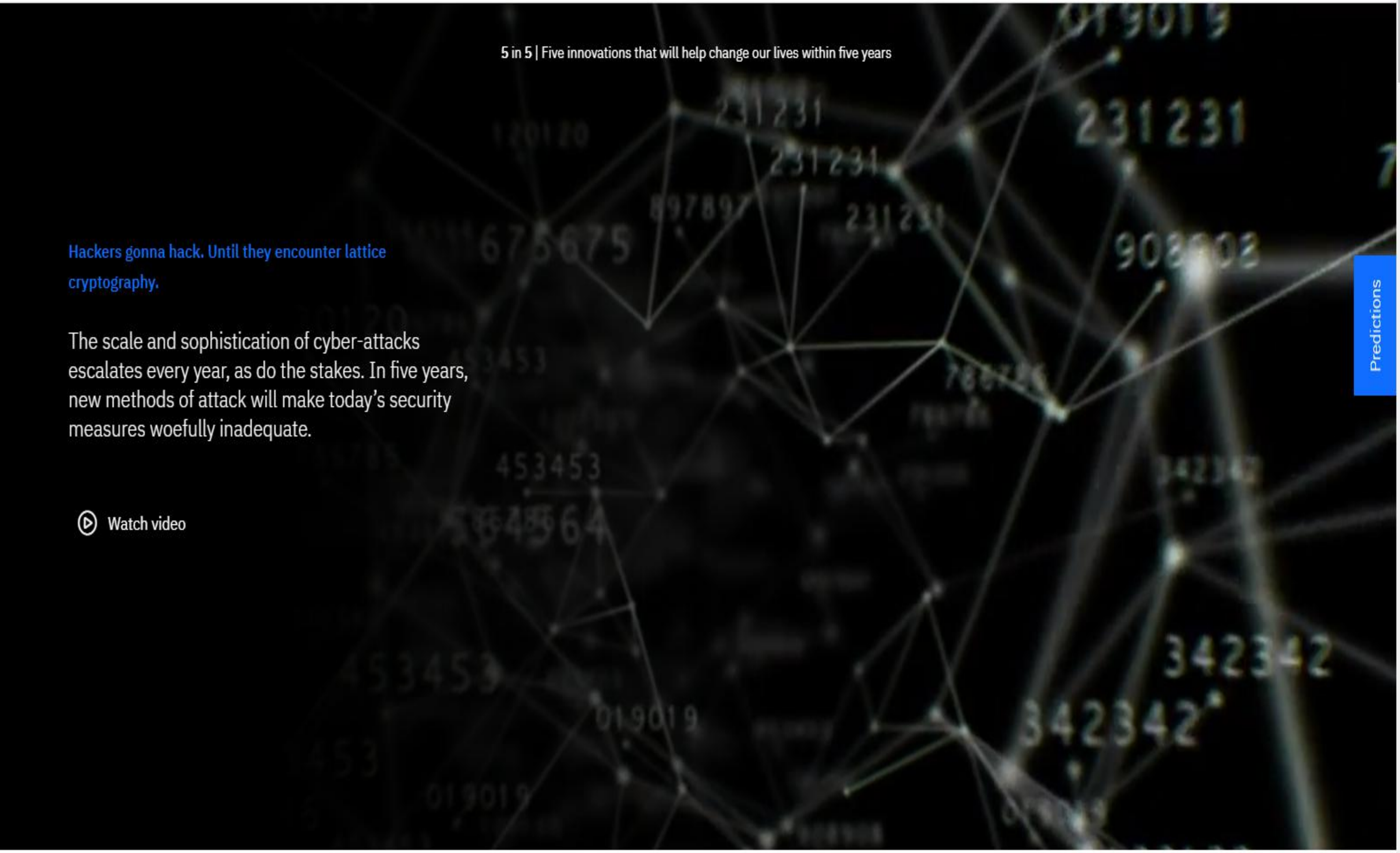
There are certain problems concerning lattices that are considered very hard to solve. Cryptographers use the hardness of solving such problems, or their inherent intractability, to build cryptographic systems.



IBM X-Force Command Center

IBM

IBM Research Research areas Work with us About us Blog



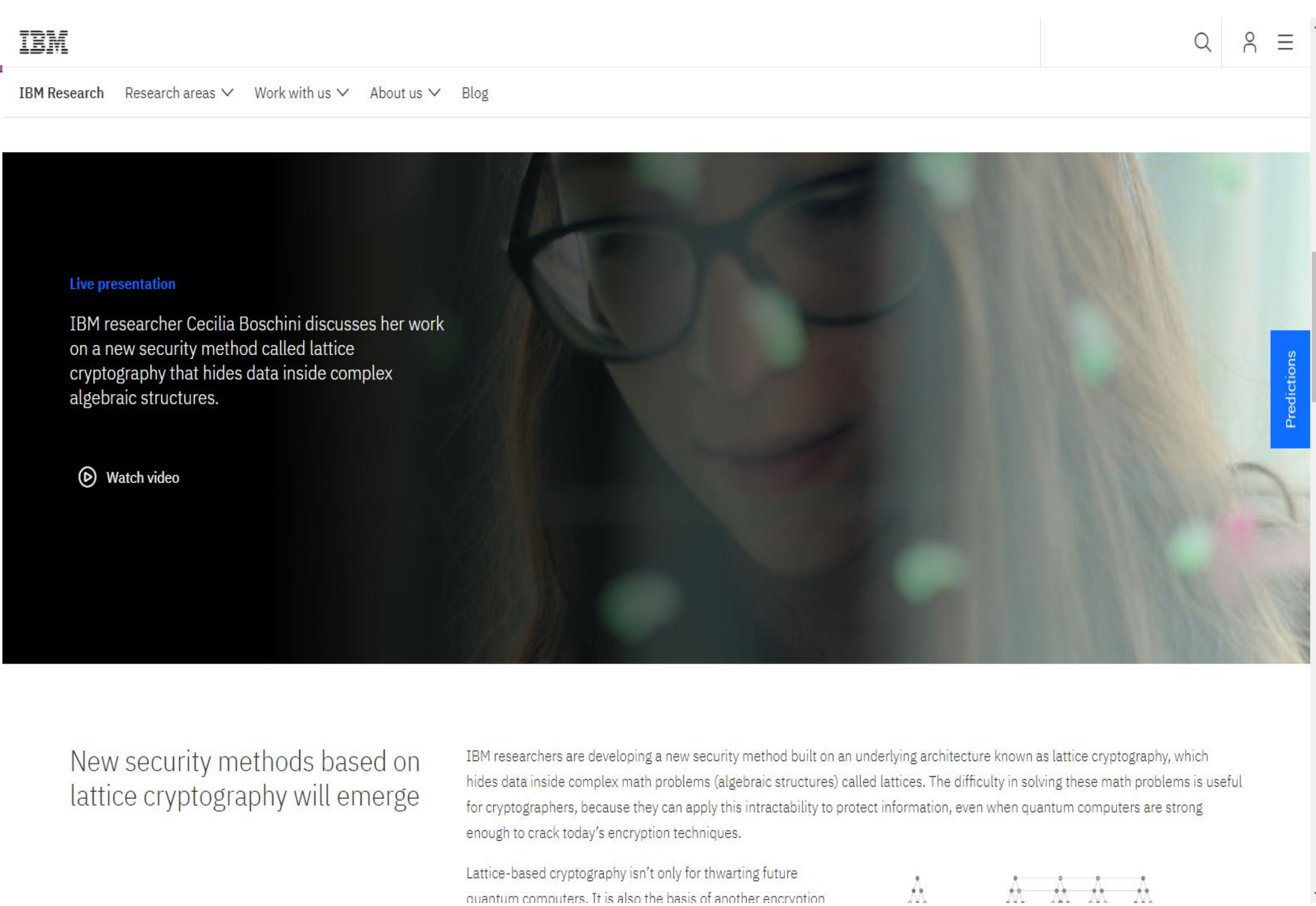
5 in 5 | Five innovations that will help change our lives within five years

Hackers gonna hack. Until they encounter lattice cryptography.

The scale and sophistication of cyber-attacks escalates every year, as do the stakes. In five years, new methods of attack will make today's security measures woefully inadequate.

Watch video

Predictions



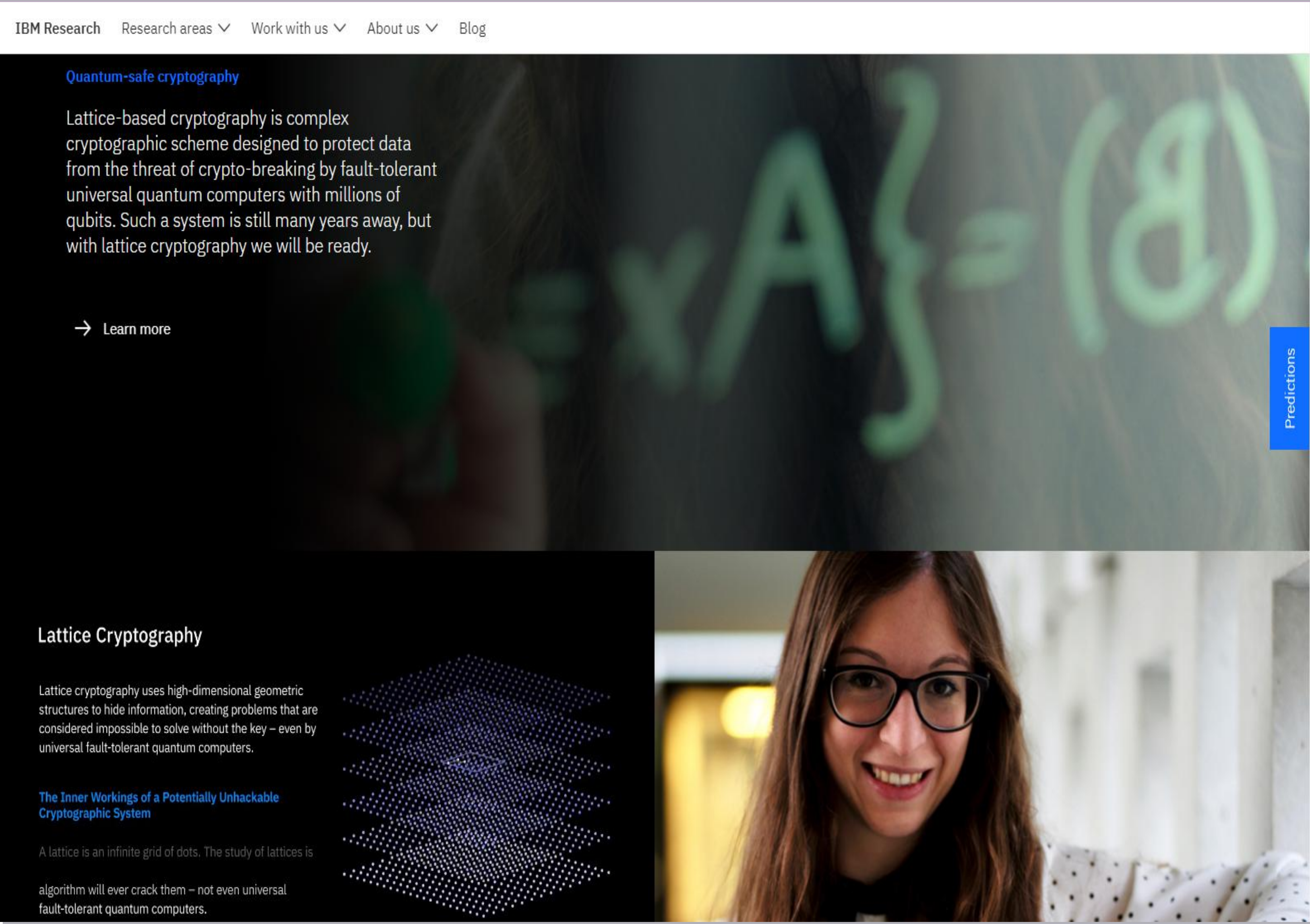
Live presentation

IBM researcher Cecilia Boschini discusses her work on a new security method called lattice cryptography that hides data inside complex algebraic structures.

New security methods based on lattice cryptography will emerge

IBM researchers are developing a new security method built on an underlying architecture known as lattice cryptography, which hides data inside complex math problems (algebraic structures) called lattices. The difficulty in solving these math problems is useful for cryptographers, because they can apply this intractability to protect information, even when quantum computers are strong enough to crack today's encryption techniques.

Lattice-based cryptography isn't only for thwarting future quantum computers. It is also the basis of another encryption



IBM Research   Research areas   Work with us   About us   Blog

Quantum-safe cryptography

Lattice-based cryptography is complex cryptographic scheme designed to protect data from the threat of crypto-breaking by fault-tolerant universal quantum computers with millions of qubits. Such a system is still many years away, but with lattice cryptography we will be ready.

→ Learn more

Predictions

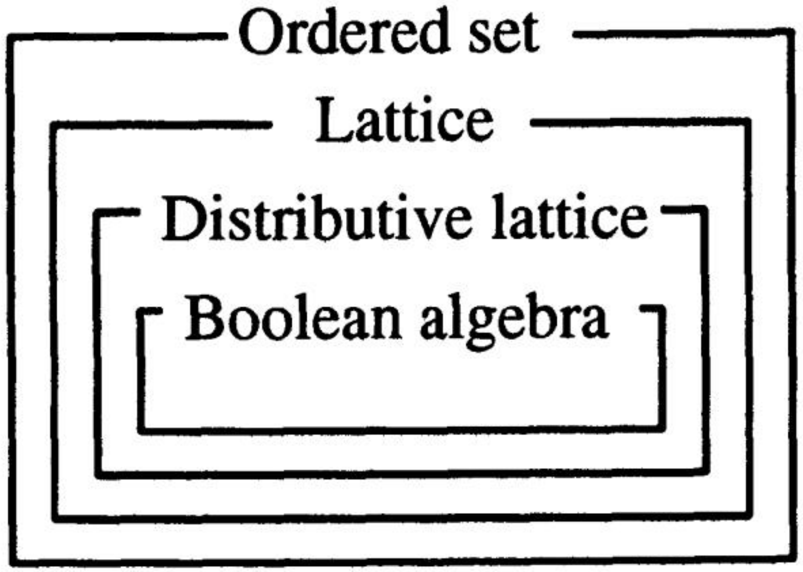
Lattice Cryptography

Lattice cryptography uses high-dimensional geometric structures to hide information, creating problems that are considered impossible to solve without the key – even by universal fault-tolerant quantum computers.

The Inner Workings of a Potentially Unhackable Cryptographic System

A lattice is an infinite grid of dots. The study of lattices is

algorithm will ever crack them – not even universal fault-tolerant quantum computers.


Ordered set
Lattice
Distributive lattice
Boolean algebra

主要内容

请回忆：偏序结构

偏序集、哈斯图

上界、最小上界

下界、最大下界

最大下界、最小上界若存在，则唯一?

格(Lattice)

设〈L，≤〉是偏序集合，若∀a，b∈L，都有最大下界、最小上界;

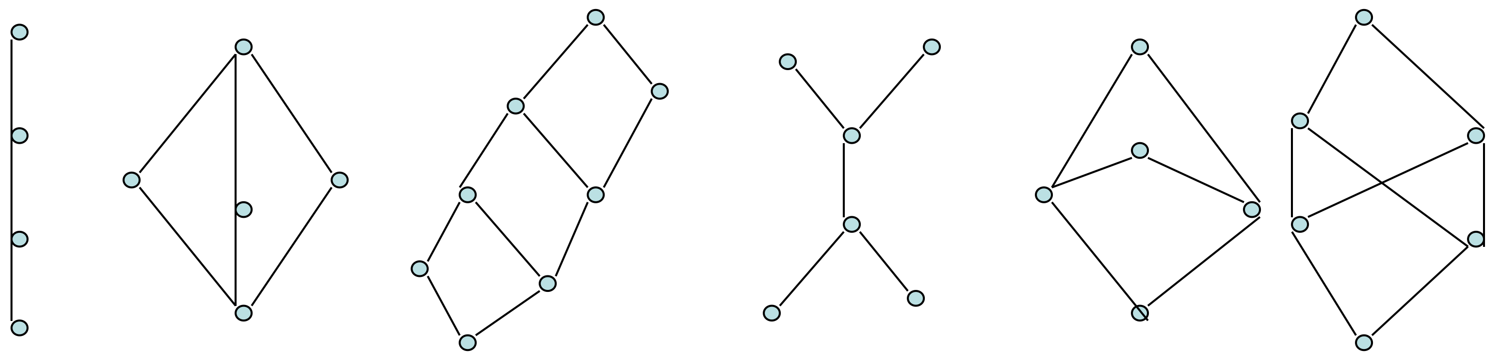
则称〈L，≤〉是格，且记glb(a,b)=a∧b,

lub(a,b)=a∨b,并称它们为交和并。

注1: 由于最大下界、最小上界若存在则唯一，所以交、并是二元运算?

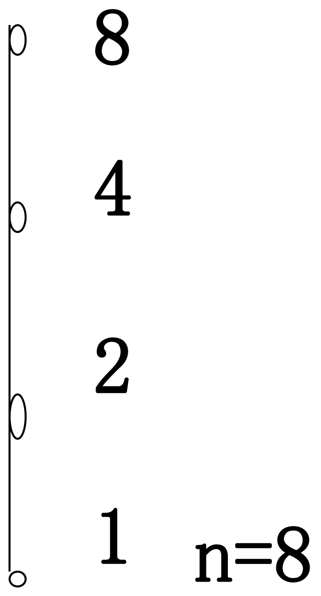
注2: 称<L; ∧,∨>为由格〈L, ≤〉所诱导的代数系统。

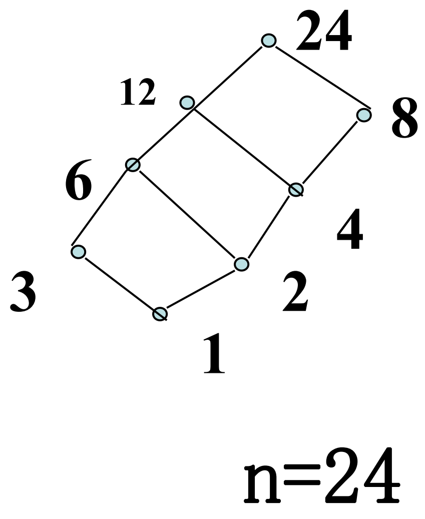
示例



示例

设n是一正整数，Sn是n的所有因子的集合，D是整除关系,则<Sn,D>是格。

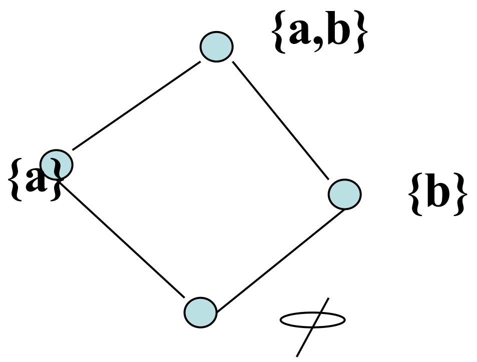


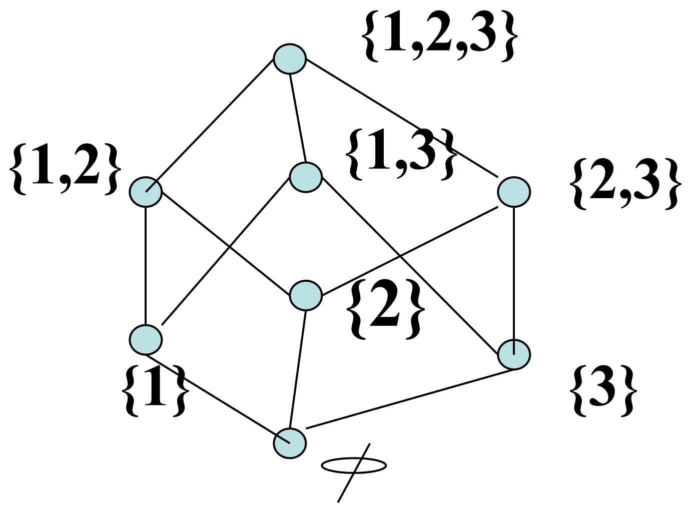


示例

设S是任意集合，P(S)是幂集，偏序集合<S, ⊆>是格，其中∀A,B∈P(S),A∧B=A∩B, A∨B=A∪B.

S={a,b}　　S={1,2,3}





格的基本性质

1)自反性

$a \le a$ 对偶: $a \ge a$

2)反对称性

$a \le b$ 且 $a \ge b \Rightarrow a = b$

对偶:$a \ge b$ 且 $a \le b \Rightarrow a = b$

3)传递性

$a \le b$ 且 $b \le c \Rightarrow a \le c$

对偶:$a \ge b$ 且 $b \ge c \Rightarrow a \ge c$

格的基本性质

4) 最大下界描述之一

$a \wedge b \leq a$ 对偶 $a \vee b \geq a$

$a \wedge b \leq b$ 对偶 $a \vee b \geq b$

5） 最大下界描述之二

$c \leq a, c \leq b \Rightarrow c \leq a \wedge b$

对偶 $c \geq a, c \geq b \Rightarrow c \geq a \vee b$

格的基本性质

6) 结合律

$a\wedge(b\wedge c)=(a\wedge b)\wedge c$

对偶 $a\vee(b\vee c)=(a\vee b)\vee c$

证： 令$R=a\wedge(b\wedge c)$, $R'=(a\wedge b)\wedge c$

则由4 $R\le a$, $R\le b\wedge c$

$\Rightarrow R\le a,\ R\le b,\ R\le c$

$\Rightarrow R\le a\wedge b,\ R\le c$

$\Rightarrow R\le(a\wedge b)\wedge c$

$\Rightarrow R\le R'$

同理可证：$R\ge R'$

所以 $R=R'$

注: 格的证明思路：总是利用反对称性。

格的基本性质

7)等幂律

$a \wedge a = a$ 对偶 $a \vee a = a$

8)吸收律

$a \wedge (a \vee b) = a$ 对偶 $a \vee (a \wedge b) = a$

证： 因为$a \le a \vee b$, $a \le a$

所以 $a \le a \wedge (a \vee b)$

又 $a \ge a \wedge (a \vee b)$

所以$a = a \wedge (a \vee b)$

格的基本性质

9) $a \leq b \Leftrightarrow a \wedge b = a$

$a \vee b = b$

证：$\Rightarrow$ $a \leq b$

$\therefore a \leq a,\ a \leq b$

$\therefore a \leq a \wedge b$

又 $a \geq a \wedge b$

$\therefore a = a \wedge b$

$\Leftarrow$ $a \wedge b = a$

若$b \leq a$且$b \neq a$，则$a \wedge b = b$,矛盾

若a,b不可比较，则与$a \wedge b = a$矛盾

$\therefore a \leq b$

格的基本性质

10) $a \le c, b \le d \Rightarrow a \wedge b \le c \wedge d$

$a \vee b \le c \vee d$

证：$a \wedge b \le a,\ a \le c \Rightarrow a \wedge b \le c$

$a \wedge b \le b,\ b \le d \Rightarrow a \wedge b \le d$

$\Rightarrow a \wedge b \le c \wedge d$

格的基本性质

11）保序性

$b \le c \Rightarrow\ a \wedge b \le a \wedge c$

$a \vee b \le a \vee c$

证： 因为$a \le a$, $b \le c$

由性质10) 知 $a \wedge b \le a \wedge c$

得证。

格的基本性质

12） 分配不等式

$$a\vee(b\wedge c)\leq(a\vee b)\wedge(a\vee c)$$

对偶 $a\wedge(b\vee c)\geq(a\wedge b)\vee(a\wedge c)$

证:

$a\leq a\vee b$, $a\leq a\vee c$

$\Rightarrow$ $a\leq(a\vee b)\wedge(a\vee c)$

$b\leq a\vee b$, $c\leq a\vee c$

$\Rightarrow$ $b\wedge c\leq(a\vee b)\wedge(a\vee c)$ （性质10）

$\therefore$ $a\vee(b\wedge c)\leq(a\vee b)\wedge(a\vee c)$

2 格是代数结构

将代数结构的有关概念应用于格

2 格是代数结构

偏序集（格）：代数结构?

代数结构：偏序集（格）？

设<L; ∧,∨>是代数结构，∧,∨是L上二个二元运算，若二元运算∧,∨均满足

结合律

交换律

吸收律(a∨(a∧b)=a,a∧ (a∨b)=a)

等幂律(a∧a=a,a∨a=a),

则<L; ∧,∨>是格。

注：定义中的等幂律可以不用，因它可由吸收律得到：

$a\wedge a = \underline{a}\wedge(\underline{a}\vee(a\wedge a))=a$

2 格是代数结构

偏序集合的格、代数结构的格等价

要证明代数结构$<L;\wedge,\vee>$是格，需要证明L中存在一偏序关系$\leq$,

对$\forall a,b\in L$，有

$\mathrm{lub}(a,b)=a\vee b$, $\mathrm{glb}(a,b)=a\wedge b$.

2 格是代数结构

如何定义偏序关系?

在集合L上定义二元关系≤如下:

$\forall a,b \in L$，若$a \le b \Leftrightarrow a \wedge b = a$

(或：$\forall a,b \in L$，若$a \le b \Leftrightarrow a \vee b = b$)

如何证明?

1) ‘≤’ 是L上的偏序关系?

2) $\forall a,b \in L$, $\{a,b\}$的最大下界存在，

且$glb(a,b)=a \wedge b$。

3) $\forall a,b \in L$, $\{a,b\}$的最小上界存在，

且$lub(a,b)=a \vee b$

2 格是代数结构

1) 证明'$\leq$'是L上的偏序关系

自反性:

$\forall a \in L$，由等幂律$a \wedge a = a$,有$a \leq a$

反对称性:

$\forall a,b \in L$, 若$a \leq b$, $b \leq a$，即$a \wedge b = a, b \wedge a = b$,

于是由交换律，有$a = a \wedge b = b \wedge a = b$

传递性:

$\forall a,b,c \in L$,若$a \leq b, b \leq c$，即$a \wedge b = a$, $b \wedge c = b$

于是，由$a \wedge c = (a \wedge b) \wedge c = a \wedge (b \wedge c) = a \wedge b = a$

得$a \leq c$

2 格是代数结构

2)证明 $\forall a,b\in L$,$\{a,b\}$的最大下界存在，且$glb(a,b)=a\wedge b$

i) 下界存在,其一为$a\wedge b$:

由$(a\wedge b)\wedge a=(a\wedge a)\wedge b=a\wedge b$

有：$a\wedge b\leq a$

同理，$a\wedge b\leq b$

从而，$a\wedge b$ 是a,b的下界。

ii) $a\wedge b$为最大下界:

设c 是a,b 的任一下界,

即$c\leq b$, $c\leq a$,

由$c\wedge(a\wedge b)= (c\wedge a)\wedge b=c\wedge b=c$

有：$c\leq a\wedge b$

所以，$a\wedge b$ 是a,b的最大下界。

2 格是代数结构

3) 同理可证?

$\forall a,b \in L$, $\{a,b\}$的最小上界存在，且$lub(a,b)=a \vee b$

综上1）2）3）知:$<L;\wedge,\vee>$是格。

注：以后可根据需要，随意使用这二种定义和记法

$<L,\leq>$或$<L;\wedge,\vee>$

2 格是代数结构

子格(subLattice)?

格同态?

<S;*,+>,<L;∧,∨>是格,f: L→S,

且∀a,b∈L 有 $f(a*b)=f(a)\wedge f(b)$

$f(a+b)=f(a)\vee f(b)$

称f是<L; *,+>到<S; ∧,∨>的格同态;

若f是双射,则<L;*,+>和<S;∧,∨>是同构的

格同态的保序性：

即：f是格$<L;*,+>$到$<L';\wedge,\vee>$的同态映射，

若$a,b\in L$，$a\leq b$ 则$f(a)\leq' f(b)$

证：$a\leq b \Leftrightarrow a*b=a$

$$f(a)\wedge f(b)=f(a*b)=f(a)$$

$$f(a)\leq' f(b)$$

格同构的充要条件

设f是双射,则f是$<A_1,\leq>$到$<A_2,\leq'>$的格同构，当且仅当

$\forall a,b \in A_1, a\leq b \Leftrightarrow f(a)\leq' f(b)$

- 分配格、模格、有补格、有补分配格
- 布尔代数

格的最大元、最小元，分别记为1、0；→有界格

✓1、0是唯一的吗？

✓0是∨的单位元，1是∧的单位元？

示例

设S=｛2，3，4，…，100｝，对于普通的数之间的小于或等于关系≤，〈S，≤〉是一个格，其最大元素1=100，最小元素0=2。

补元

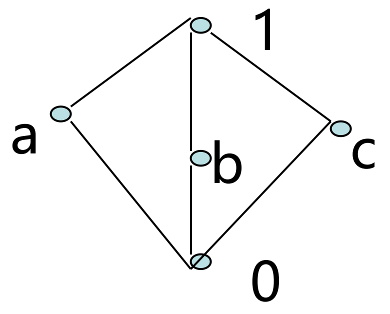
设$<L,\wedge,\vee,0,1>$是有界格，$a\in L$，若存在$b\in L$，有$a\wedge b=0, a\vee b=1$，则称b为a的补元，记为a′。

1 若a是b的补元，则b也是a的补元?

2 有界格中，0，1互为补元?

3 补元可以不存在，可以不唯一?

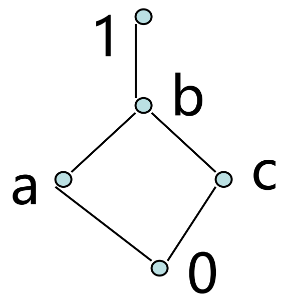
示例



a) a的补元是b,c，

b的补元是a,c,

c的补元是a,b



b) a,b,c 均不存在补元

3 布尔代数

有补格

若在一个有界格中，每个元素至少有一个补元，则称此格为有补格。

有补分配格

若一个格既是有补格，又是分配格，则此格称为有补分配格，又称布尔代数(Boolean Algebra) 。

3 布尔代数

示例：布尔代数

集合代数，命题代数，开关代数为布尔代数

(因为满足结合律、交换律、吸收律、分配律、存在补元及0，1)

1) 集合代数 $<P(S), \cap, \cup, \sim, \varnothing, S>$ 是布尔代数

2) 开关代数 $<\{0,1\}, \wedge, \vee, \neg, 0, 1>$ 是布尔代数,其中 $\wedge$ 为与运算, $\vee$ 为或运算, $\neg$ 为非运算.

十条定律

设<S;∨,∧,->是一个布尔代数,a,b,c是S中任意三个元素，在S中成立以下算律。

1) 交换律	2) 结合律
3) 等幂律	4) 吸收律
5) 分配律	6) 同一律
7) 零一律	8) 互补律
9) 对合律	10) 德·摩根律

以上十条定律并不都是独立的, 均可由交换律、分配律、同一律和互补律得到。

几个结论

Bool(n)中任意函数可以表示为多个Bool(n)的原子(也称为基本积或最小项)的∨运算；

布尔代数的每一子代数仍是布尔代数；

一个布尔代数的每一满同态像均是布尔代数；

有限布尔代数与某集合代数同构。

小结

1）格;

2）格与代数结构

3）布尔代数;