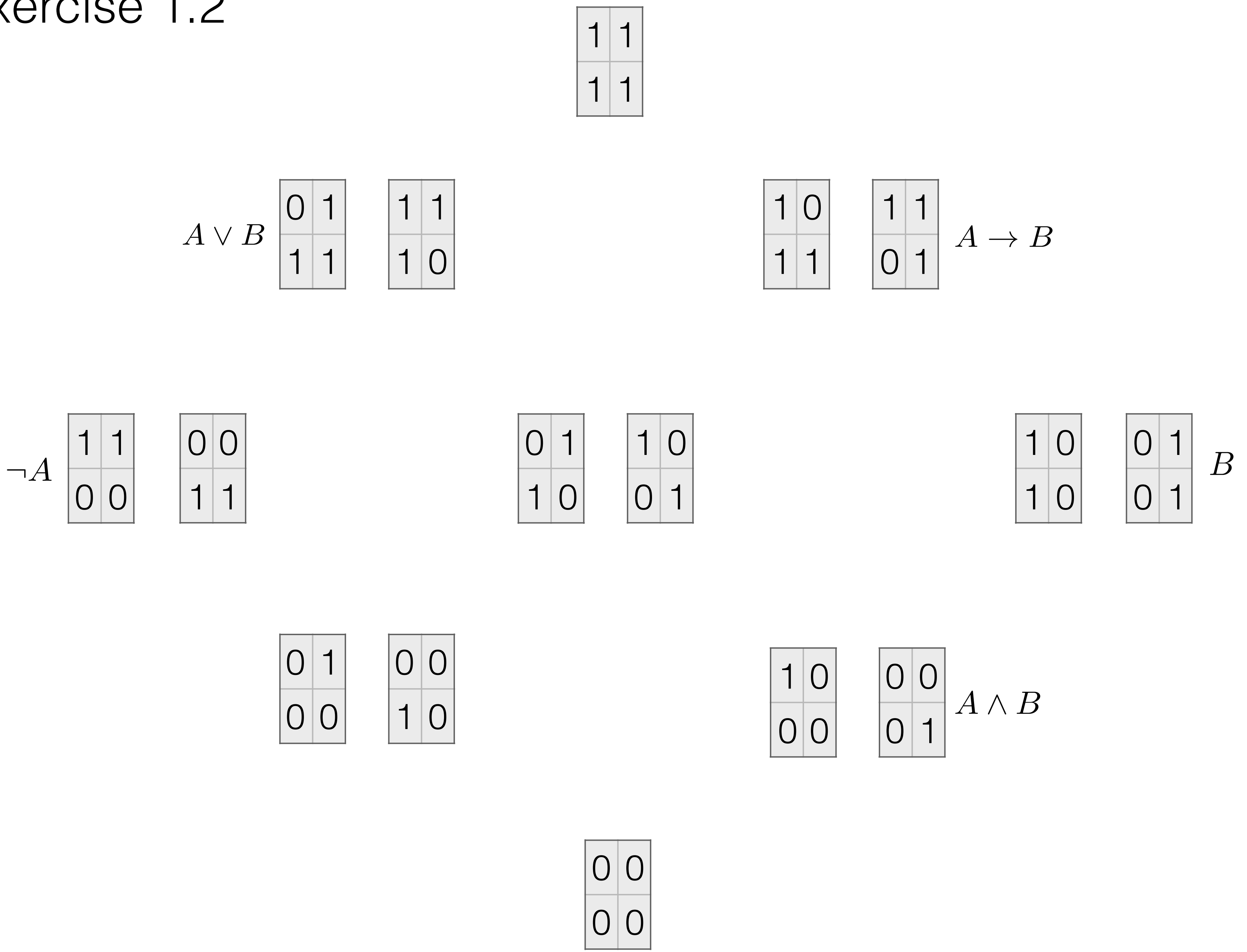
Informatics 1

Computation and Logic

Karnaugh Maps

Michael Fourman

Exercise 1.2



Exercise 1.3

$\top$: $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$A \vee B$: $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$

$\neg(A \wedge B)$: $\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$

$B \rightarrow A$: $\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$

$A \rightarrow B$: $\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$

$\neg A$: $\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$

A: $\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$

$A \oplus B$: $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

$A \leftrightarrow B$: $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$\neg B$: $\begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$

B: $\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$

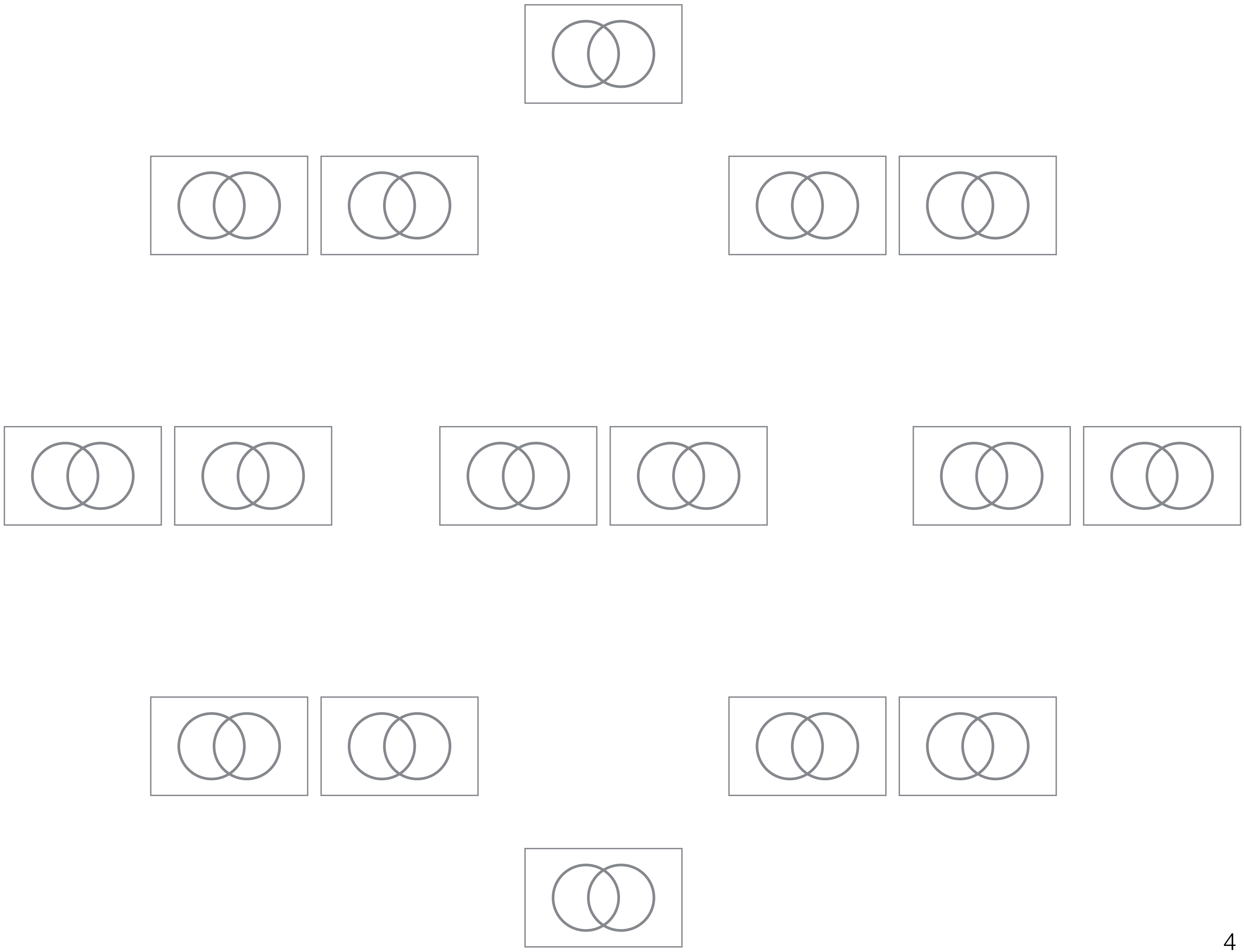
$\neg(B \rightarrow A)$: $\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$

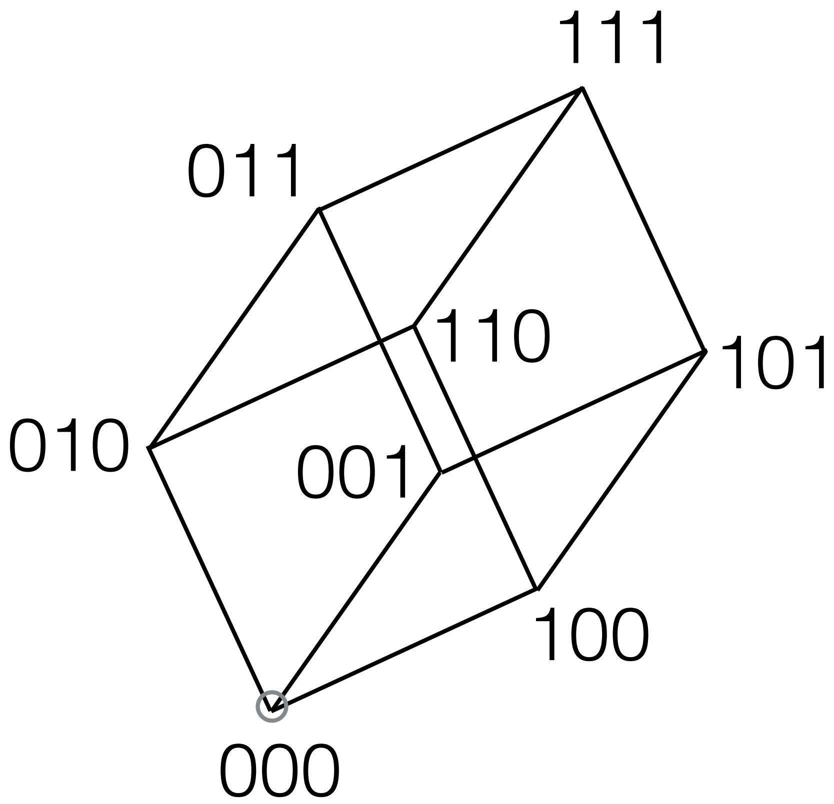
$\neg(A \rightarrow B)$: $\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$

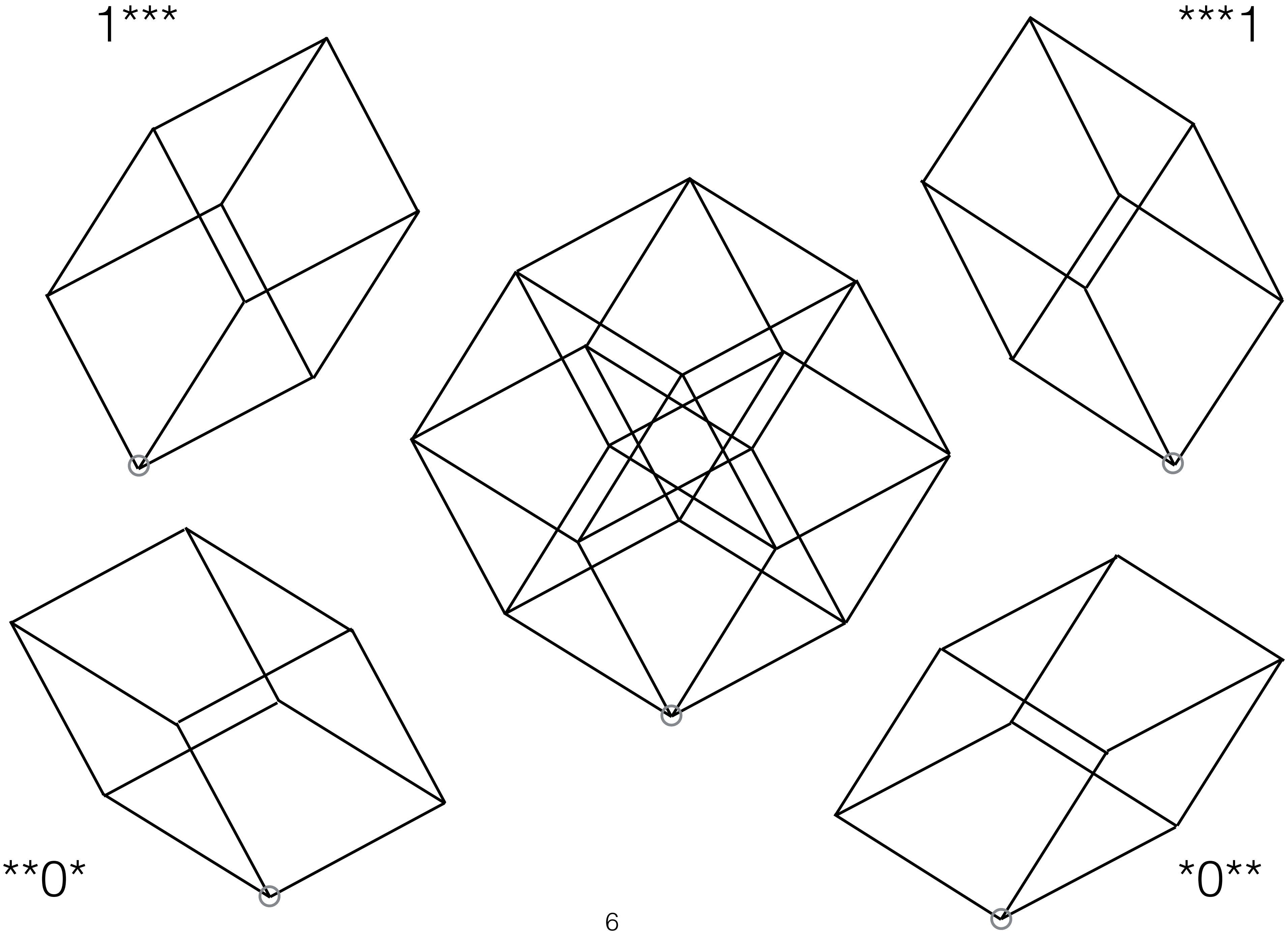
$\neg(A \vee B)$: $\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$

$A \wedge B$: $\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$

$\bot$: $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

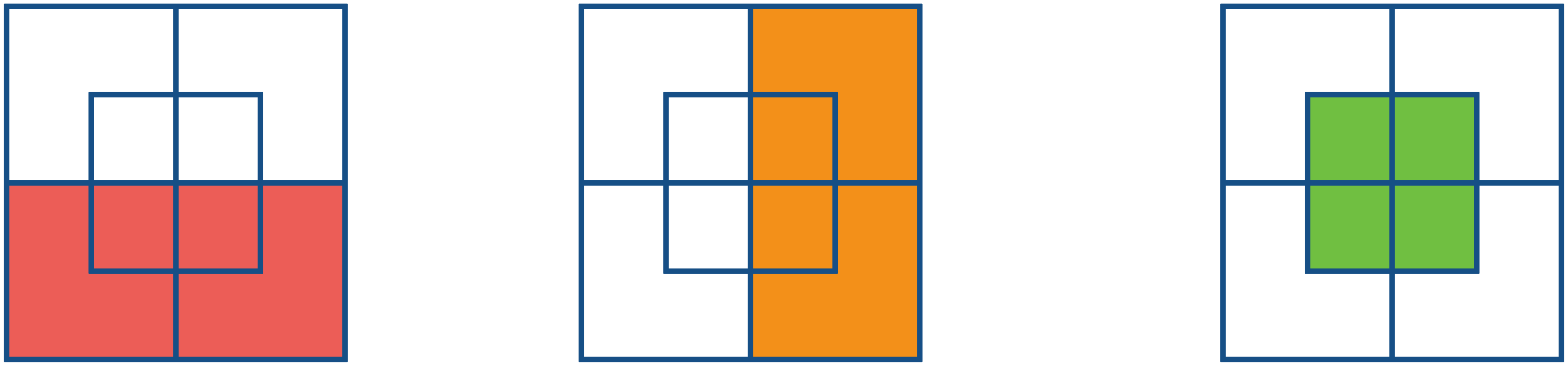



111
011
110
101
010
001
100
000

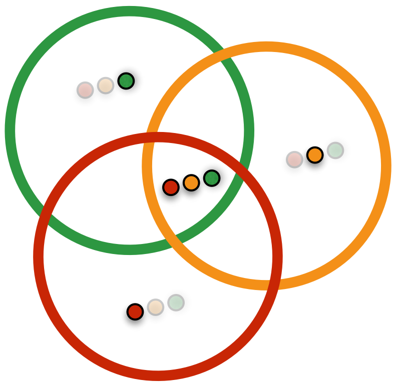

1***
***1
**0*
*0**

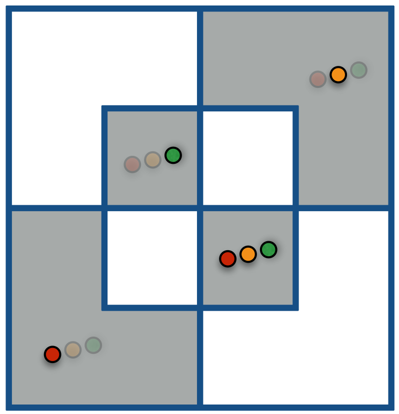
Lewis Carroll (The Rev. C.L. Dodgson)

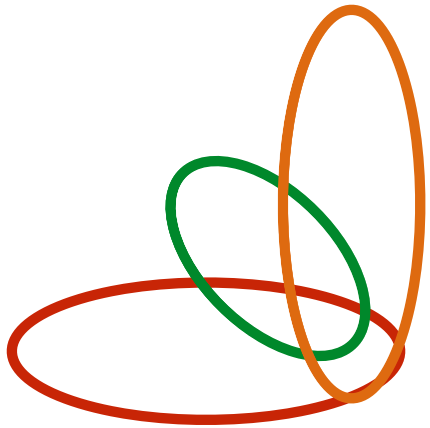
R A G

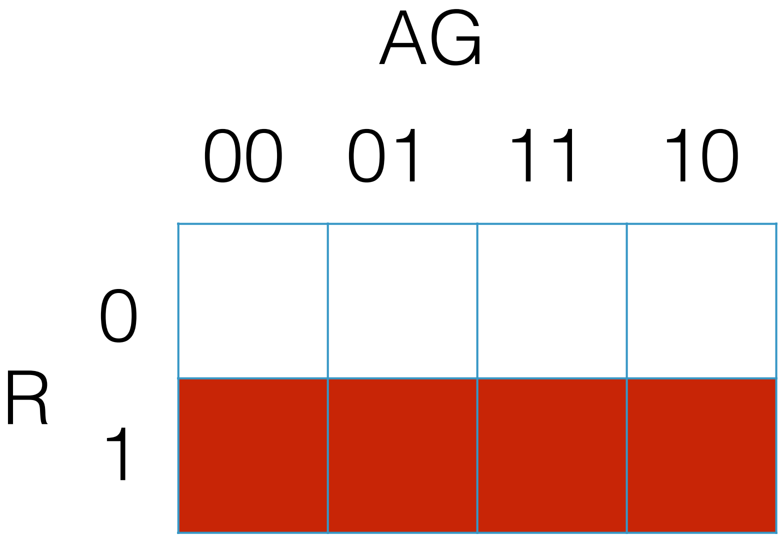


$$\{x \mid G(x) \leftrightarrow R(x) \leftrightarrow A(x)\}$$

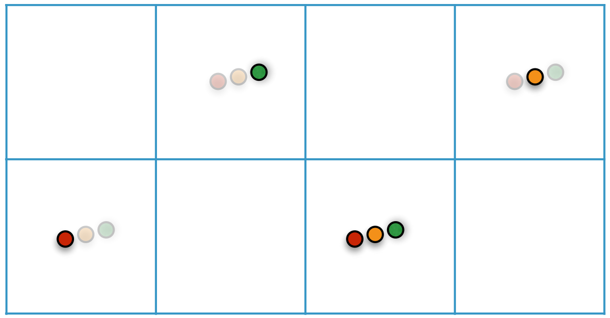


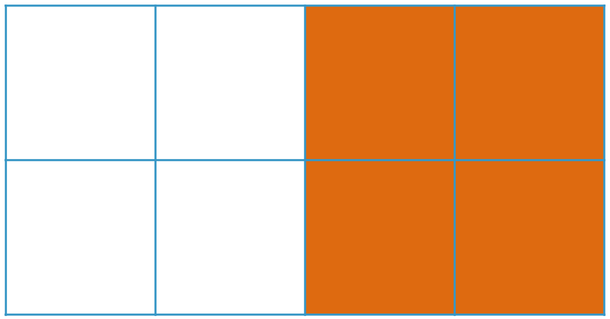


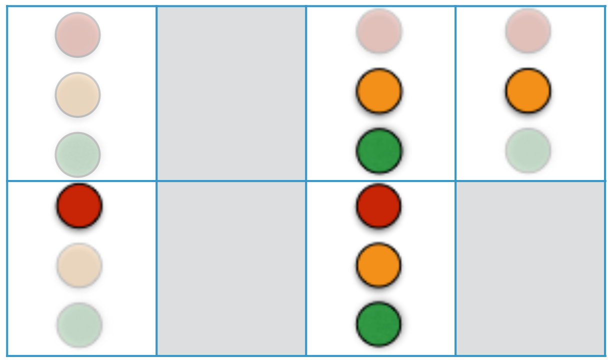


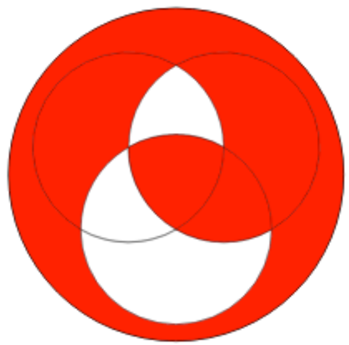


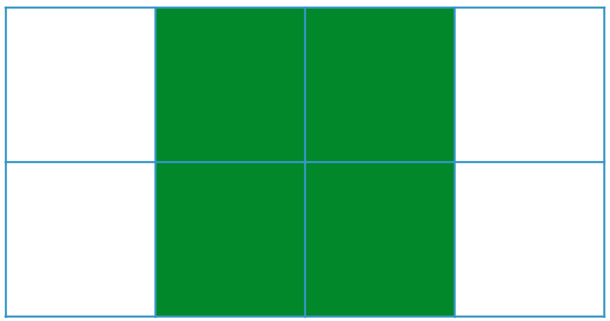
$$\{x \mid G(x) \leftrightarrow R(x) \leftrightarrow A(x)\}$$



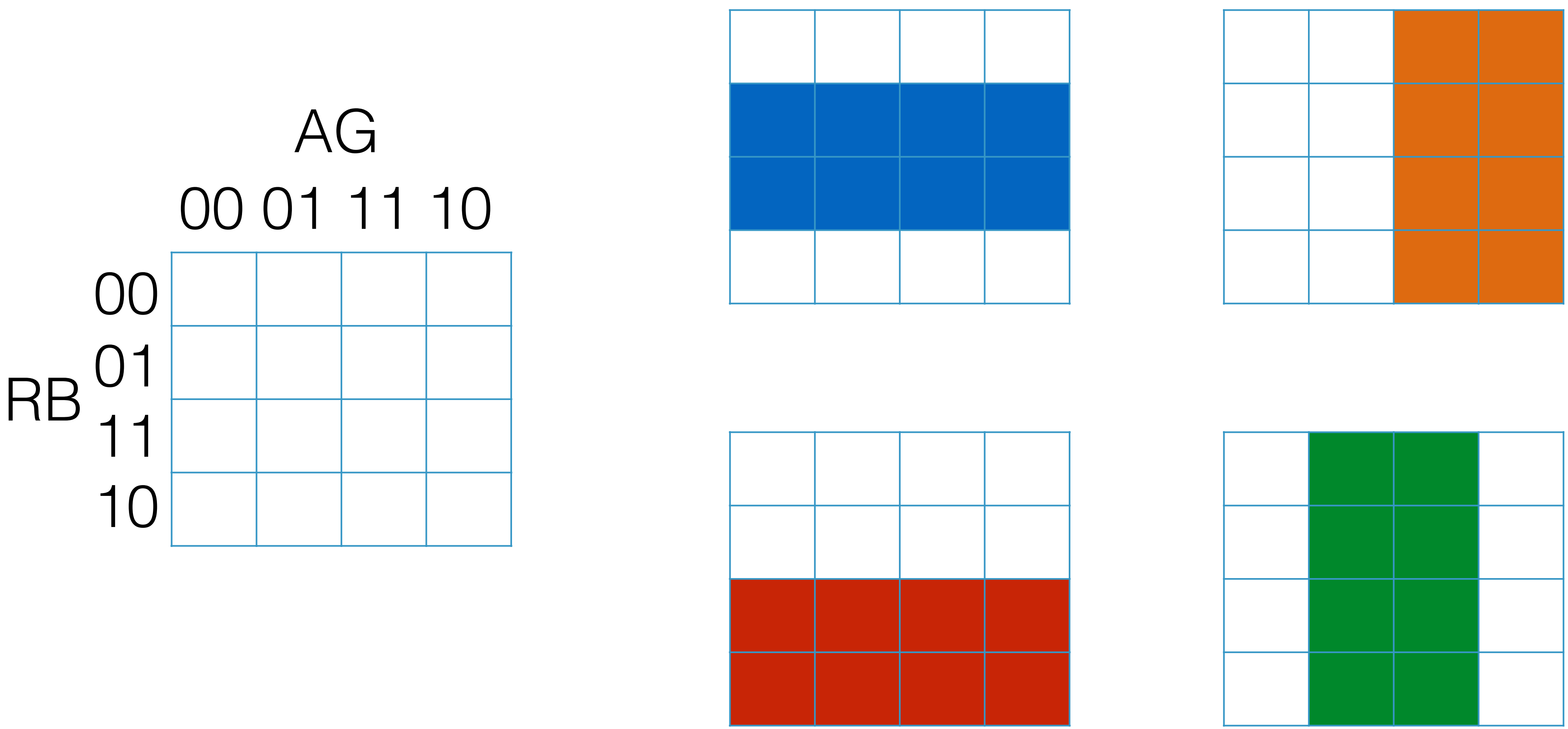




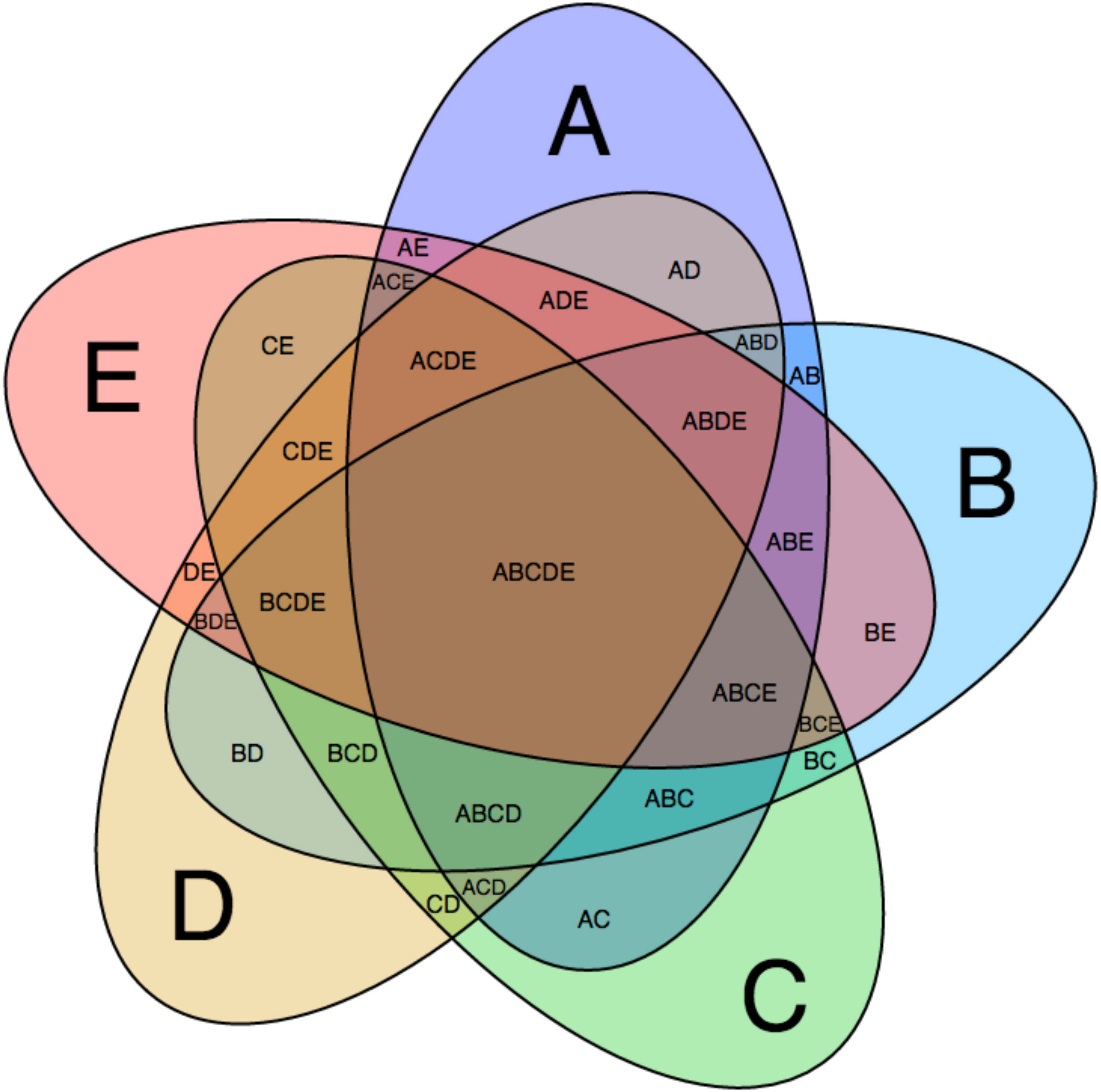


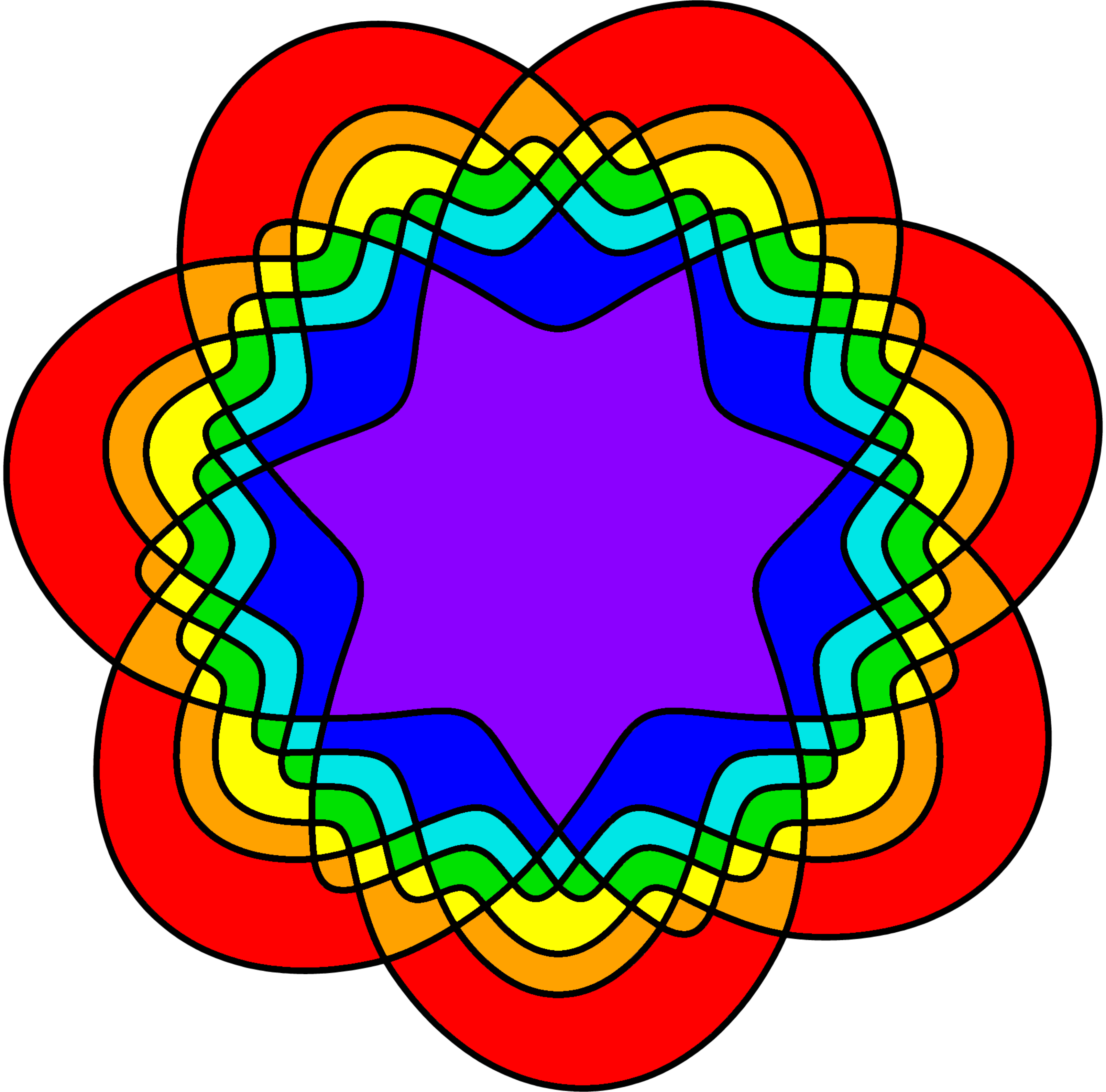


Karnaugh Maps



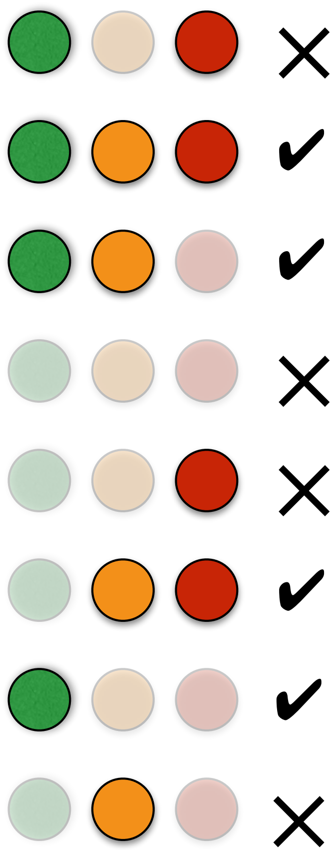
4 atoms:16 states: 64K subsets

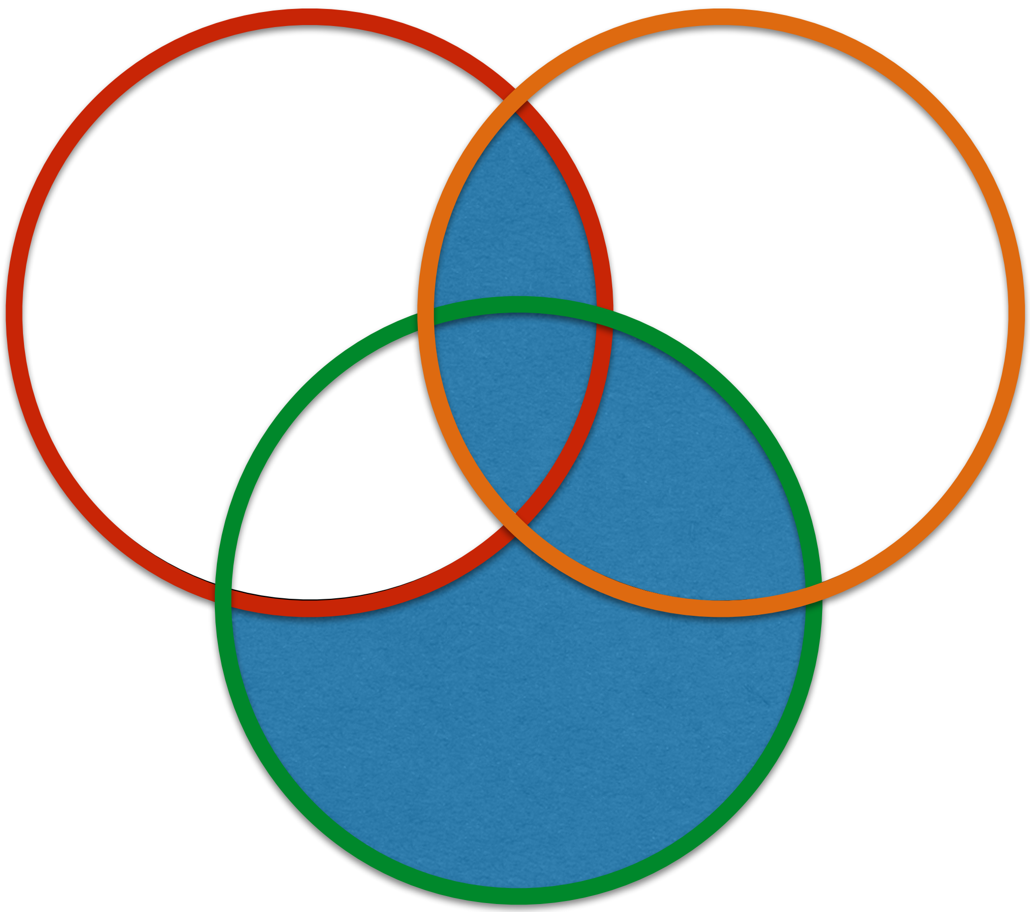

A
B
C
D
E
AE
ACE
AD
ADE
CE
ACDE
ABD
AB
ABDE
CDE
ABE
DE
ABCDE
BDE
BCDE
BE
ABCE
BCE
BD
BCD
BC
ABC
ABCD
ACD
CD
AC

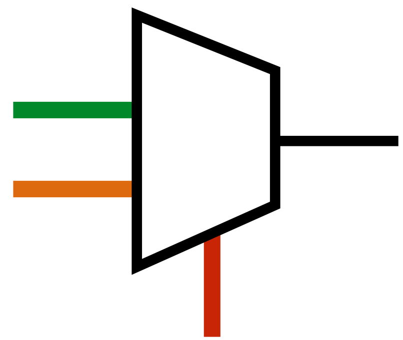


(R?A:G)

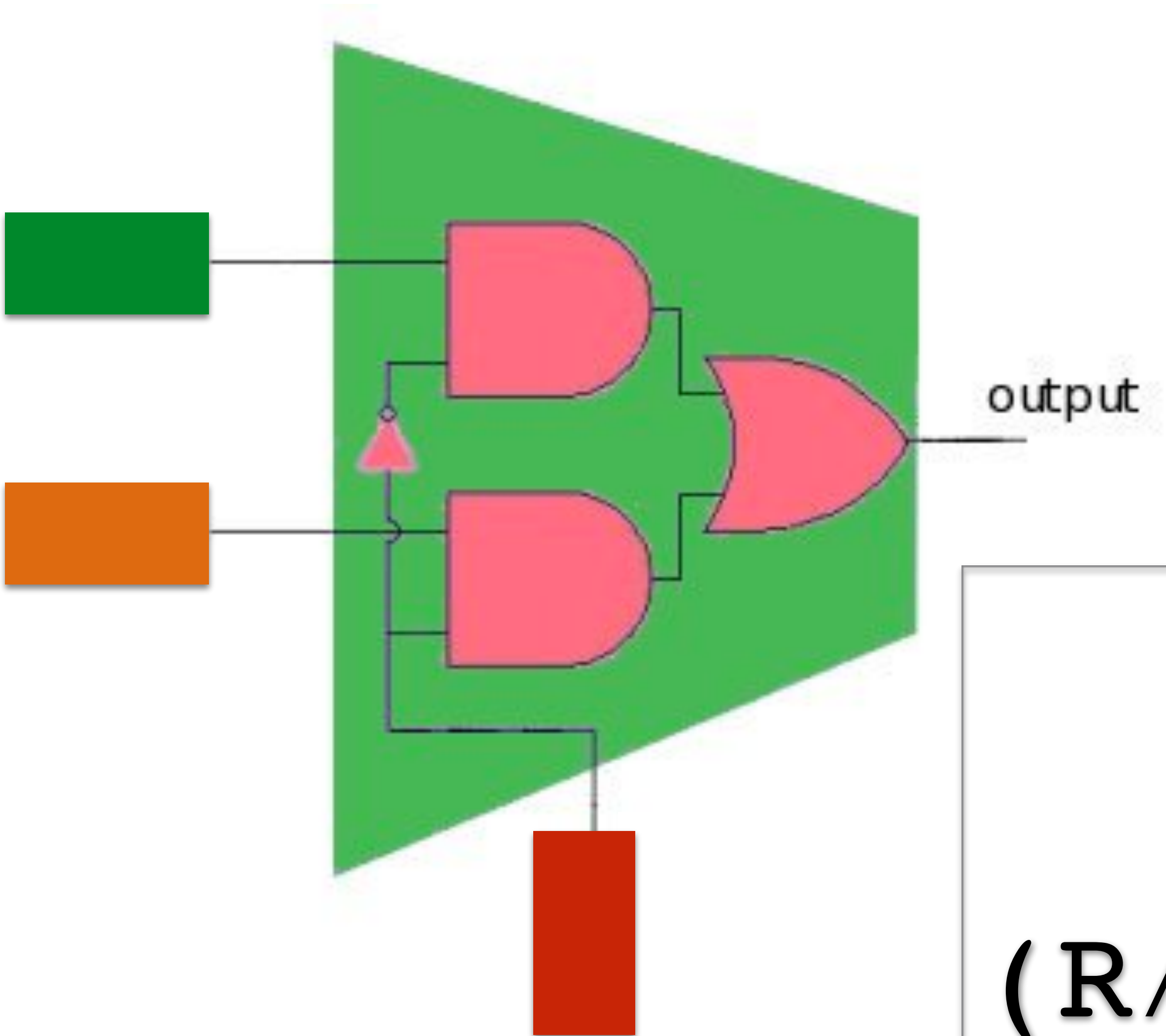
if R then A else G

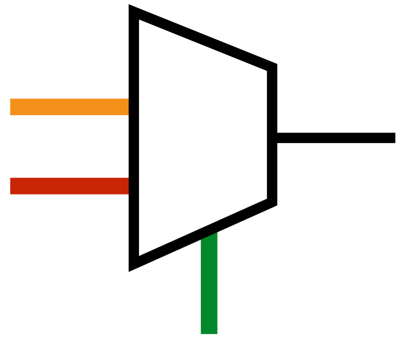




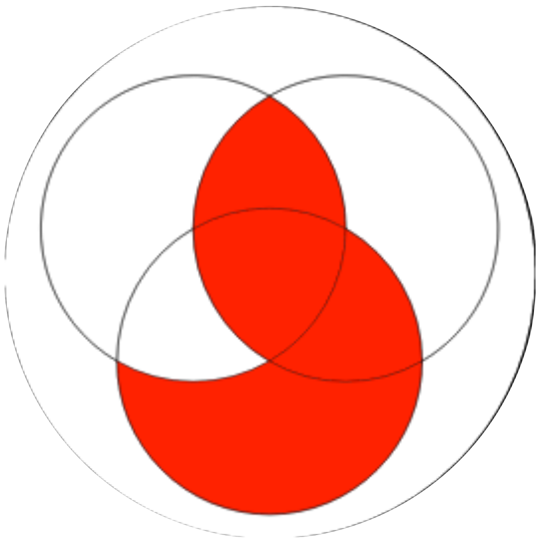


multiplexer – ITE





$$(R ? A : G)$$
$$=$$
$$(R \wedge A) \vee (\neg R \wedge G)$$



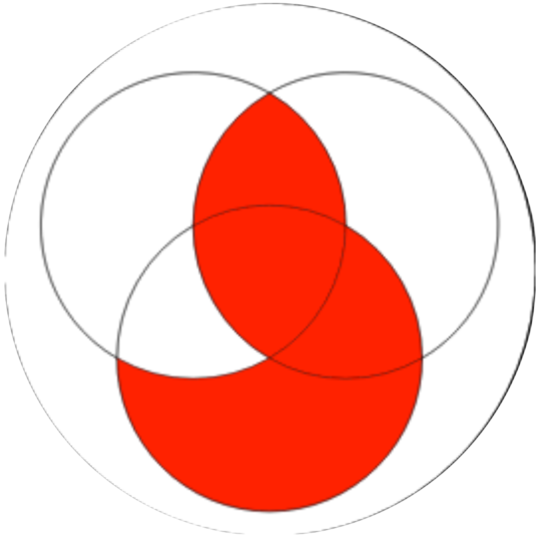
$R?\bot:\top$

$R?\top:A$

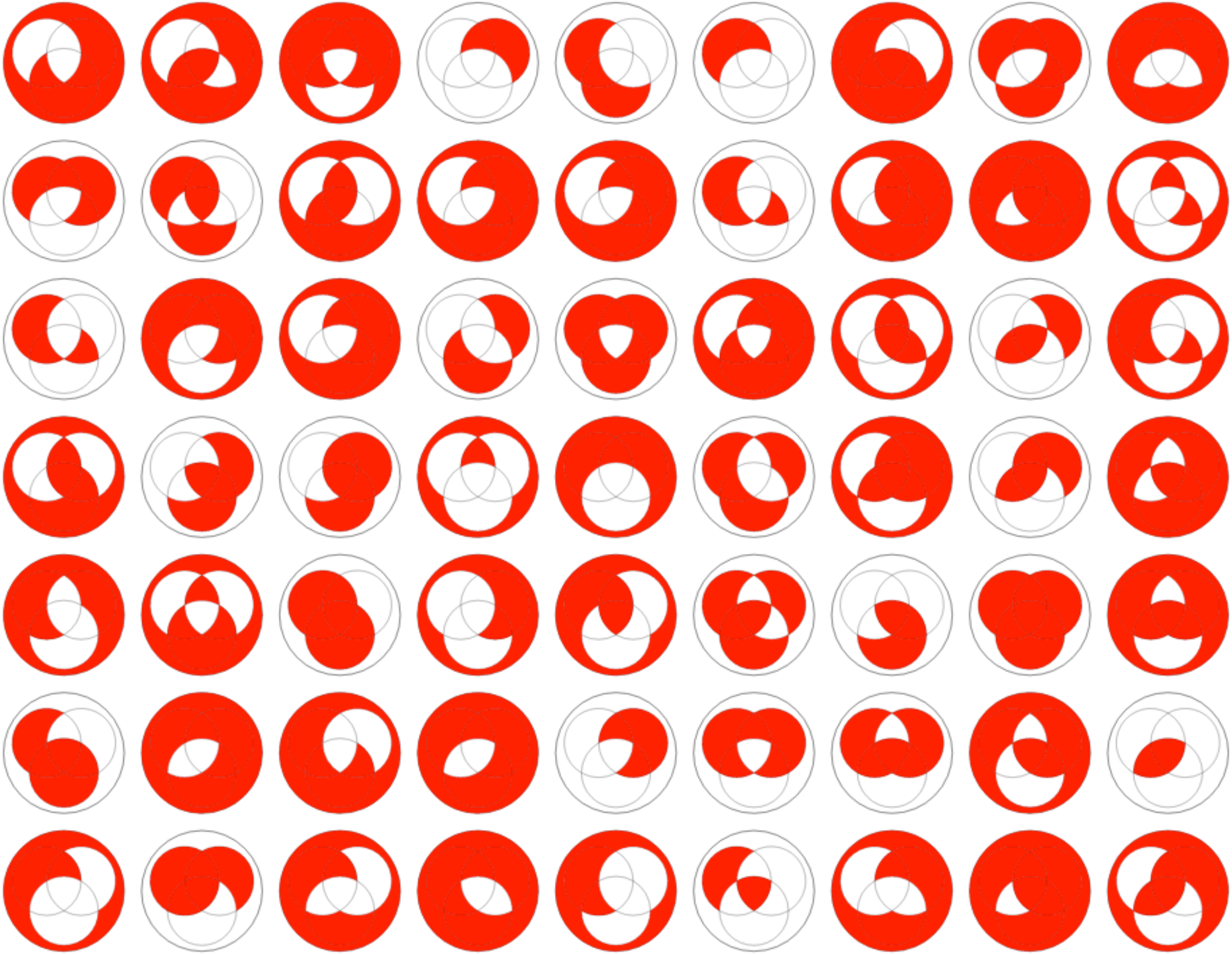
$R?A:\bot$

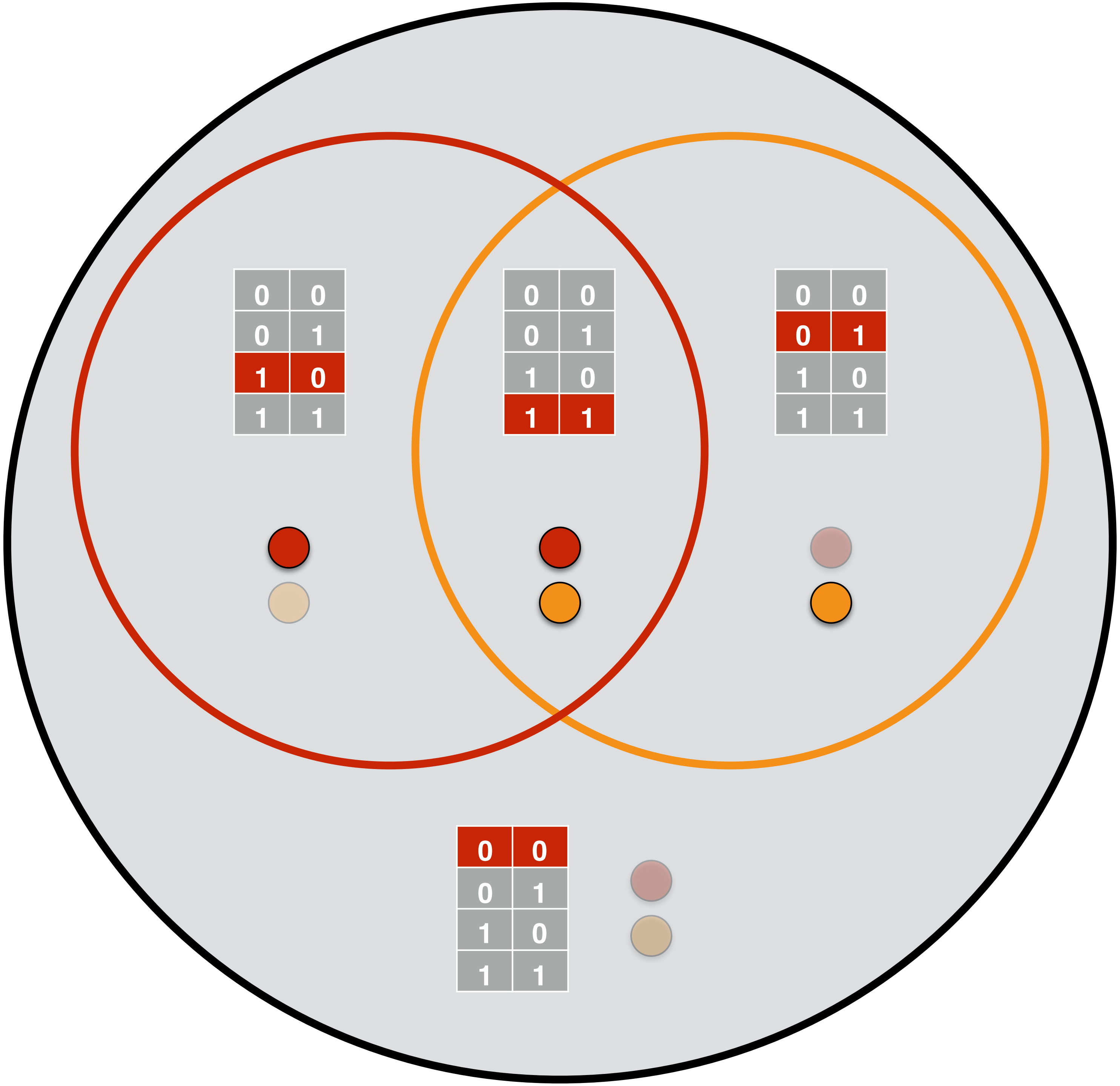
$R?A:\top$

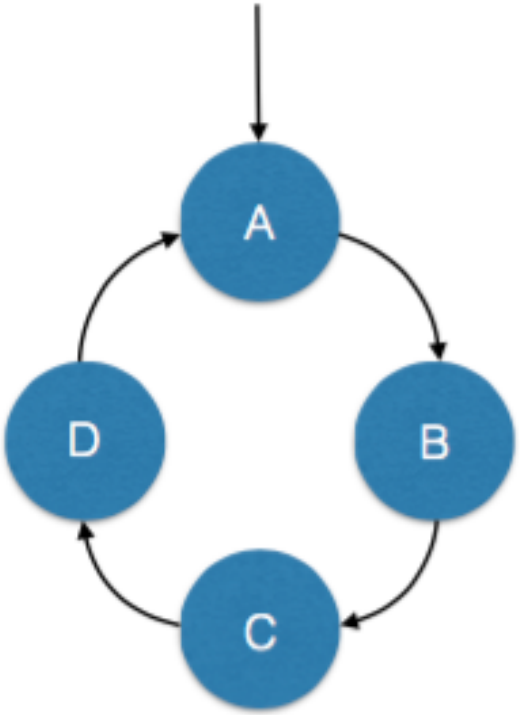
$R?\bot:A$

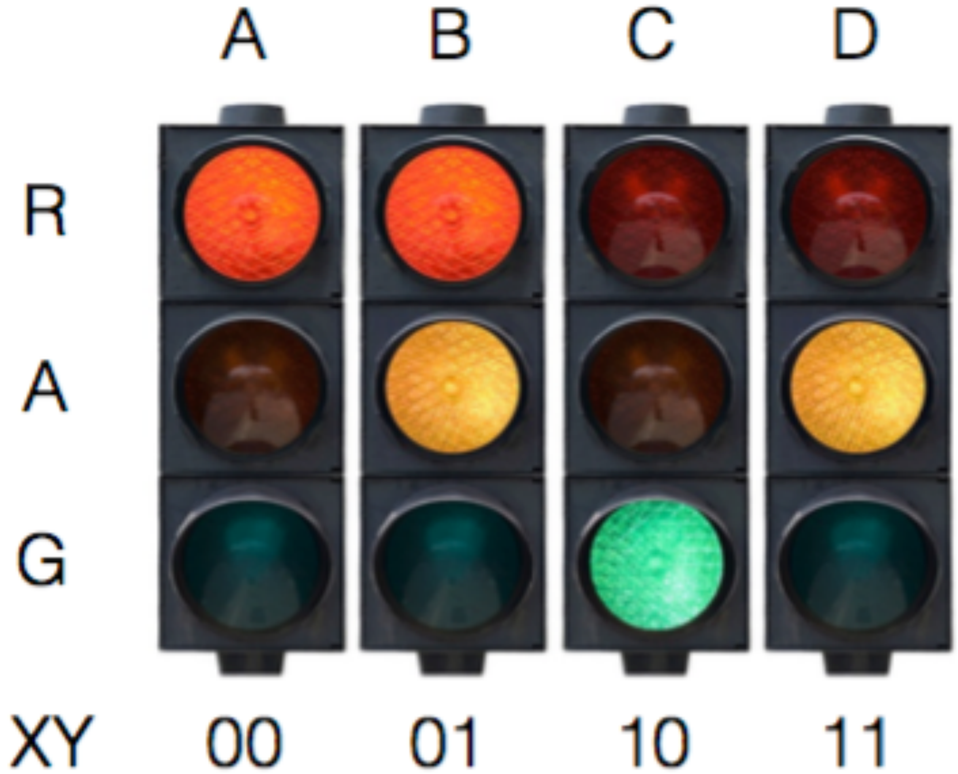


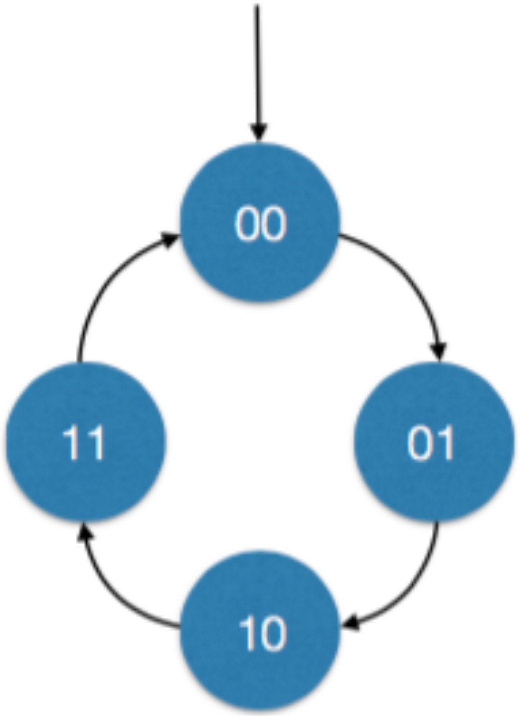
$R?\bot:\top$	$\neg R$
$R?\top:A$	$R \vee A$
$R?A:\bot$	$R \wedge A$
$R?A:\top$	$R \rightarrow A$
$R?\bot:A$	$\neg(A \rightarrow R)$




0 0
0 1
1 0
1 1
0 0
0 1
1 0
1 1
0 0
0 1
1 0
1 1
0 0
0 1
1 0
1 1







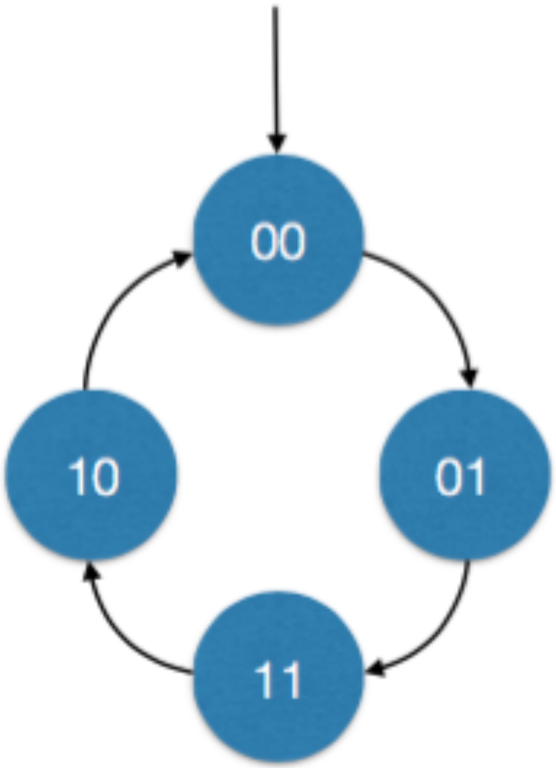
The traffic light has only four states.
the diagram shows a two-bit encoding of these four states. If we call the two bits X and Y then the next state logic can be given by

$$X' = X \oplus Y \text{ and } Y' = \neg Y$$

and the output logic (the signals to the lights) by

$$R = \neg X \quad A = Y \quad G = \neg X \wedge Y$$

This question concerns a different two-bit encoding of the four states, as shown below.

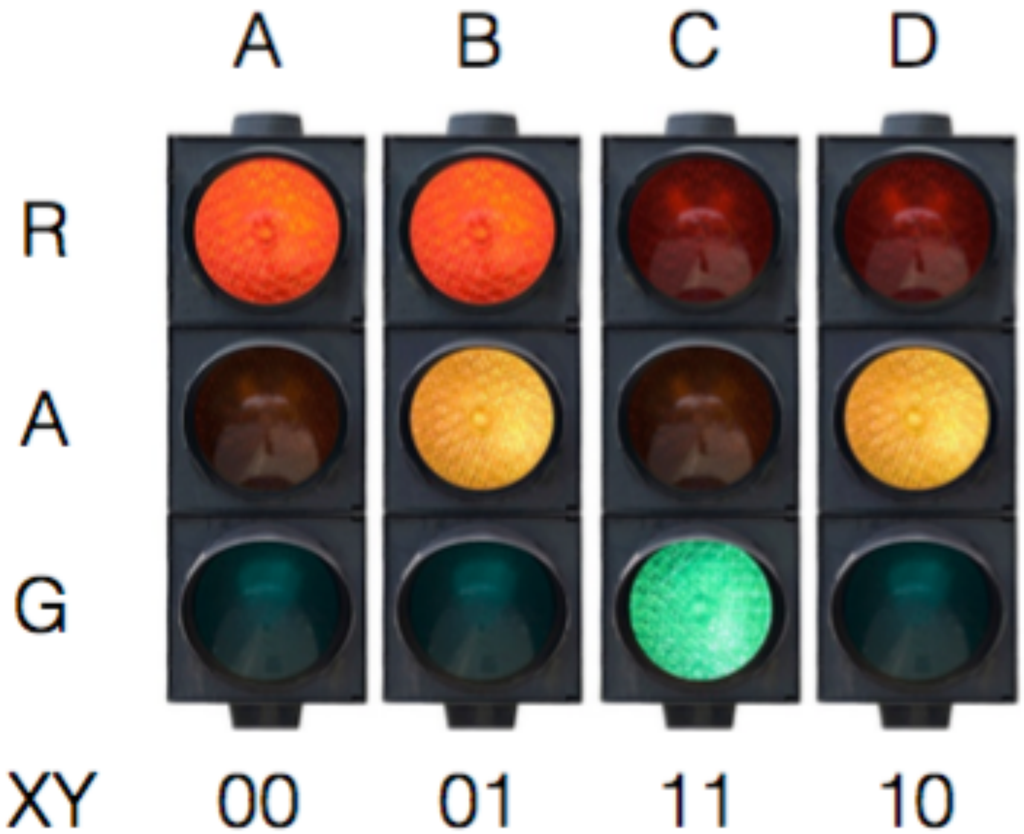


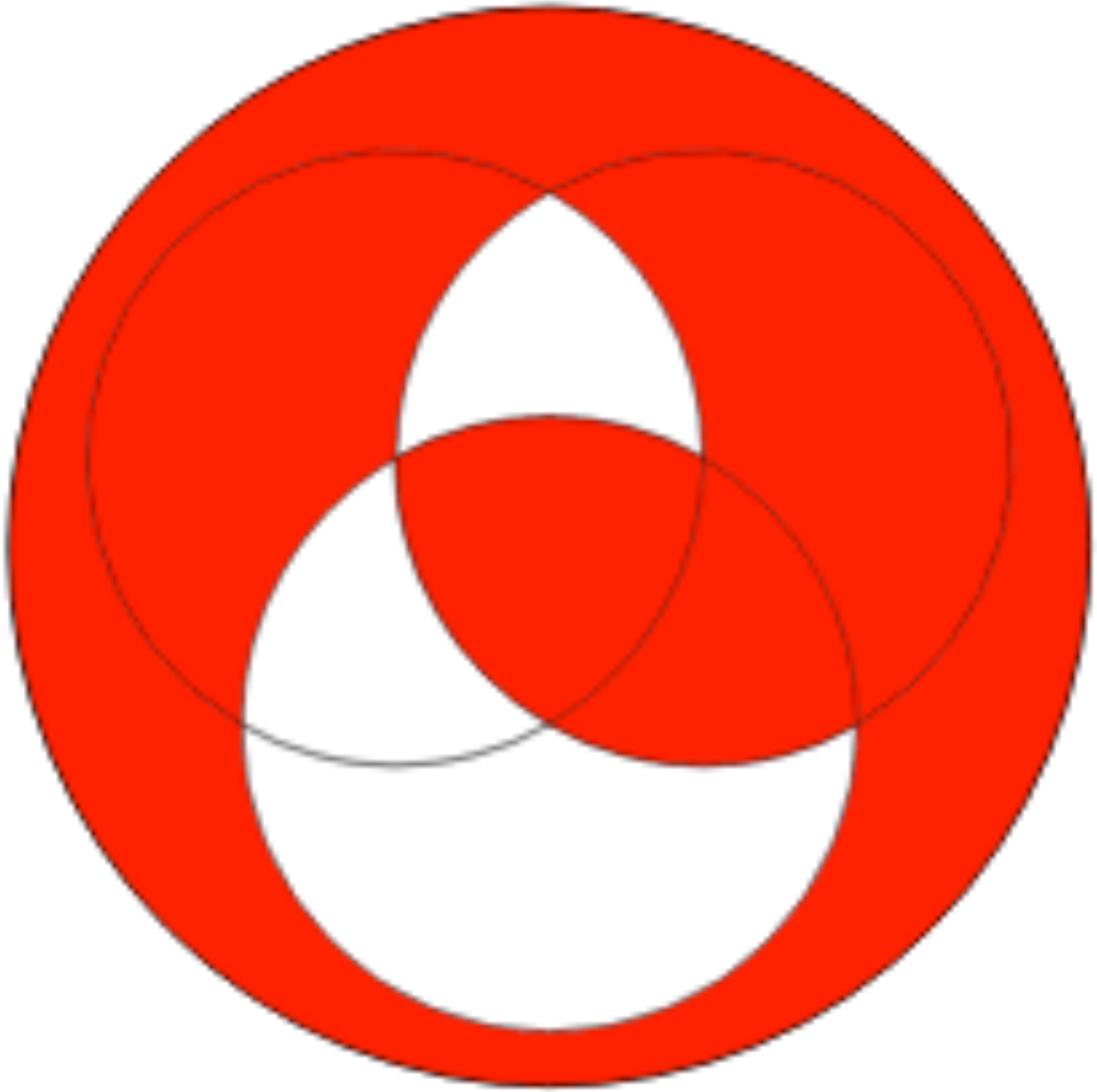
Give expressions for the next state logic

$X' = Y$ $\quad$ $Y' = \neg X$

and the output logic

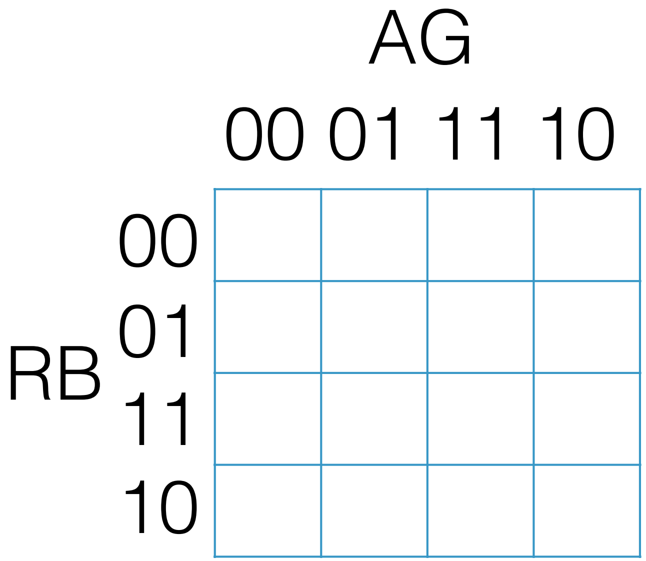
$R = \neg X$ $\quad$ $A = X \oplus Y$ $\quad$ $G = X \wedge Y$

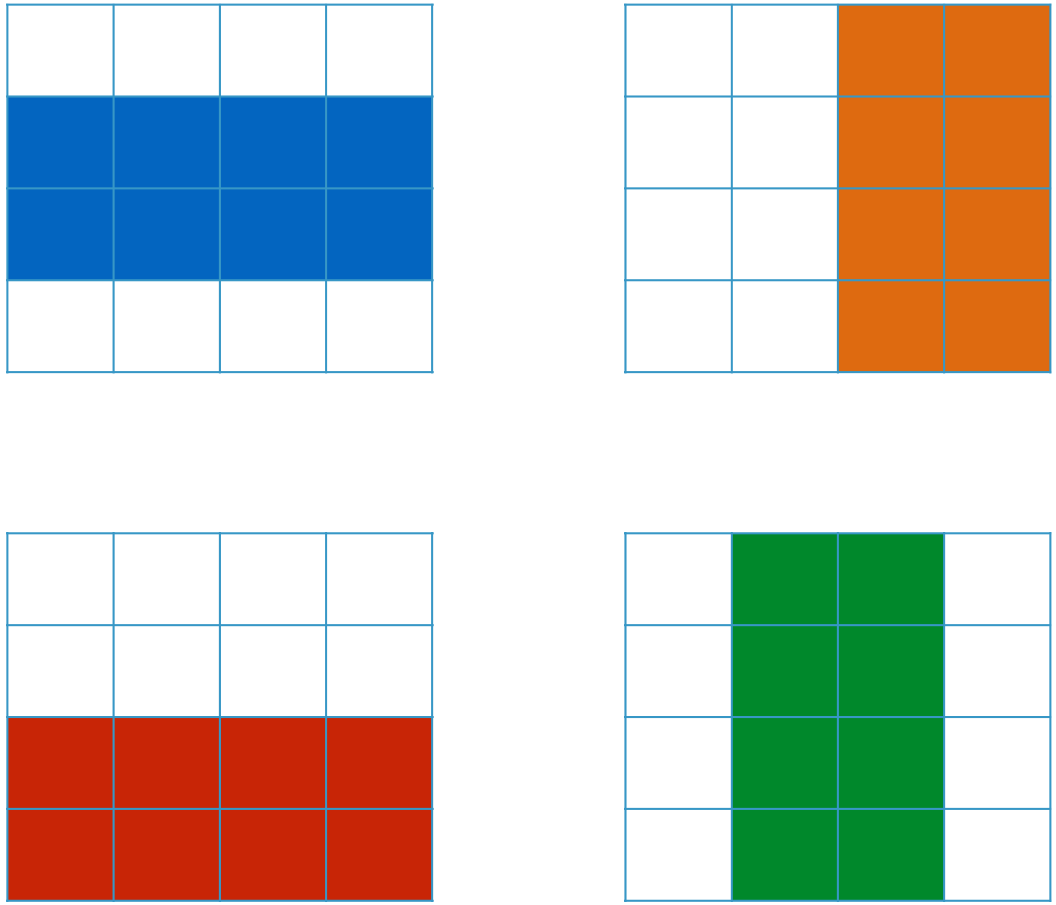




R	A	G	**??**
0	0	0	**1**
0	0	1	**0**
0	1	0	**1**
0	1	1	**1**
1	0	0	**1**
1	0	1	**0**
1	1	0	**0**
1	1	1	**1**

Karnaugh Maps





4 atoms:16 states: 64K subsets

R	A	G	??
0	0	0	1
0	**0**	**1**	**0**
0	1	0	1
0	1	1	1
1	0	0	1
1	**0**	**1**	**0**
1	**1**	**0**	**0**
1	1	1	1

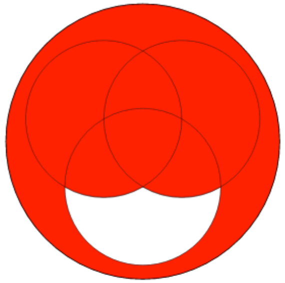
R ∨ A ∨ ¬G

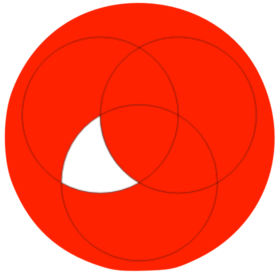
∧

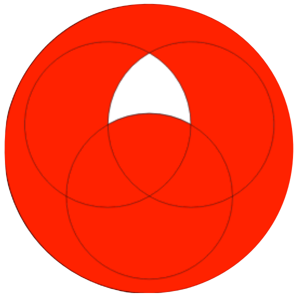
¬R ∨ A ∨ ¬G

∧

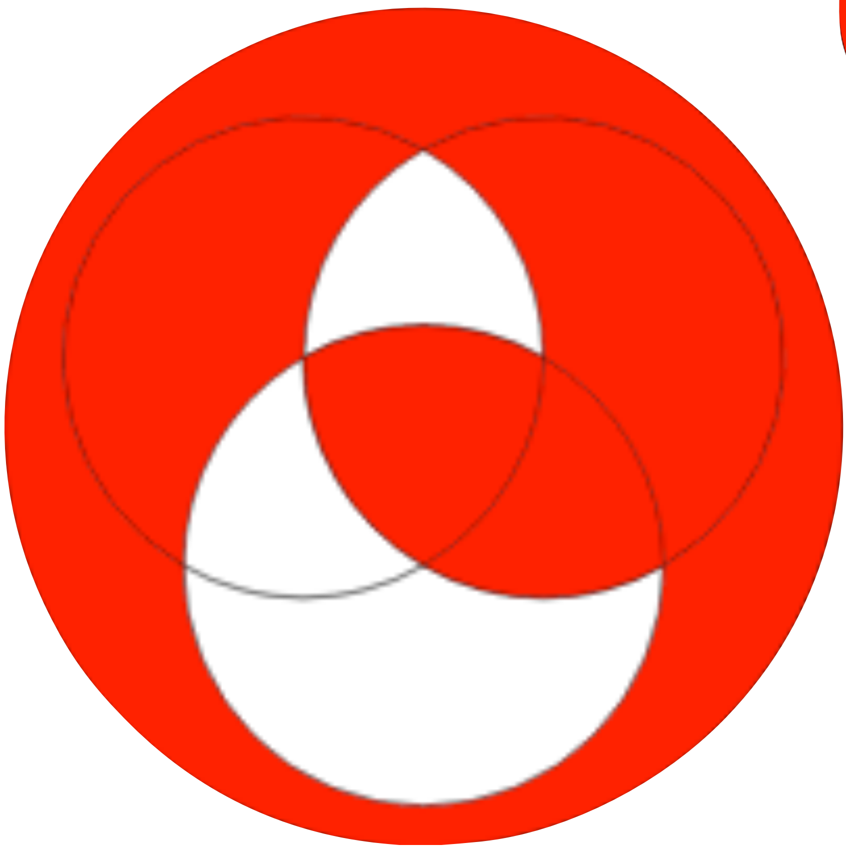
¬R ∨ ¬A ∨ G





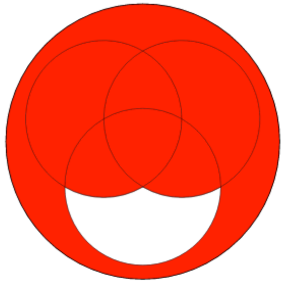


Three **constraints**



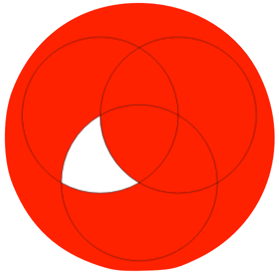
R	A	G	??
0	0	0	1
0	**0**	**1**	**0**
0	1	0	1
0	1	1	1
1	0	0	1
1	**0**	**1**	**0**
1	**1**	**0**	**0**
1	1	1	1

$\mathbf{R \lor A \lor \neg G}$



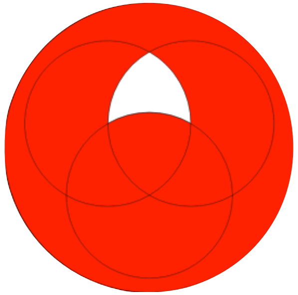
$\land$

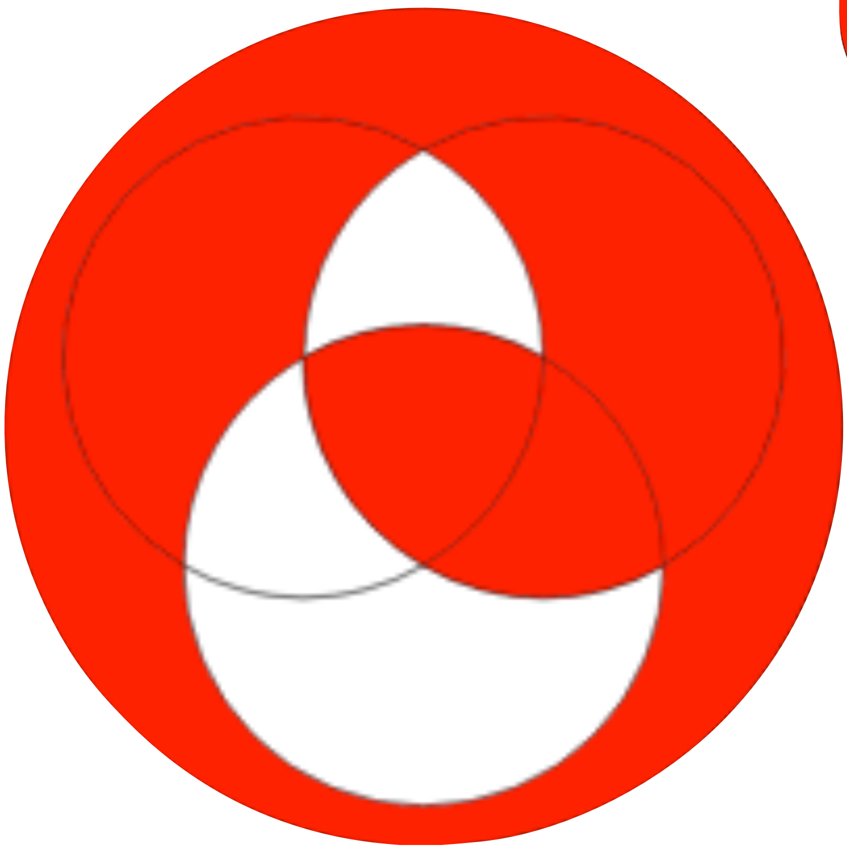
$\mathbf{\neg R \lor A \lor \neg G}$



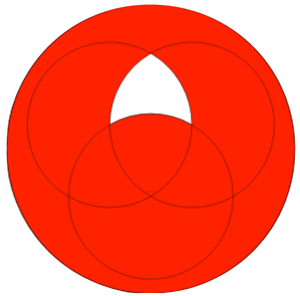
$\land$

$\mathbf{\neg R \lor \neg A \lor G}$



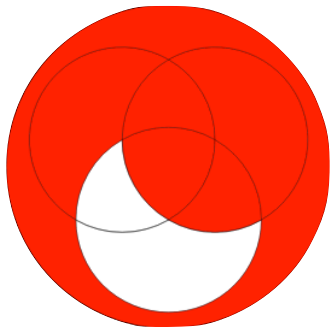


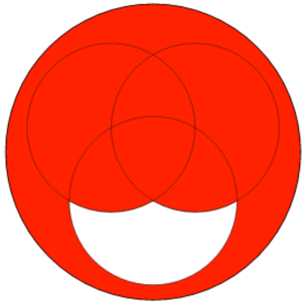
$\mathbf{\neg R \lor \neg A \lor G}$



$\land$

$\mathbf{A \lor \neg G}$

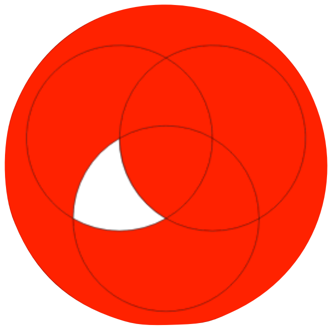




$\mathbf{R} \vee \mathbf{A} \vee \neg\mathbf{G}$

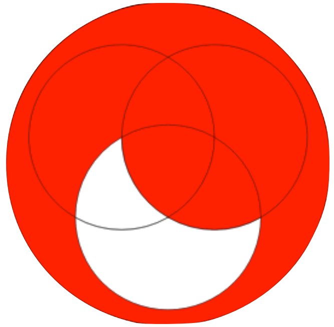
$\wedge$

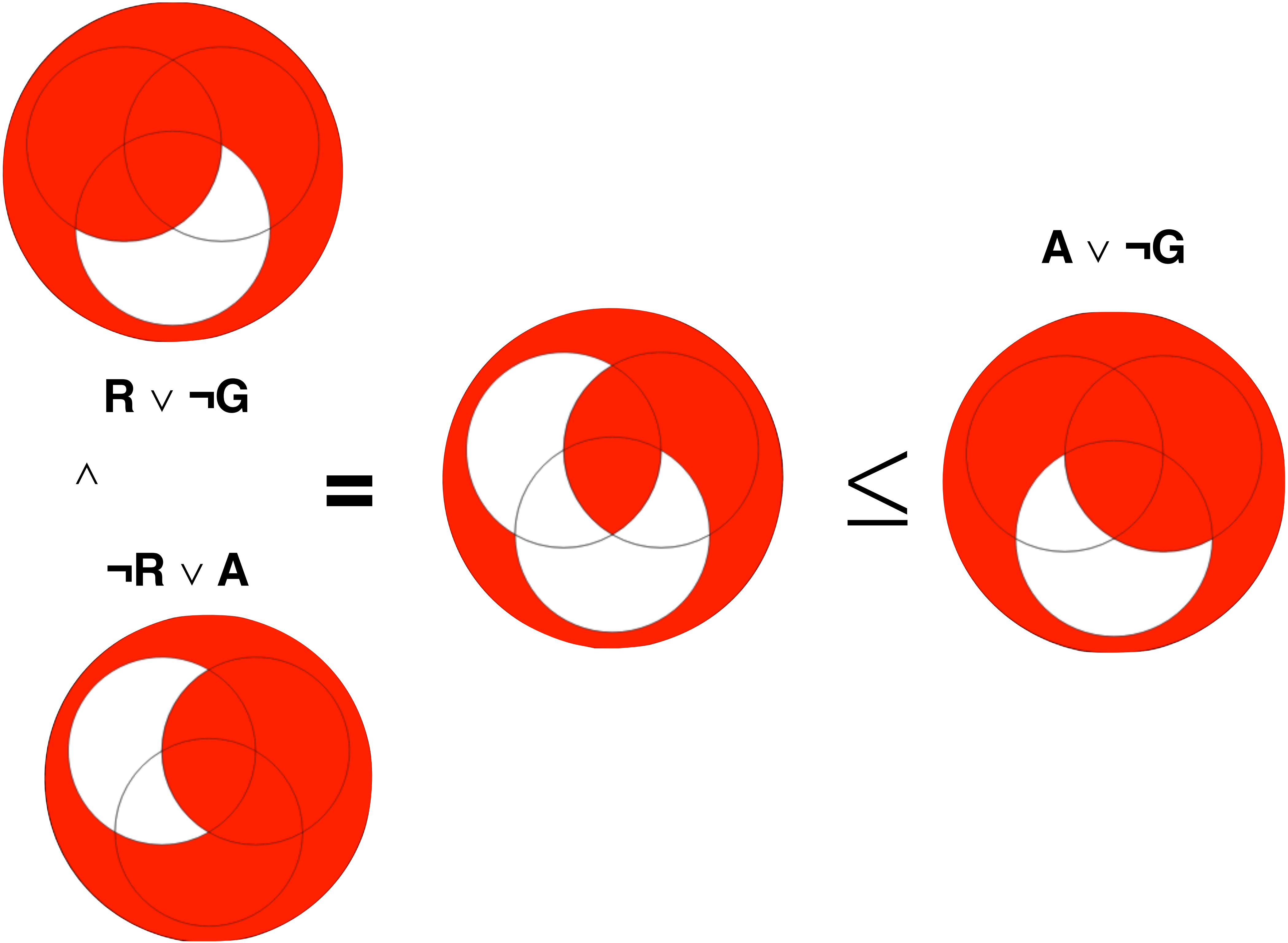
$\neg\mathbf{R} \vee \mathbf{A} \vee \neg\mathbf{G}$



$=$

$\mathbf{A} \vee \neg\mathbf{G}$




R ∨ ¬G
∧
¬R ∨ A
=
≤
A ∨ ¬G