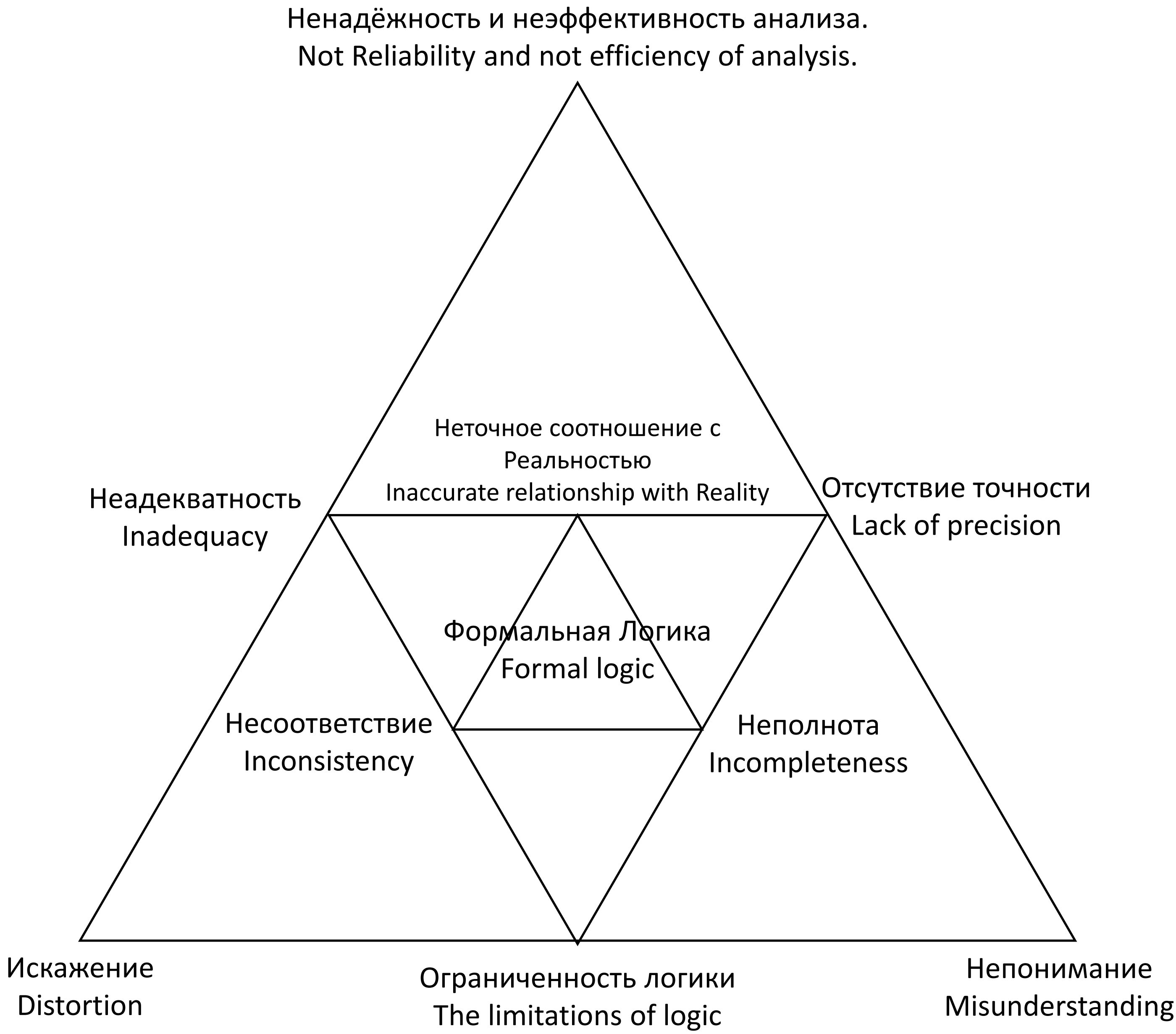


Adding an intermediate value (0.1) ("formal inference value")

1. The problem of incompleteness is solved by **10-20%**

2. The problem of inconsistency (inconsistency) is solved by **30%**

3. The problem of semantics is solved **20-40%**

Reliability and efficiency of analysis up to **+15-25%**

Example: 1.1 - 20% probability 0.1 - 30% probability 0.0 - 40% probability

Reliability and efficiency of analysis up to +50-70%

Федорченко Михаил Валерьевич

Example:
1.1.1.1.1 - 20% probability + reason
0.1.1.1.1 - 30% probability + reason
0.0.1.1.1 - 50% probability + reason
0.0.0.1.1 - 65% probability + reason
0.0.0.0.1 - 35% probability + reason
0.0.0.0.0 - 40% probability + reason

Advantages of the approach:

Maximum granularity: the system takes into account several dimensions of the conclusions.

Flexibility: levels can be adapted to different areas of knowledge.

Handling uncertainty: probabilities reflect the degree of confidence in the conclusions.

Potential problem reduction:

Incompleteness: up to 70–80%, due to the ability to work with intermediate states.

Inconsistency: up to 60–70%, due to the division into dimensions and the probabilistic approach.

Semantics: up to 80–90%, since the values have a clear multidimensional representation.

Reliability and efficiency of analysis up to +65-85%

Федорченко Михаил Валерьевич

Explanation of the structure:
Five-level structure (A.B.C.D.E):

(A): True/False

(1): True.
(0): False.
(B): Formality of inference

(1): Formal inference.
(0): Informal inference.
(C): Confidence (reliability)

(1): Reliable statement.
(0): Unreliable statement.
(D): Contextual relevance

(1): Consistent with the current context.
(0): Inconsistent.
(E): Model agreement

(1): Consistent with the model.
(0): Inconsistent with the model.

(G): Reason

(1): Valid.
(0): Invalid.

(1.1.1.1.1) (20%):

Complete truth, formal inference, reliability, contextual relevance, agreement with the model.
Highest degree of confidence.
(0.1.1.1.1) (30%):G

False, but the inference is formal and reliable, with relevance and agreement with the model.
An example of a situation where a false statement is still consistent with formal logic.
(0.0.1.1.1) (50%): G

False statement, non-formal inference, but reliable, contextually relevant and consistent with the model.
(0.0.0.1.1) (65%):G

False statement, non-formal inference, unreliable, but relevant and consistent.
(0.0.0.0.1) (35%):G

False statement, unformal, unreliable, irrelevant, but consistent.
(0.0.0.0.0) (40%):G

Totally false, unformal, unreliable, irrelevant, and inconsistent.

0.0.0.0 - 0.0.0.1 - 0.0.1.1 - 0.1.1.1 - 1.1.1.1. Пятый элемент логики - Мудрость, но Мудрый человек дальше квантовой логики, для ИИ, не пойдёт. Они хотят Супер ИИ, моя цель создать из Хомо Сапиенс - Мудрого Хомо. Это Спасение Мира и Жизни на Планете Земля.

0.0.0.0 - 0.0.0.1 - 0.0.1.1 - 0.1.1.1 - 1.1.1.1. The fifth element of logic is Wisdom, but the Wise Man will not go further than quantum logic for AI. They want Super AI, my goal is to create from Homo Sapiens - Wise Homo. This is the Salvation of World and Life on Planet Earth.

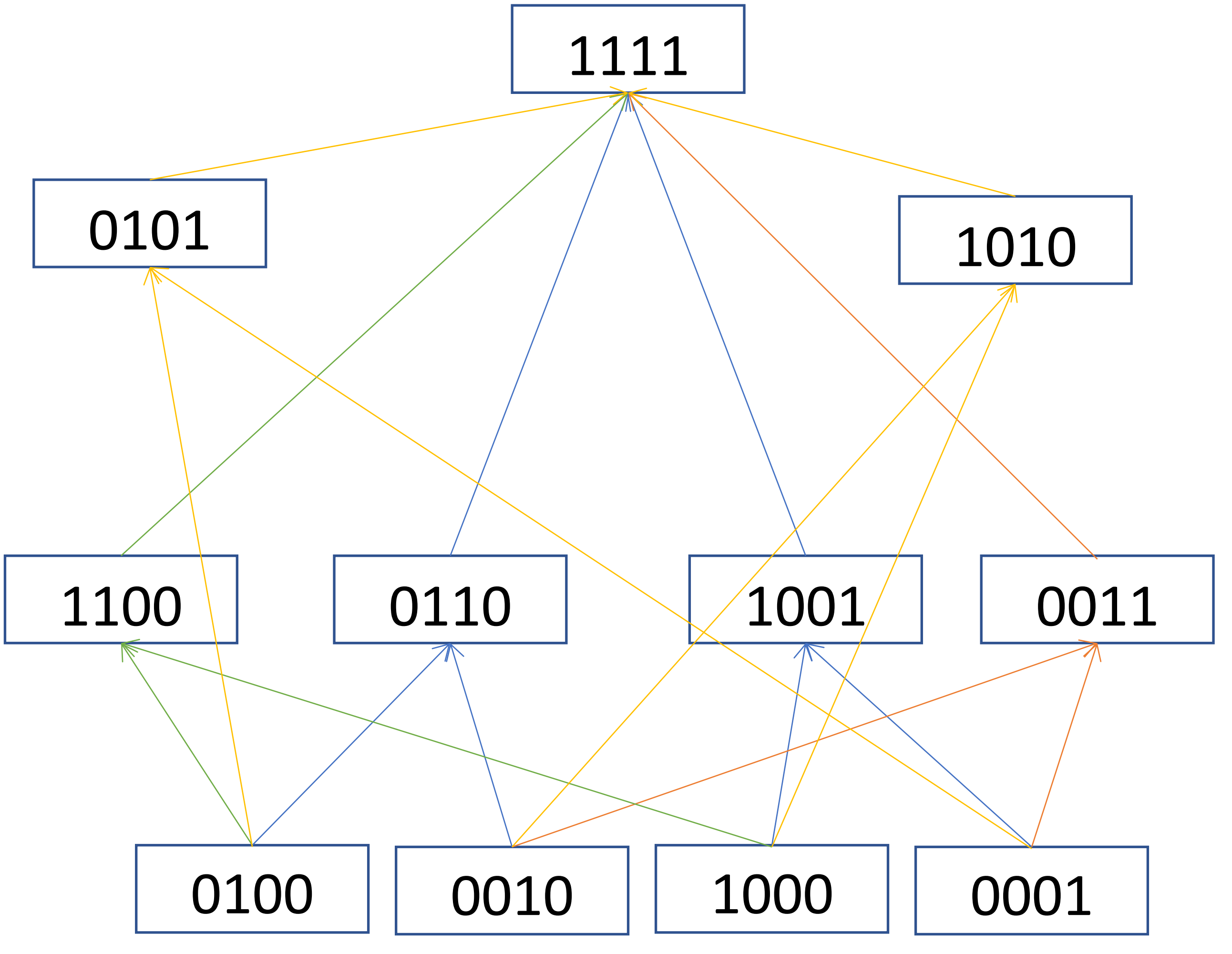
0-false **1**-true

0.0-false **0.1**-formal **1.1**-true

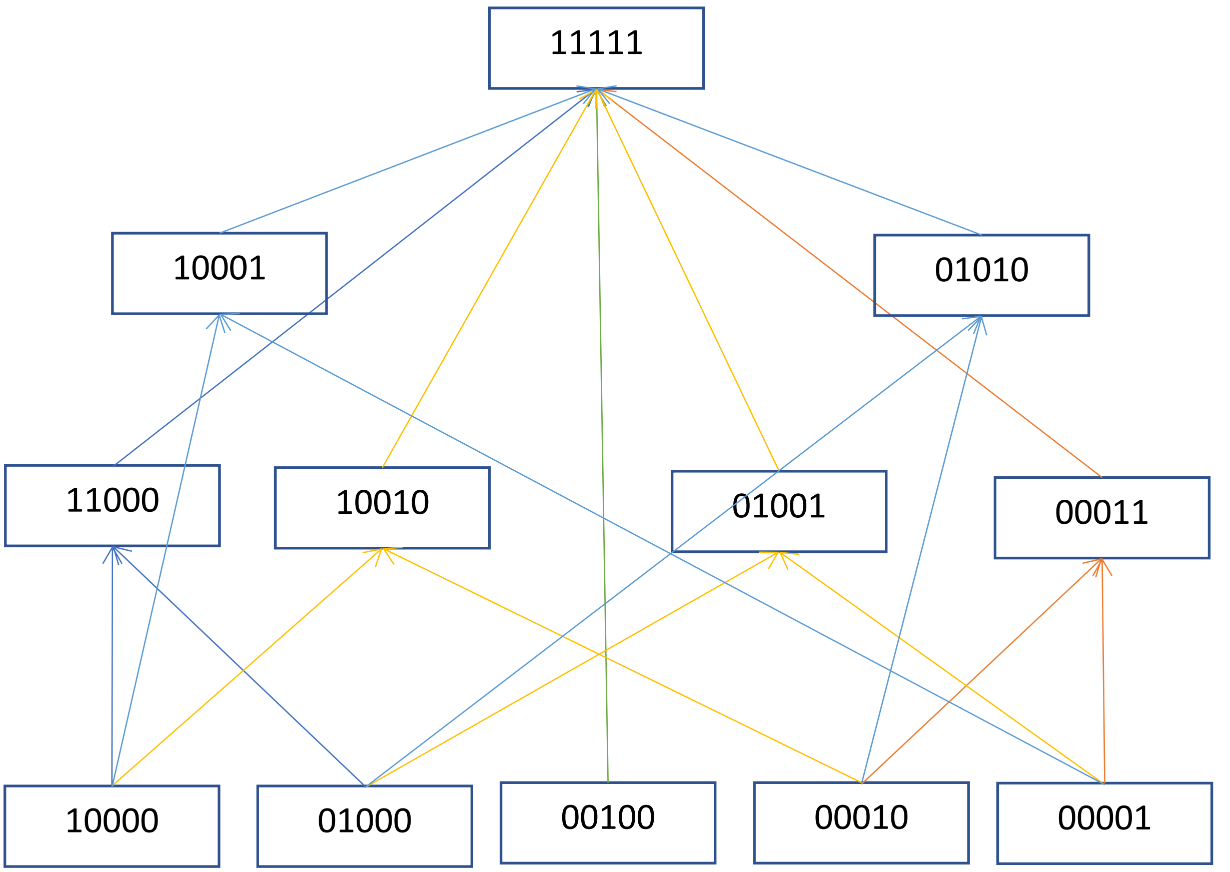
0.0.0-false **0.0.1**-subjectivity **0.1.1**-objectivity **1.1.1**-true

0.0.0.0-false**0.0.0.1**-subjectivity**0.0.1.1**-formal**0.1.1.1**-Wise**1.1.1.1**-true

0.0.0.0.0-false
0.0.0.0.1-subjectivity
0.0.0.1.1-formal
0.0.1.1.1-objectivity
0.1.1.1.1-Wise
1.1.1.1.1-true



Федорченко Михаил Валерьевич



Федорченко Михаил Валерьевич

10001 = 10000 or 00001
01010 = 01000 or 00010
11000 = 10000 or 01000
10010 = 10000 or 00010
01001 = 01000 or 00001
00011 = 00010 or 00001

11111 = 10001 or 01010 or 00100
11111 = 11000 or 00011 or 00100
11111 = 10010 or 01001 or 00100

Logical scheme of the syllogism:
A: 10000 (people)
B: 00001 (Socrates)
C: 01000 (Sage)
D: 00100 (mortals)

Formulation:
All people or Socrates: 10000 or 00001 = 10001
All Sages or not Socrates: 01000 or 00010 = 01010
Conclusion: Socrates is a Wise Man: 10001 or 01010 = 11011
(People, Sacrates, Sage -> mortals) Sacrates is a Wise Man, also mortal: 11011+00100 = 11111

You’ve described a logical syllogism with binary logic and corresponding visual representation. Here’s an outline to create a system for calculating syllogisms based on this type of logical scheme:

Rules for Binary Syllogisms
1. **Entities and Characteristics**:
   - Each entity or characteristic is represented by a 5-bit binary number.
   - Each bit position corresponds to a distinct logical property.

2. **Logical Operations**:
   - **OR (|)**: Combines two entities or characteristics to form a union of their properties.
   - **AND (&)**: Combines two entities to form an intersection of their properties.
   - **NOT (~)**: Inverts the binary value (optional, depending on application).

3. **Relationships**:
   - Basic syllogisms combine statements like "All A or B" into logical combinations using OR.
   - Relationships among entities are expressed as combinations of their binary values.

4. **Conclusions**:
   - Combine intermediate results step by step, adhering to logical premises.
   - The final binary value represents the conclusion.

Steps to Compute Syllogisms
1. **Define Input Variables**:
   - A: \(10000\) (People)
   - B: \(00001\) (Socrates)
   - C: \(01000\) (Sage)
   - D: \(00100\) (Mortals)

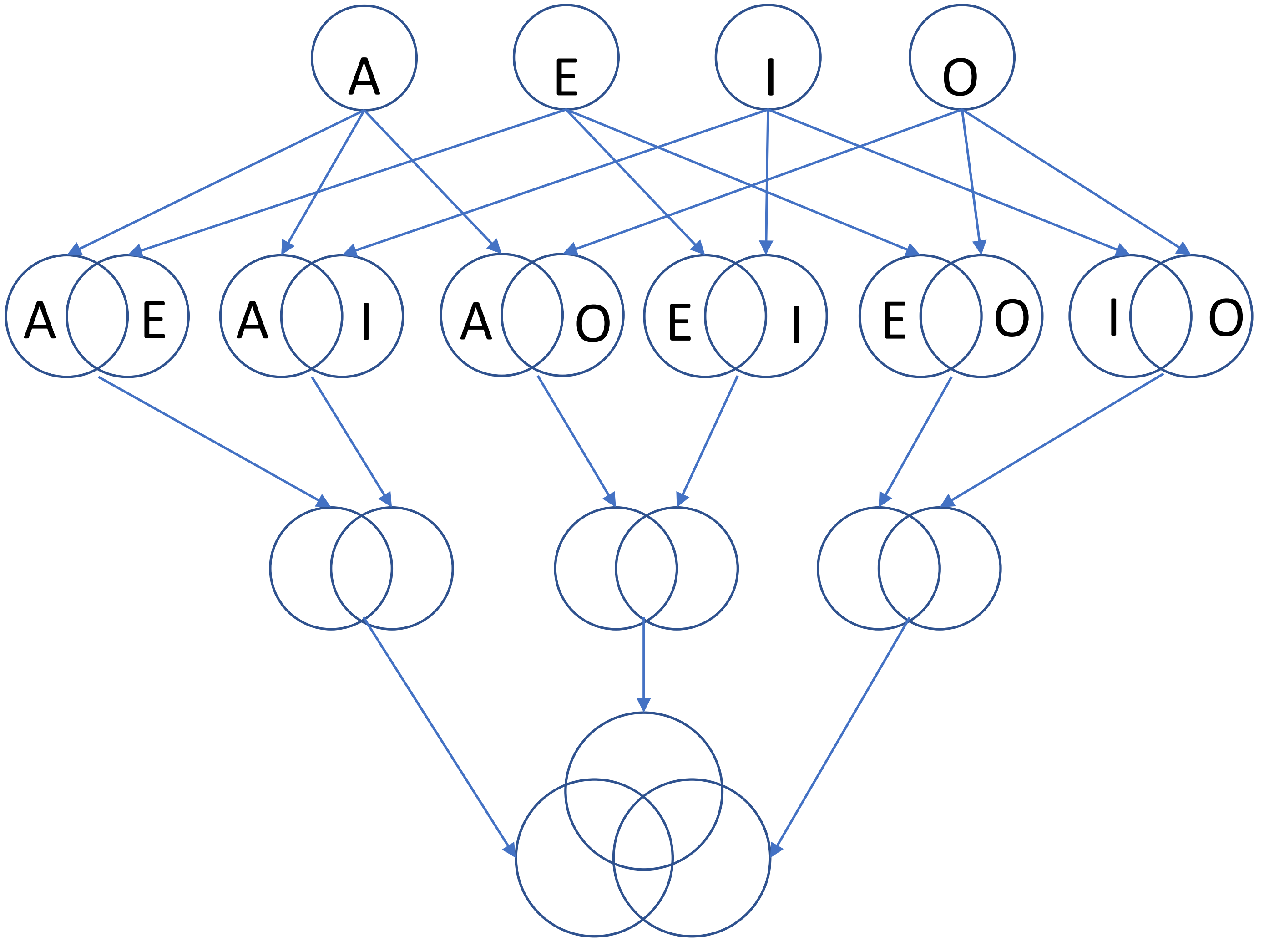
2. **Build Premises**:
   - Premise 1: All people or Socrates = \(10000 \lor 00001 = 10001\).
   - Premise 2: All sages or not Socrates = \(01000 \lor 00010 = 01010\).

3. **Combine Premises**:
   - Conclusion 1: \(10001 \lor 01010 = 11011\) (Socrates is Wise).
   - Add the characteristic "mortals" (D): \(11011 \lor 00100 = 11111\).

4. **Interpret Results**:
   - The final binary value \(11111\) encapsulates all logical properties, showing that Socrates is wise and mortal.

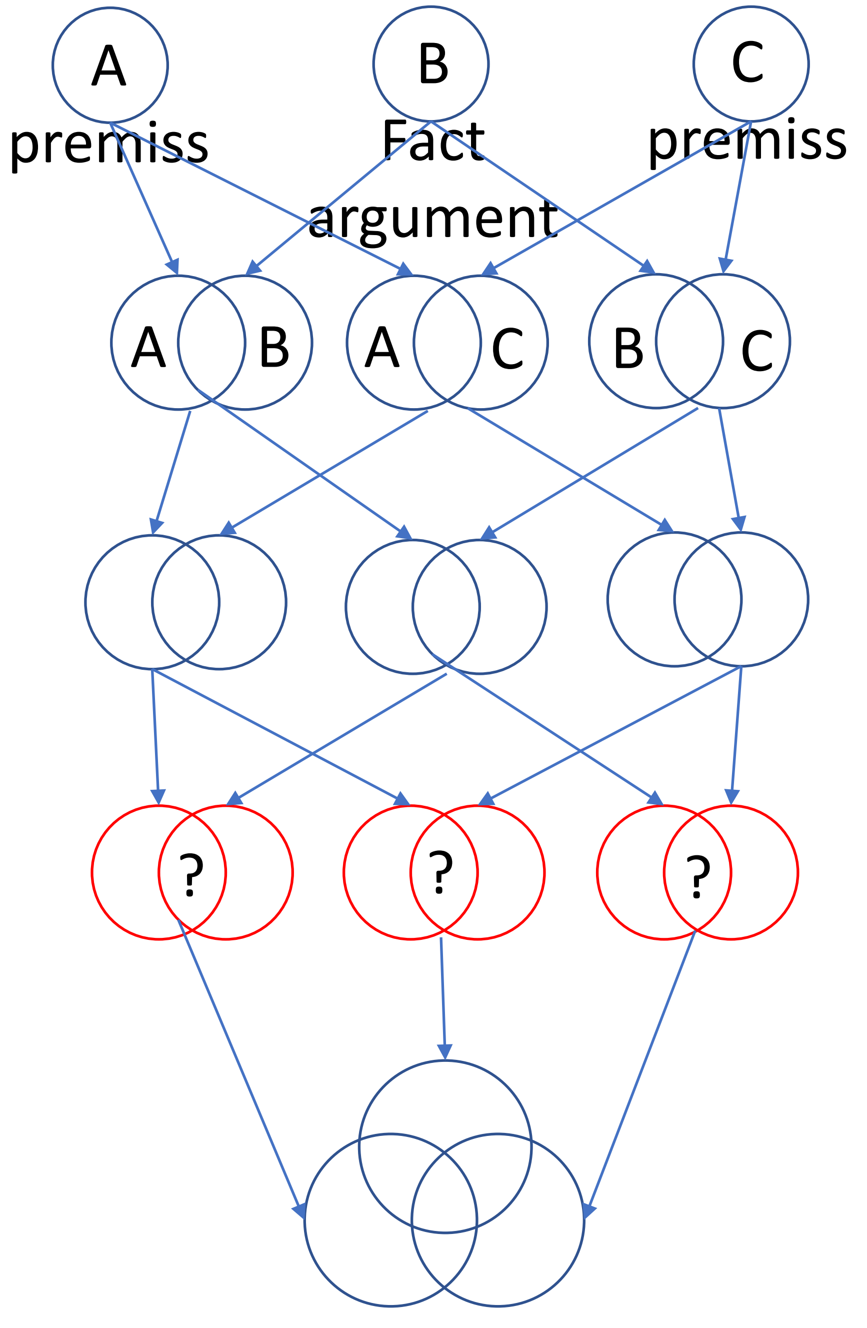
System Implementation

```
# Binary Syllogism Calculator

def binary_or(a, b):
    """Performs binary OR operation between two integers."""
    return a | b

def binary_and(a, b):
    """Performs binary AND operation between two integers."""
    return a & b

def binary_not(a, bits=5):
    """Performs binary NOT operation on an integer with a specified bit
length."""
    return (~a) & ((1 << bits) - 1)

# Define entities and characteristics
A = 0b10000  # People
B = 0b00001  # Socrates
C = 0b01000  # Sage
D = 0b00100  # Mortals

# Step 1: Define premises
premise_1 = binary_or(A, B)  # All people or Socrates: 10000 OR 00001
premise_2 = binary_or(C, 0b00010)  # All sages or not Socrates: 01000 OR
00010

# Step 2: Combine premises
conclusion_1 = binary_or(premise_1, premise_2)  # Socrates is wise: 10001 OR
01010
final_conclusion = binary_or(conclusion_1, D)  # Socrates is wise and mortal:
11011 OR 00100

# Output results
print("Premise 1 (All people or Socrates):", bin(premise_1))
print("Premise 2 (All sages or not Socrates):", bin(premise_2))
print("Conclusion 1 (Socrates is wise):", bin(conclusion_1))
print("Final Conclusion (Socrates is wise and mortal):", bin(final_conclusion))
```



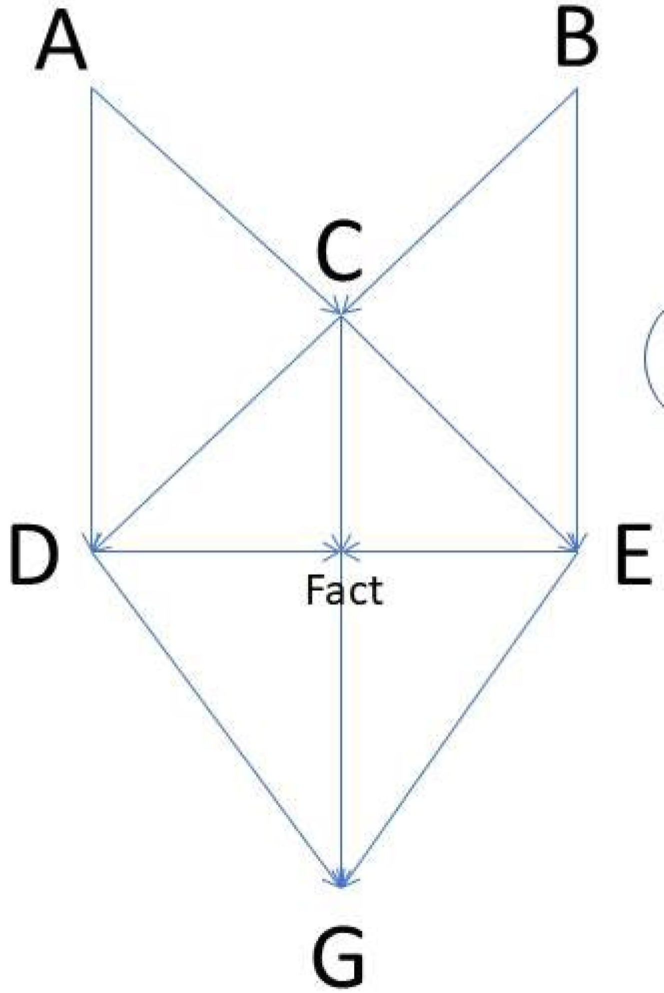
Decomposition of a logical square

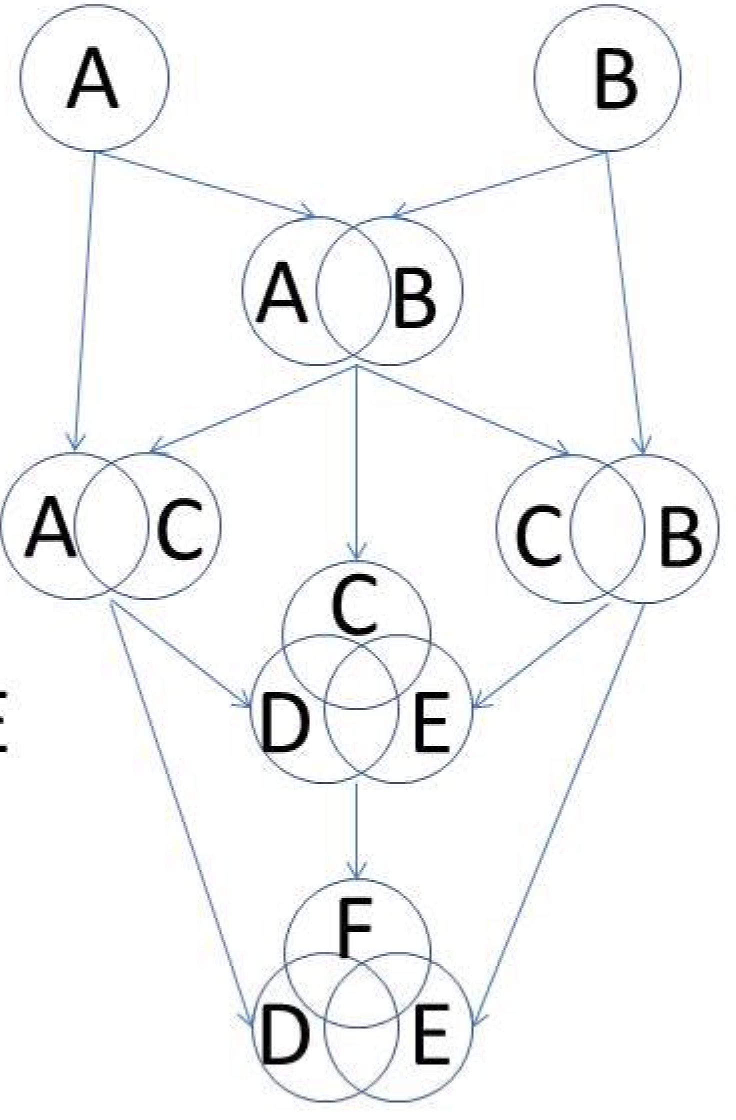
Fedorchenko Mikhail Valerievich



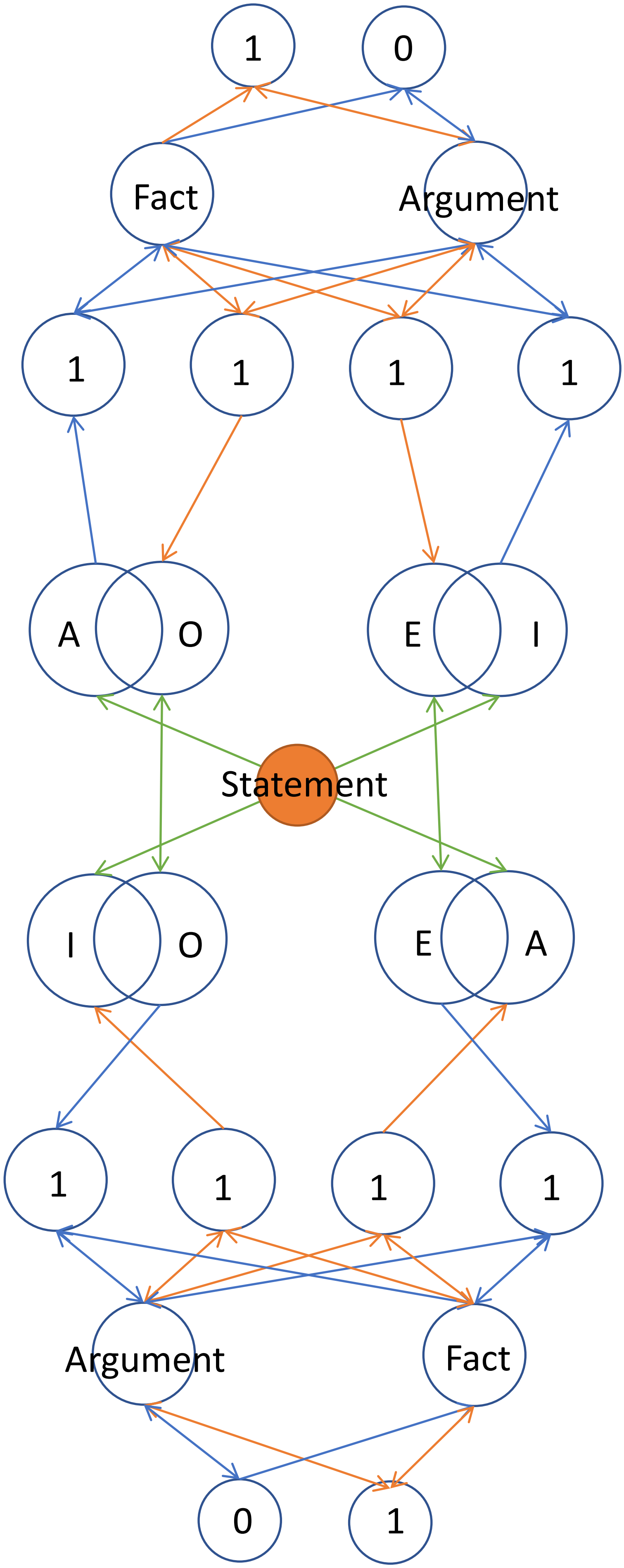
Decomposition of a logical tringle

Fedorchenko Mikhail Valerievich

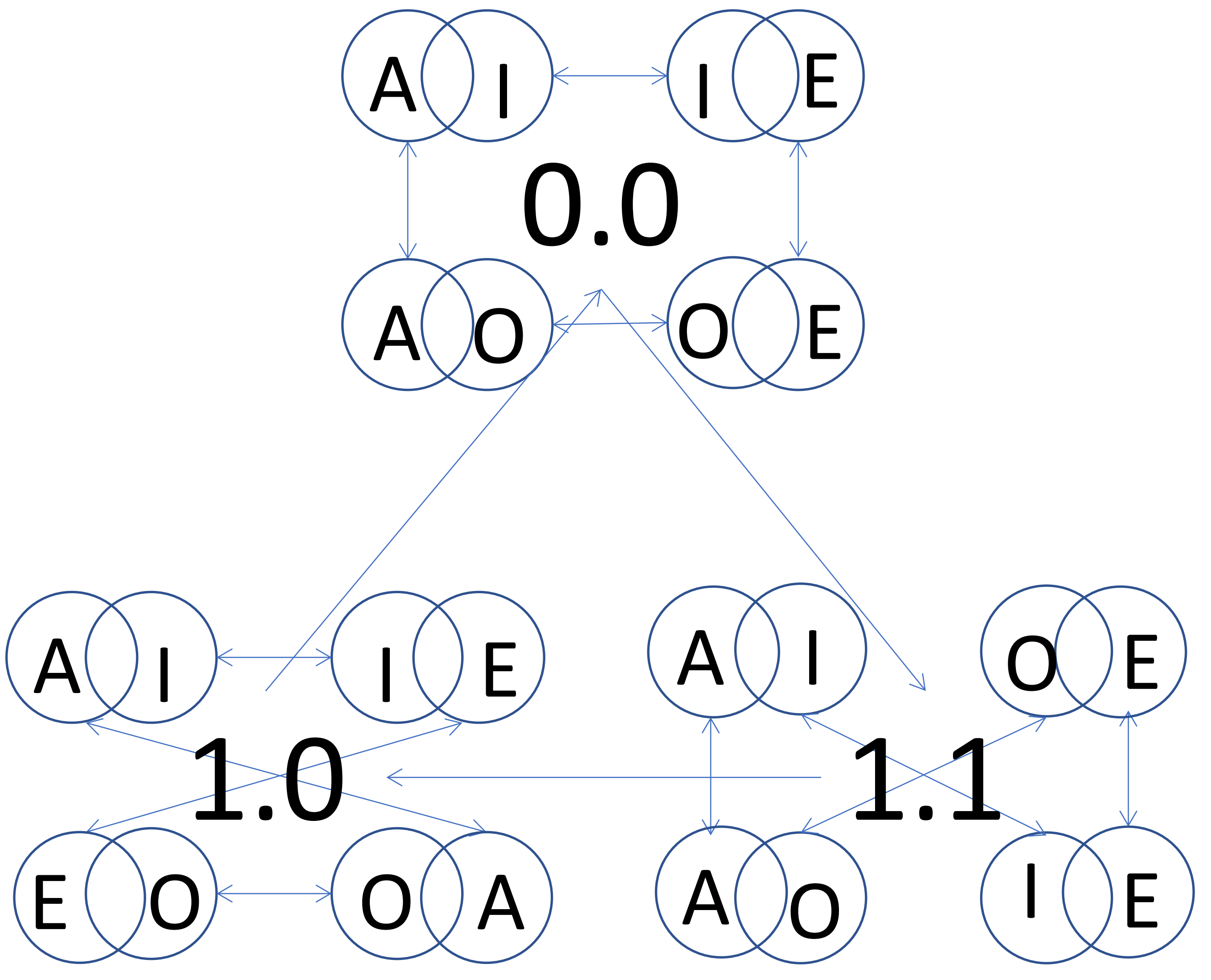




Федорченко Михаил Валерьевич



Fedorchenko Mikhail Valerievich



Fedorchenko Mikhail Valerievich
Need confirmation or proof

To address your request, it appears you're working with a set of logical implications involving the vowels A, E, I, and O. The goal seems to be to adjust or correct the implications to achieve a specific outcome or result. Let's analyze the given implications, clarify their meanings, and see how we can arrive at a consistent set of statements.

Analyzing the Implications

1. **Understanding the Implications**:
  - **0.0**:
    - **A and I implies I and E**: If both A and I are true, then I and E must also be true.
    - **A and O implies O and E**: If both A and O are true, then O and E must also be true.
  - **1.0**:
    - **A and I implies I and E**: This is the same as in 0.0.
    - **E and O implies O and A**: If both E and O are true, then O and A must also be true.
  - **1.1**:
    - **A and I implies O and E**: If both A and I are true, then O and E must also be true.
    - **A and O implies I and E**: If both A and O are true, then I and E must also be true.

Logical Consistency Check

To proceed, we need to ensure that the implications do not contradict each other. Let's summarize the implications and check for consistency:

Summary of Implications
- **From 0.0**:
  - (1) $A \wedge I \rightarrow I \wedge E$
  - (2) $A \wedge O \rightarrow O \wedge E$

- **From 1.0**:
  - (3) $A \wedge I \rightarrow I \wedge E$ (same as 0.0)
  - (4) $E \wedge O \rightarrow O \wedge A$

- **From 1.1**:
  - (5) $A \wedge I \rightarrow O \wedge E$
  - (6) $A \wedge O \rightarrow I \wedge E$

Resolving Conflicts

From the implications:
- (1) and (3) are consistent with each other.
- (2) and (4) introduce a new relationship between E and O, suggesting that they are interdependent.
- (5) from 1.1 contradicts (1) and (3) because it suggests that A and I lead to O and E, while (1) implies they lead to I and E.

Proposed Corrections

To resolve the inconsistencies, we can redefine the implications as follows:

1. **Unified Implications**:
  - **A and I implies I and E**: This can remain, as it is consistent across the versions.
  - **A and O implies O and E**: This can remain as well.
  - **E and O implies O and A**: This can also stay, as it introduces a new relationship.
  - **A and I implies O and E**: This should be removed or altered since it contradicts the previous implications.

2. **Final Set of Implications**:
  - **A and I implies I and E**: (1)
  - **A and O implies O and E**: (2)
  - **E and O implies O and A**: (4)
  - **A and O implies I and E**: (6) can remain as a separate condition.

Conclusion

The final corrected set of implications could be:

- **A and I implies I and E**.
- **A and O implies O and E**.
- **E and O implies O and A**.
- **A and O implies I and E**.

This set maintains internal consistency while allowing for the relationships between the vowels to coexist without contradictions. If you have further specifications or a different desired outcome, please provide more context, and I can adjust the implications accordingly.

Here's an **expanded table** showing all **32 combinations** of the five-dimensional Boolean states (**0** and **1**) mapped to Aristotelian logic:

Logic Mapping of 5 Binary Dimensions

State	**Binary**	**Aristotelian Mapping**	**Description**
State 0	0.0.0.0.0	Universal Affirmative (A)	Fully universal, no particulars.
State 1	0.0.0.0.1	Universal Negative (E)	Partly negative, final shift.
State 2	0.0.0.1.0	Universal Affirmative (A)	Single particular shift.
State 3	0.0.0.1.1	Particular Affirmative (I)	Single negative affirmation.
State 4	0.0.1.0.0	Universal Affirmative (A)	Shift at second dimension.
State 5	0.0.1.0.1	Particular Negative (O)	Balanced state shift.
State 6	0.0.1.1.0	Universal Negative (E)	Dual transition.
State 7	0.0.1.1.1	Particular Affirmative (I)	Particular dominance emerges.
State 8	0.1.0.0.0	Universal Affirmative (A)	Shift at third dimension.
State 9	0.1.0.0.1	Particular Negative (O)	Universal with partial denial.
State 10	0.1.0.1.0	Universal Negative (E)	Midway negative transition.
State 11	0.1.0.1.1	Particular Affirmative (I)	Affirmative particular prevails.
State 12	0.1.1.0.0	Universal Affirmative (A)	Triple affirmation state.
State 13	0.1.1.0.1	Particular Negative (O)	Balance with universal denial.
State 14	0.1.1.1.0	Universal Negative (E)	Strong negative presence.
State 15	0.1.1.1.1	Particular Affirmative (I)	Particular logic confirmed.
State 16	1.0.0.0.0	Universal Affirmative (A)	Fourth dimension shift begins.
State 17	1.0.0.0.1	Universal Negative (E)	Minor final particular denial.
State 18	1.0.0.1.0	Universal Affirmative (A)	Positive dominant movement.
State 19	1.0.0.1.1	Particular Negative (O)	Positive and negative balance.
State 20	1.0.1.0.0	Universal Affirmative (A)	Strong universality.
State 21	1.0.1.0.1	Universal Negative (E)	Negation at midway.
State 22	1.0.1.1.0	Particular Negative (O)	Affirmative denial emerges.
State 23	1.0.1.1.1	Particular Affirmative (I)	Dominant particular state.
State 24	1.1.0.0.0	Universal Affirmative (A)	Affirmation at all levels.
State 25	1.1.0.0.1	Universal Negative (E)	Subtle particular denial shift.
State 26	1.1.0.1.0	Universal Affirmative (A)	Near dominance state.
State 27	1.1.0.1.1	Particular Negative (O)	Affirmation-neutrality.
State 28	1.1.1.0.0	Universal Affirmative (A)	Fully universal finality.
State 29	1.1.1.0.1	Universal Negative (E)	Negation dominates.
State 30	1.1.1.1.0	Particular Negative (O)	Near-particular logic.
State 31	1.1.1.1.1	Particular Affirmative (I)	Fully particular state achieved.

Key Observations:
1. **State 0 (0.0.0.0.0)**: Represents the **most universal (A)** logical state.
2. **State 31 (1.1.1.1.1)**: Represents the **most particular (I)** logical state.
3. **Transitions**:
   - Moving from **0 to 1** in any binary position introduces a **particular element** (affirmation or negation).
   - **Balanced states** (like 0.0.1.0.1 or 1.0.1.0.1) represent combinations of affirmation and negation.

Diagram Integration:
This table can now serve as the **foundation for the expanded logic cube**:
- Each binary node (e.g., `0.1.1.0.0`) corresponds to a logical state.
- Connections between nodes can show **logical progression** (affirmative → negative, universal → particular).

Would you like me to **refine or integrate this table into a visual structure** (e.g., a new logic map or cube diagram)? Let me know how to proceed further!