Informal Intro to "Gold"-Style Language Learning

—

With a Sample Target of RAIR-Lab Interest

(or: Human Language Acquisition is Hard!; Sorry Nonhuman Animals)

Selmer Bringsjord

Naveen Sundar Govindarajulu

Are Humans Rational?

RPI

11/18/19

See *Systems That Learn: An Introduction to Learning Theory* (2nd ed). Adapted from Bringsjord-Govindarajulu seminar *Computational Learning Theory & Science*.

How would *you* formalize/ model language learning in the human case?

Informal Intro to Gold-style Learning …

Formalization of ...

Formalization of ...

1. *theoretically possible realities*;

Formalization of ...

1. *theoretically possible realities*;
2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality *is* reality);

Formalization of ...

1. *theoretically possible realities*;
2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality *is* reality);
3. *the data about any given theoretically possible reality, were it actual* (the list corresponds to things the language learner perceives in the empirical world);

Formalization of ...

1. *theoretically possible realities*;
2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality *is* reality);
3. *the data about any given theoretically possible reality, were it actual* (the list corresponds to things the language learner perceives in the empirical world);
4. the (*language, machine, system, …) learner;*

Formalization of ...

1. *theoretically possible realities*;
2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality *is* reality);
3. *the data about any given theoretically possible reality, were it actual* (the list corresponds to things the language learner perceives in the empirical world);
4. the (*language, machine, system, …) learner;*
5. *successful/unsuccessful behavior by a learner trying to learn a given, possible reality*.

Simple Game Setup

Simple Game Setup

- A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let *D* be the set of such sets.

Simple Game Setup

- A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let *D* be the set of such sets.
- E.g., "all positive even integers." I.e., {2, 4, 6, 8, ...}. These are the aforementioned *theoretically possible realities*.

Simple Game Setup

- A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let *D* be the set of such sets.
- E.g., "all positive even integers." I.e., {2, 4, 6, 8, ...}. These are the aforementioned *theoretically possible realities*.
- *C* is a proper subset of *D* defined as follows: All sets that contain every positive integer, save for one.

Simple Game Setup

- A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let *D* be the set of such sets.
- E.g., "all positive even integers." I.e., {2, 4, 6, 8, ...}. These are the aforementioned *theoretically possible realities*.
- *C* is a proper subset of *D* defined as follows: All sets that contain every positive integer, save for one.
  - E.g., here is a member of *C*: "all positive integers except for 2." I.e., {1, 3, 4, 5, ...}.

More Setup

More Setup

- Nature: "I've selected a member of *C*. You, (language-learning) Child, must discover the set I have in mind. I shall give you clues, as follows. My secret member shall be ordered in a list *L* that contains all members of my secret set, and I shall then present the members of *L* one at a time to you. Each time I do so, you offer a hypothesis in the form of an English expression that uniquely describes a set of positive integers. You win the game iff you make only a finite number of conjectures, and the last one is correct."

More Setup

- Nature: "I've selected a member of *C*. You, (language-learning) Child, must discover the set I have in mind. I shall give you clues, as follows. My secret member shall be ordered in a list *L* that contains all members of my secret set, and I shall then present the members of *L* one at a time to you. Each time I do so, you offer a hypothesis in the form of an English expression that uniquely describes a set of positive integers. You win the game iff you make only a finite number of conjectures, and the last one is correct."
- Nature: "Let's play!"

More Setup

- Nature: "I've selected a member of *C*. You, (language-learning) Child, must discover the set I have in mind. I shall give you clues, as follows. My secret member shall be ordered in a list *L* that contains all members of my secret set, and I shall then present the members of *L* one at a time to you. Each time I do so, you offer a hypothesis in the form of an English expression that uniquely describes a set of positive integers. You win the game iff you make only a finite number of conjectures, and the last one is correct."
- Nature: "Let's play!"
- Child: "Okay!"

Some Playing

Some Playing

- N: 1.

Some Playing

- N: 1.
- Child: Silence.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (mutiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (mutiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (mutiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.
- N: 14.

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (mutiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.
- N: 14.
- ...

Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (mutiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.
- N: 14.
- ...
- Is Child using some algorithm?

Guessing Algorithm (G)

Guessing Algorithm (*G*)

Let *S* be the set of numbers presented so far. And let *m* be the smallest positive integer that isn't a member of *S*. Output the hypothesis: "All positive integers except for *m*." If this was your last hypothesis, remain silent.

Guessing Algorithm (G)

Let S be the set of numbers presented so far. And let m be the smallest positive integer that isn't a member of S. Output the hypothesis: "All positive integers except for m." If this was your last hypothesis, remain silent.

So, e.g., $G([4, 5, 8, 1]) = 2$.

Theorem I.

Theorem 1.

Regardless of what member of *C* is chosen by Nature at the start of the game, and no matter what list *L* is produced from that member, *G* will win the game for you!

Theorem 1.

Regardless of what member of *C* is chosen by Nature at the start of the game, and no matter what list *L* is produced from that member, *G* will win the game for you!

(Sedulous-and-inquisitive: Prove it!)

One Slight Mod

One Slight Mod

We stipulate that $\mathbf{Z}^+$ is in C.

Theorem 2.

Theorem 2.

There exists no algorithm that is guaranteed to win this new game. More precisely, for every algorithm A there is a set in the (expanded!) set C, and a list L of C, such that A fails to produce a last, correct hypothesis.

Theorem 2.

There exists no algorithm that is guaranteed to win this new game. More precisely, for every algorithm *A* there is a set in the (expanded!) set *C*, and a list *L* of *C*, such that *A* fails to produce a last, correct hypothesis.

(Sedulous-and-inquisitive: Prove it!)

Some Issues to Keep in Mind

Some Issues to Keep in Mind

1. For this class, and where we are in this class, a *language* is no doubt more appropriate than a set of integers as a "possible reality," but the idea is that a set of integers can "code" a language ...

Some Issues to Keep in Mind

1. For this class, and where we are in this class, a *language* is no doubt more appropriate than a set of integers as a "possible reality," but the idea is that a set of integers can "code" a language ...

2. The set of possible realities must on this approach be *countable*. What then about physical quantities whose values are arbitrary real numbers? Isn't this what we see in physics as carried out by real physicists?

Some Issues to Keep in Mind

1. For this class, and where we are in this class, a *language* is no doubt more appropriate than a set of integers as a "possible reality," but the idea is that a set of integers can "code" a language ...

2. The set of possible realities must on this approach be *countable*. What then about physical quantities whose values are arbitrary real numbers? Isn't this what we see in physics as carried out by real physicists?

3. Language-learning Children are assumed here to be *mechanical*. In traditional CLT, language learners are identified with standard computer programs/Turing machines. (Since Selmer thinks that human persons are capable of information processing beyond Turing machines, the assumption here is one he can't ultimately swallow.)

Some Issues to Keep in Mind

1. For this class, and where we are in this class, a *language* is no doubt more appropriate than a set of integers as a "possible reality," but the idea is that a set of integers can "code" a language ...

2. The set of possible realities must on this approach be *countable*. What then about physical quantities whose values are arbitrary real numbers? Isn't this what we see in physics as carried out by real physicists?

3. Language-learning Children are assumed here to be *mechanical*. In traditional CLT, language learners are identified with standard computer programs/Turing machines. (Since Selmer thinks that human persons are capable of information processing beyond Turing machines, the assumption here is one he can't ultimately swallow.)

4. Knowledge is ignored. For example, does the Child in the above game know that some winning hypothesis is a winning hypothesis? If that were the case, then given that knowledge is at least "justified true belief," where is the justification?

Some Issues to Keep in Mind

1. For this class, and where we are in this class, a *language* is no doubt more appropriate than a set of integers as a "possible reality," but the idea is that a set of integers can "code" a language ...

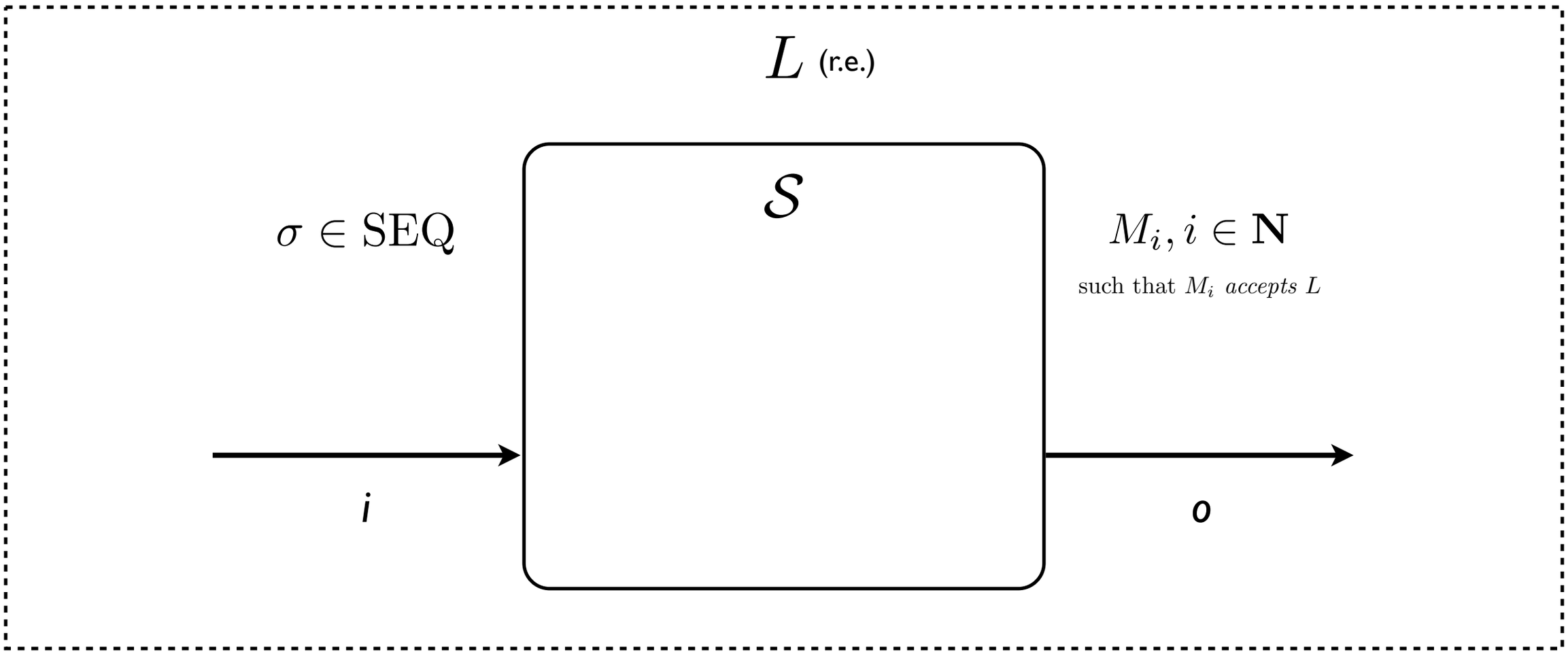
2. The set of possible realities must on this approach be *countable*. What then about physical quantities whose values are arbitrary real numbers? Isn't this what we see in physics as carried out by real physicists?

3. Language-learning Children are assumed here to be *mechanical*. In traditional CLT, language learners are identified with standard computer programs/Turing machines. (Since Selmer thinks that human persons are capable of information processing beyond Turing machines, the assumption here is one he can't ultimately swallow.)

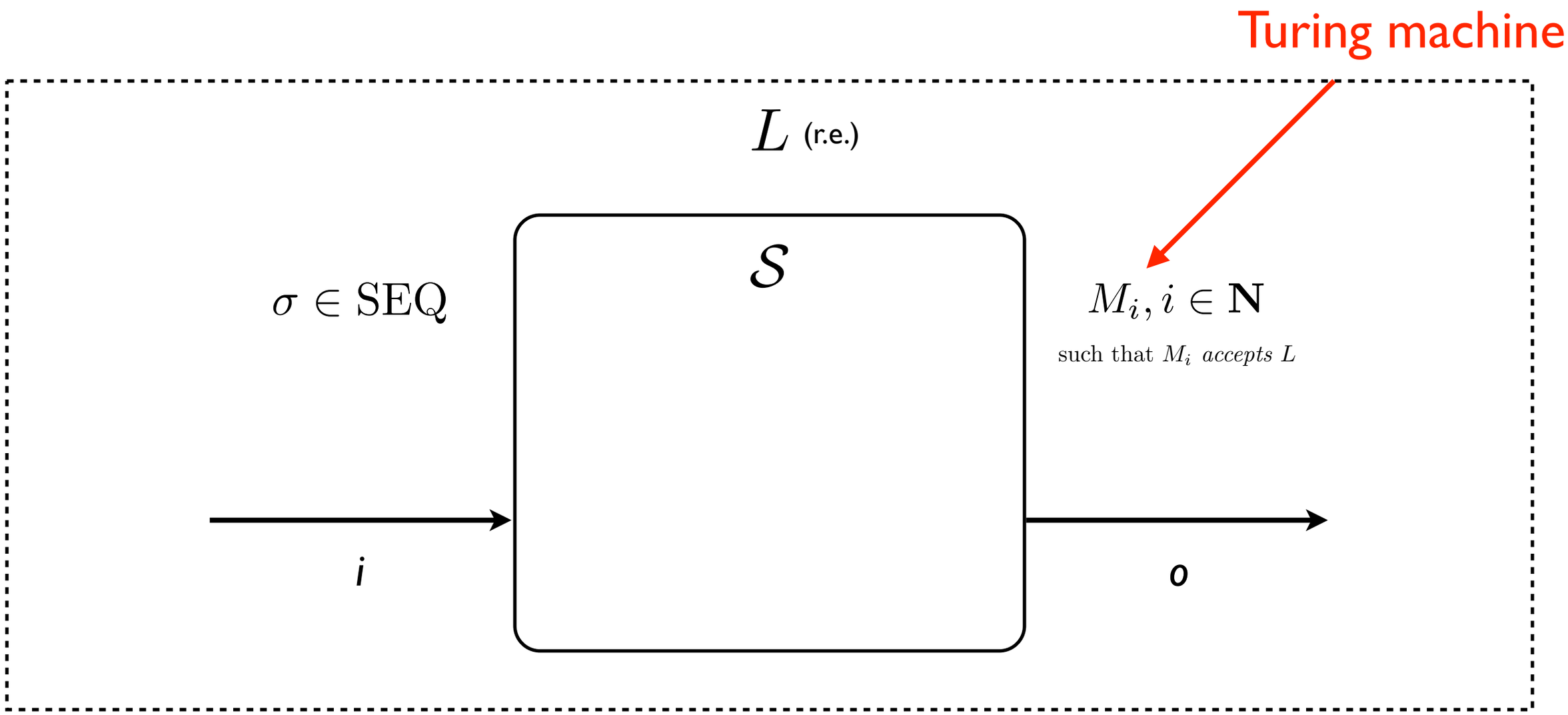
4. Knowledge is ignored. For example, does the Child in the above game know that some winning hypothesis is a winning hypothesis? If that were the case, then given that knowledge is at least "justified true belief," where is the justification?

    - Of course, given our Theorems 1 & 2 from above, *we* can in *some* sense know that the system Child is using will win. But the Child himself/herself doesn't know.

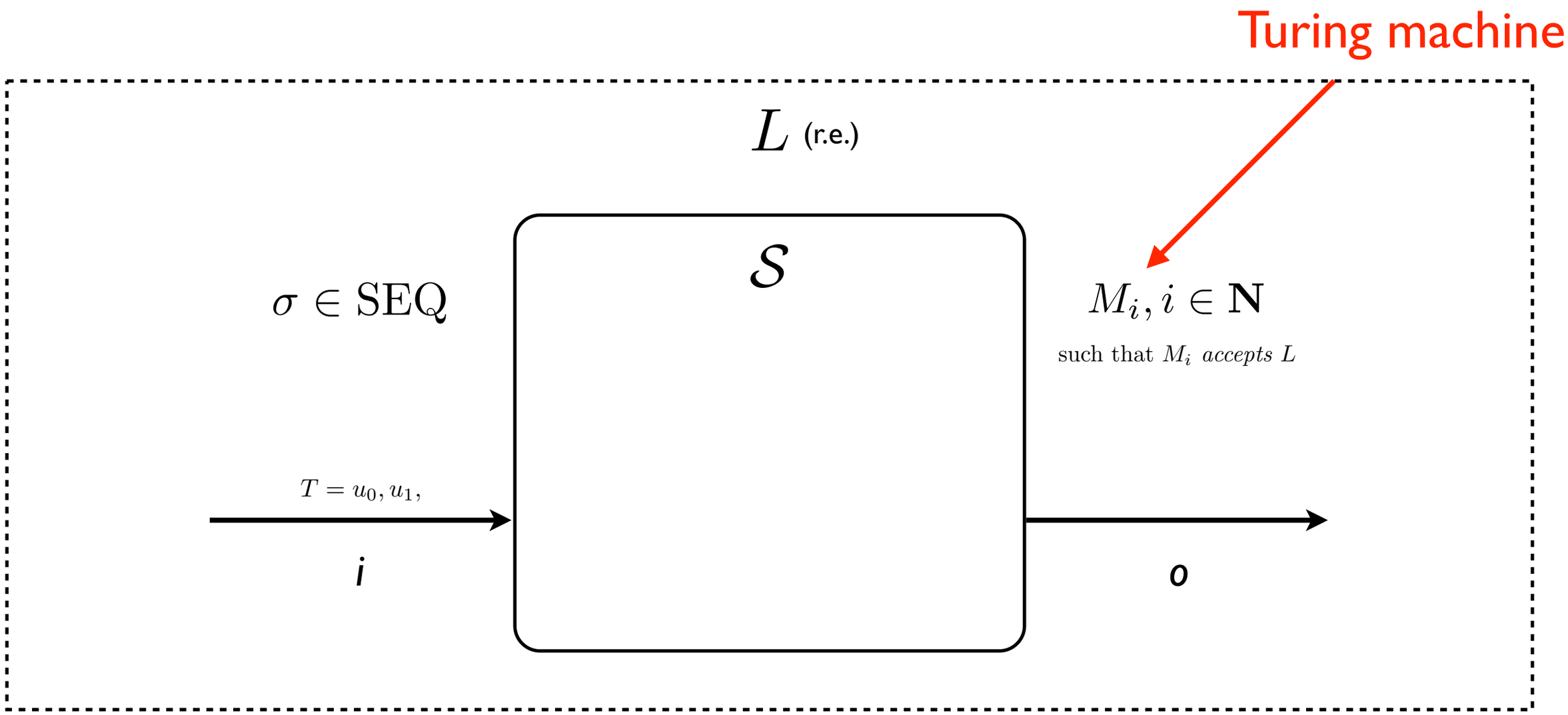
CLT-based Model of Language Learning



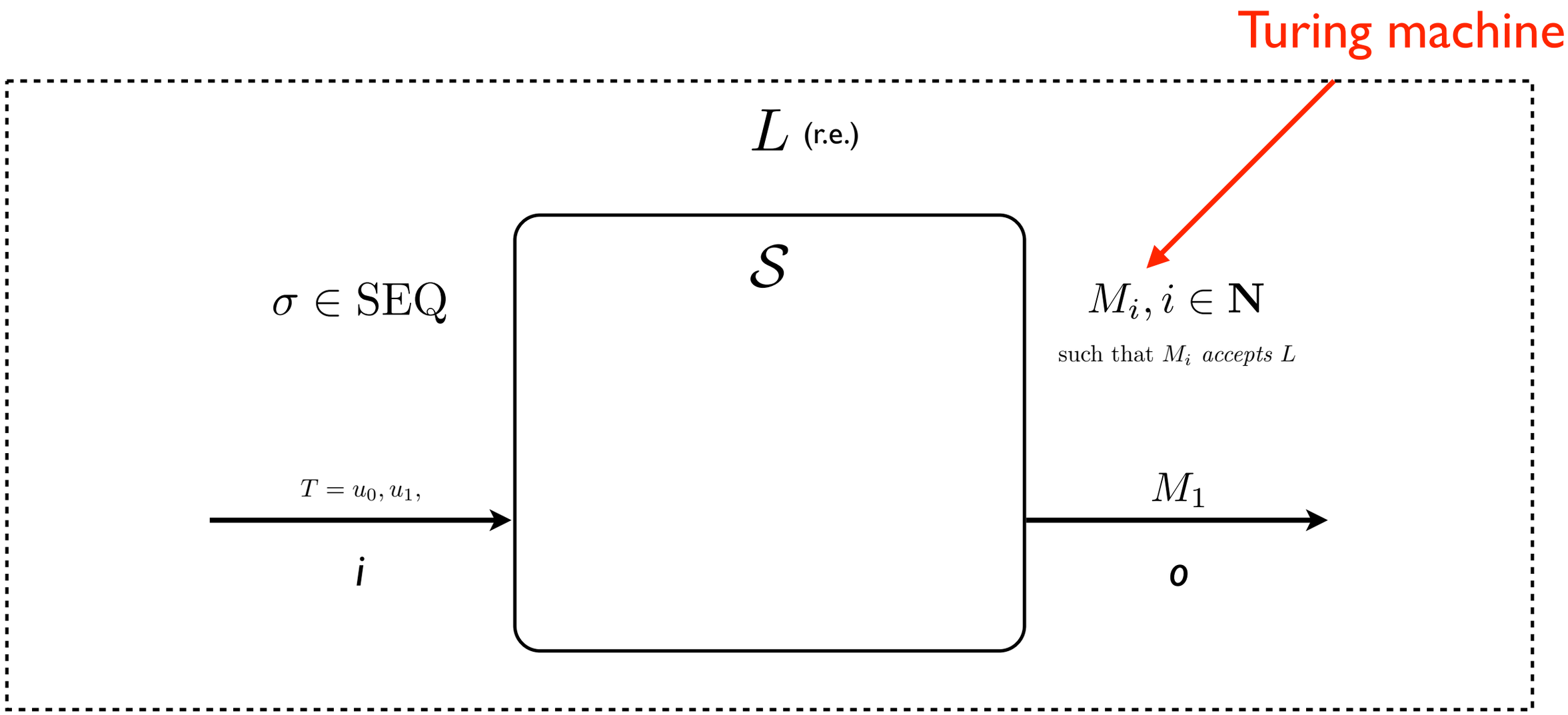
CLT-based Model of Language Learning



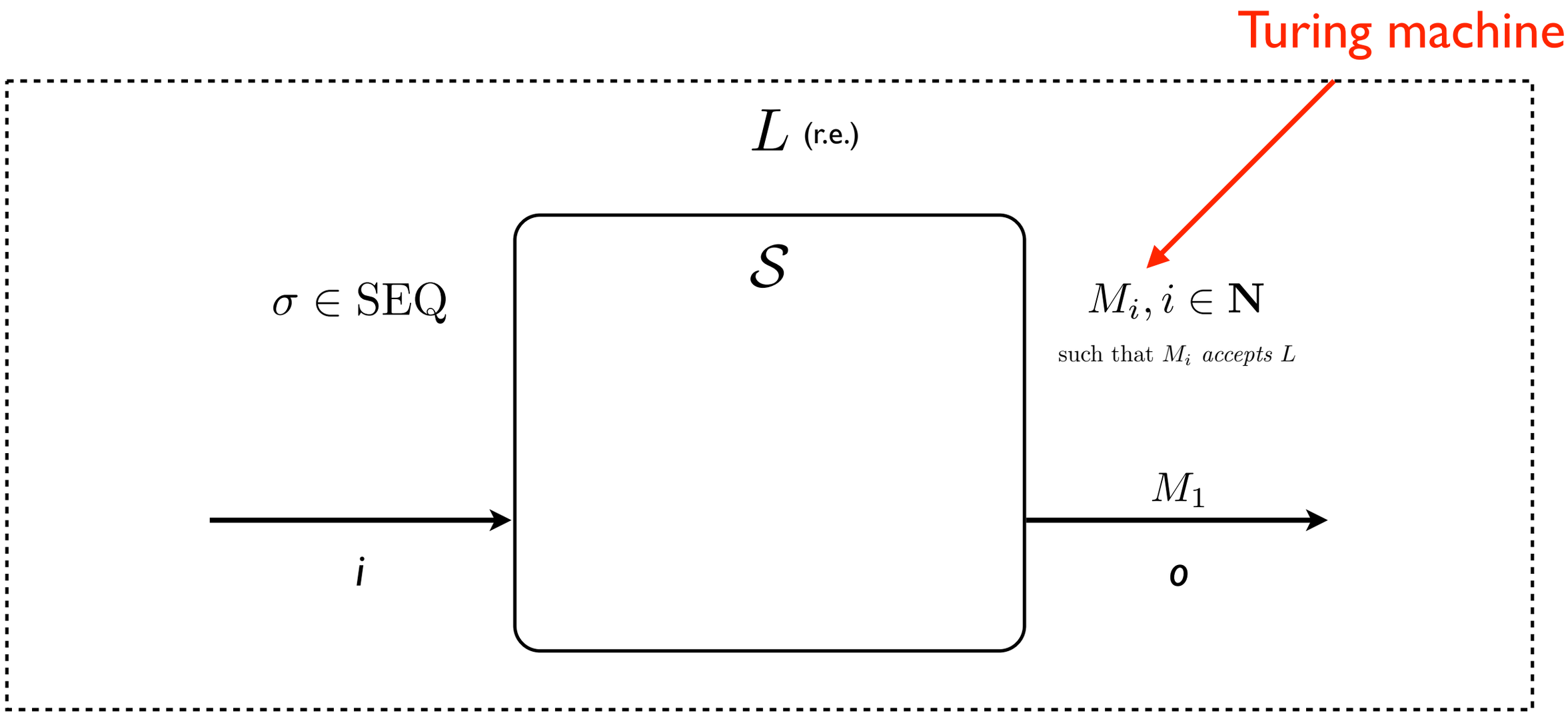
CLT-based Model of Language Learning



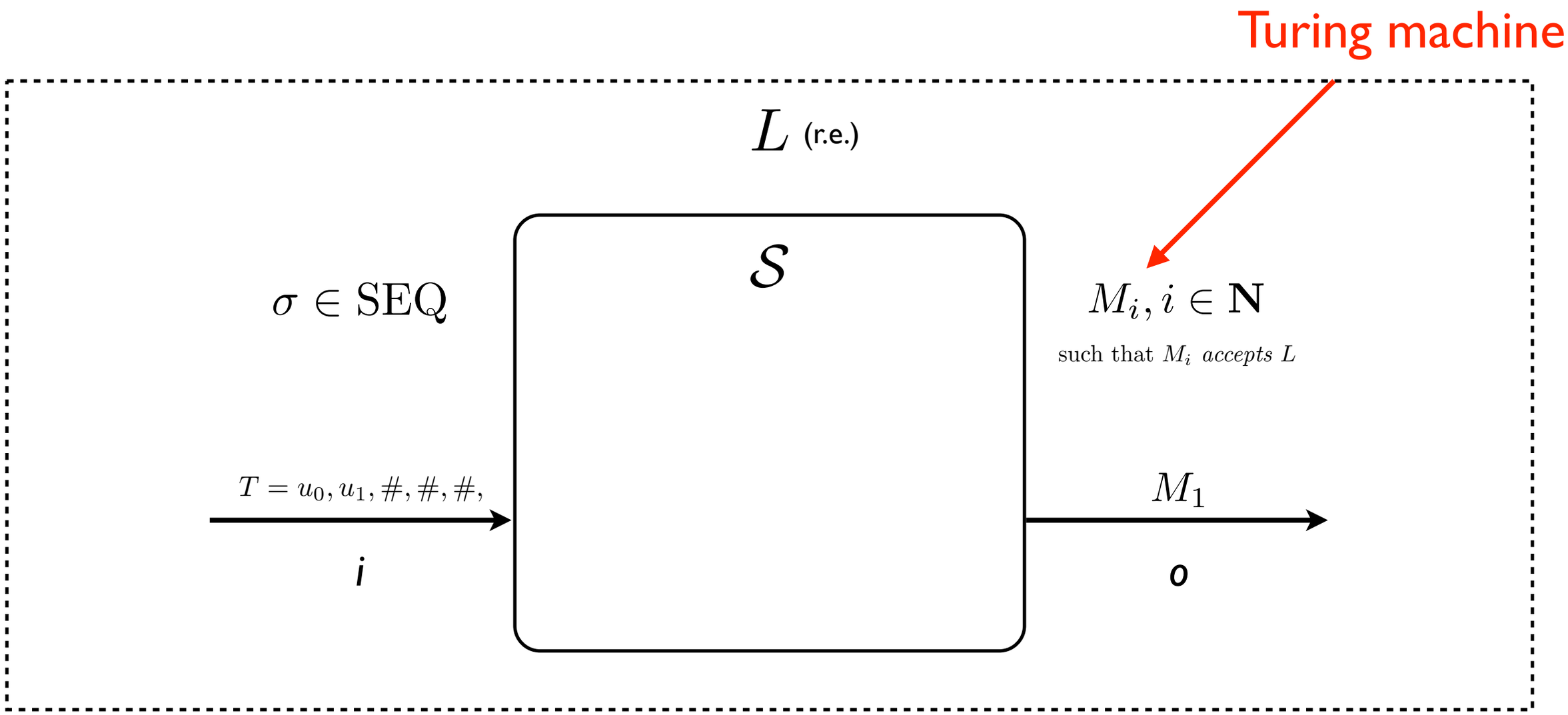
CLT-based Model of Language Learning



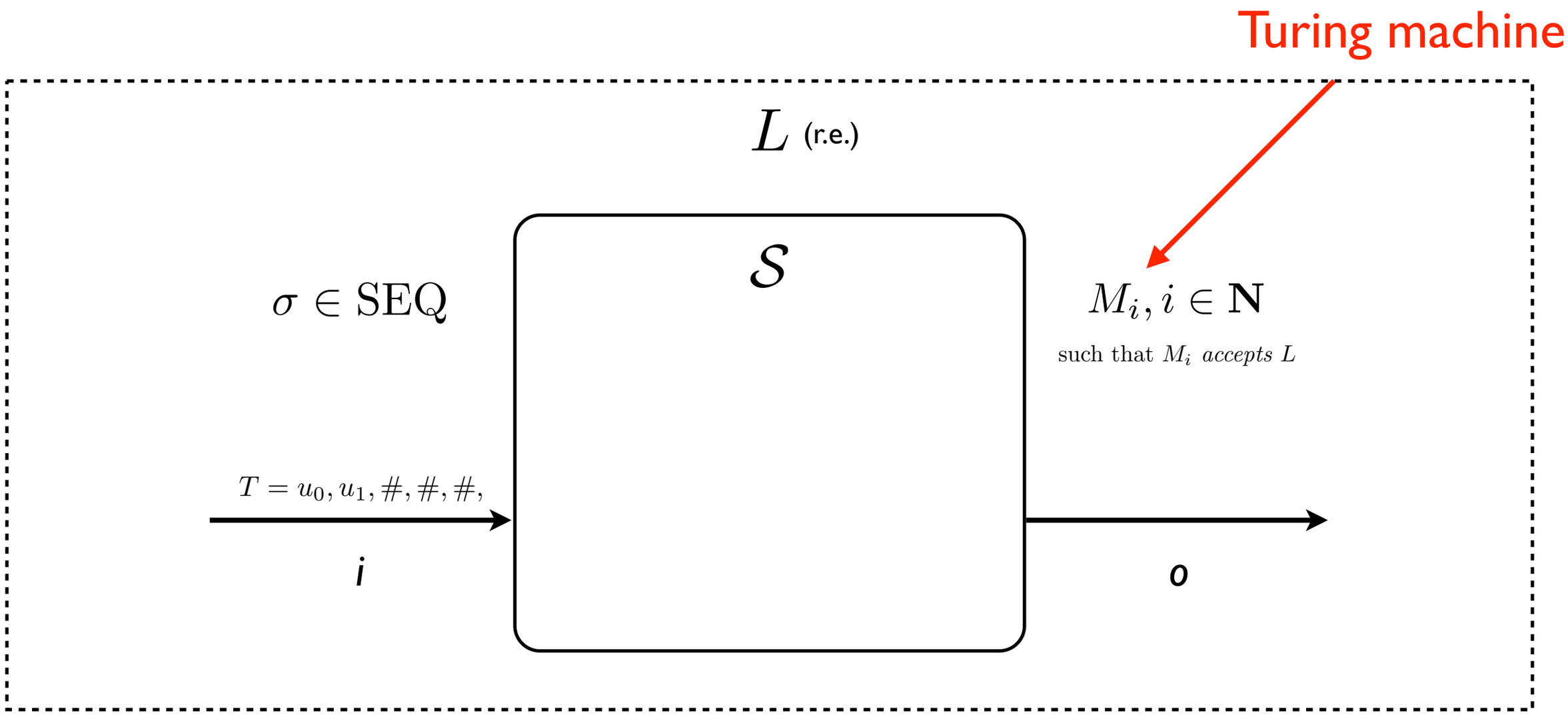
CLT-based Model of Language Learning



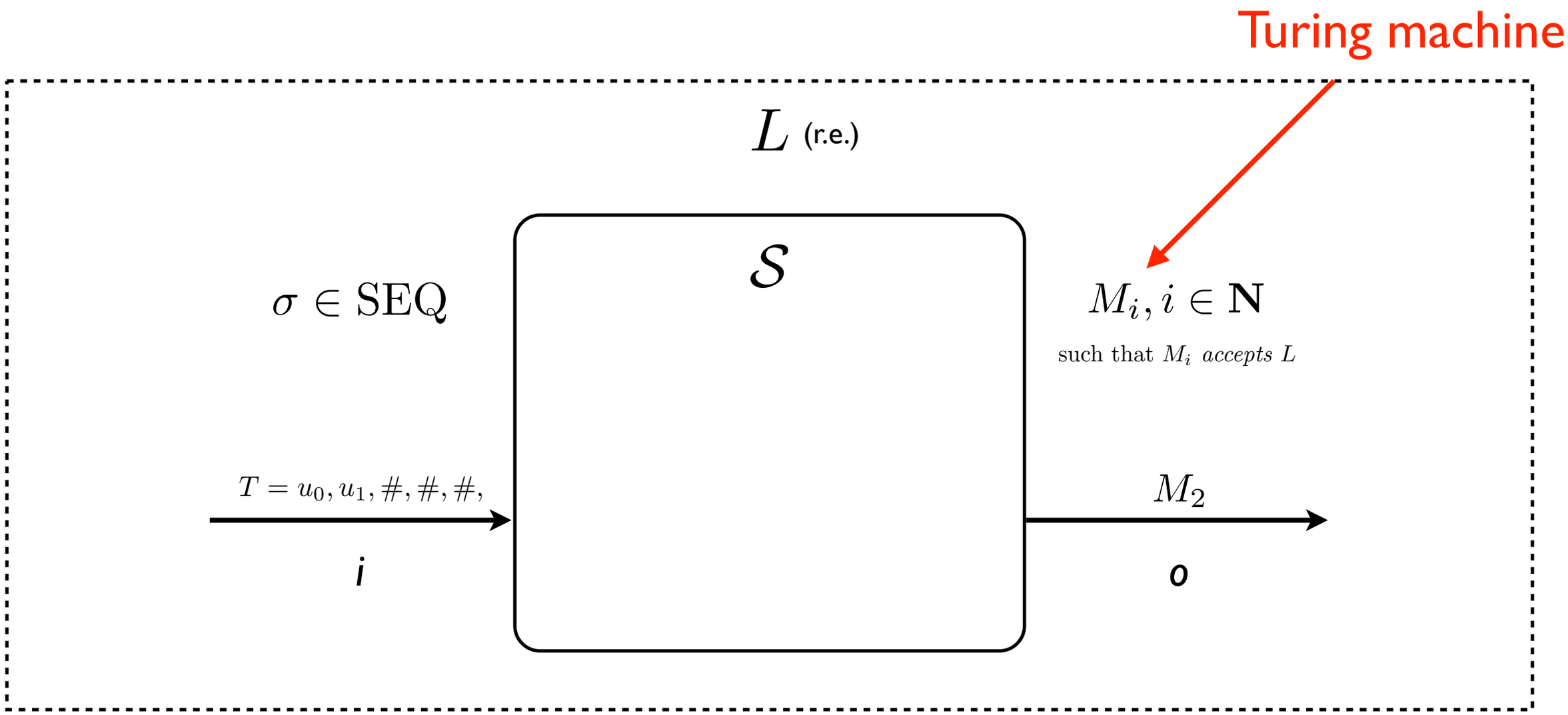
CLT-based Model of Language Learning



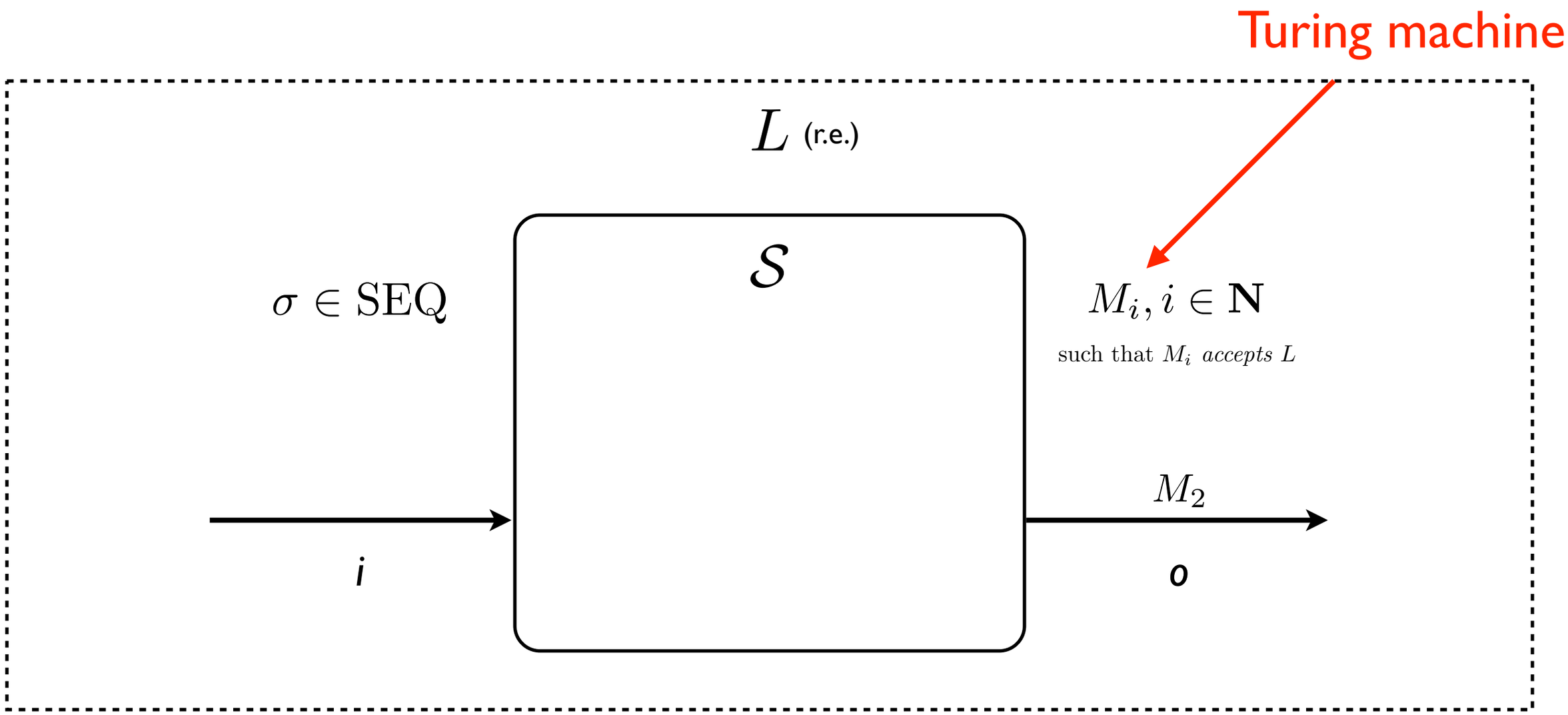
CLT-based Model of Language Learning



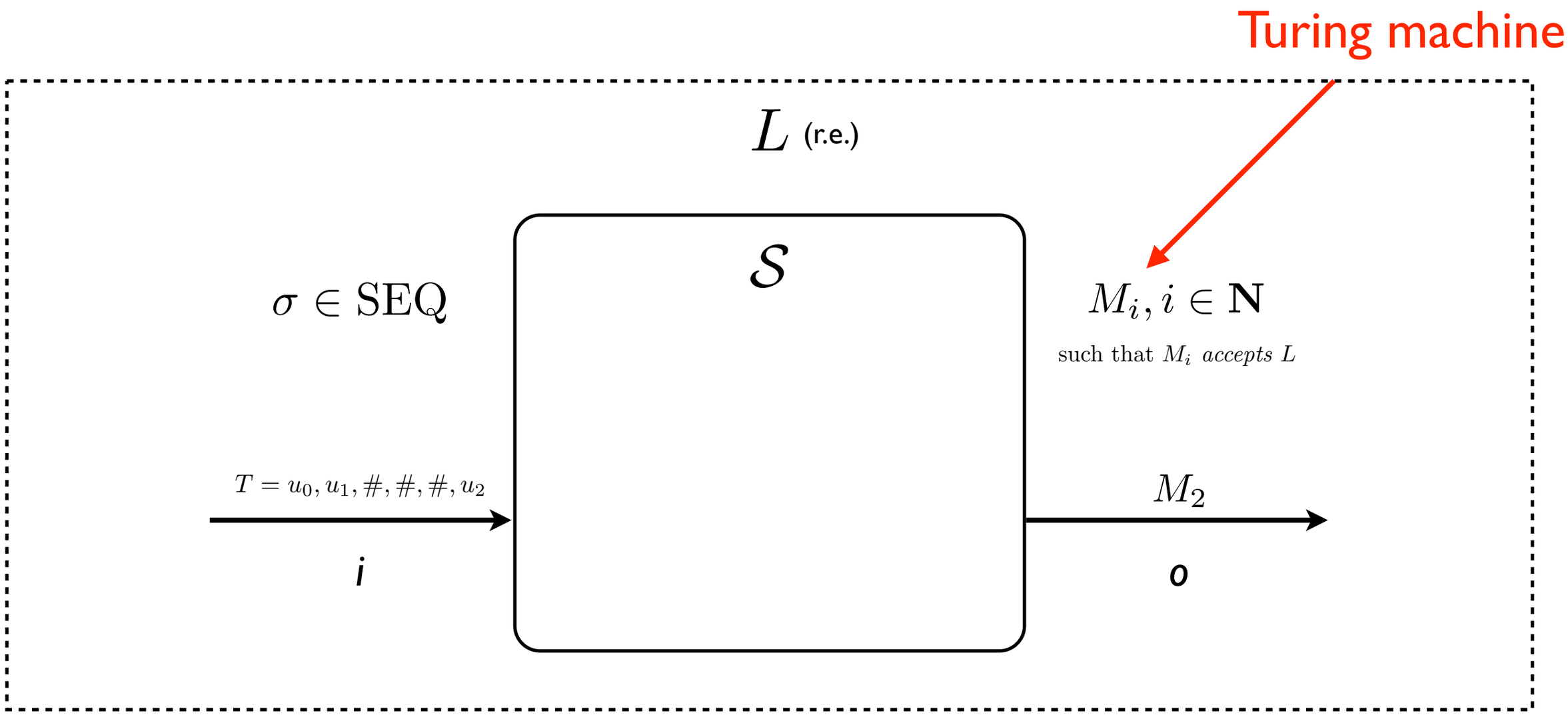
CLT-based Model of Language Learning



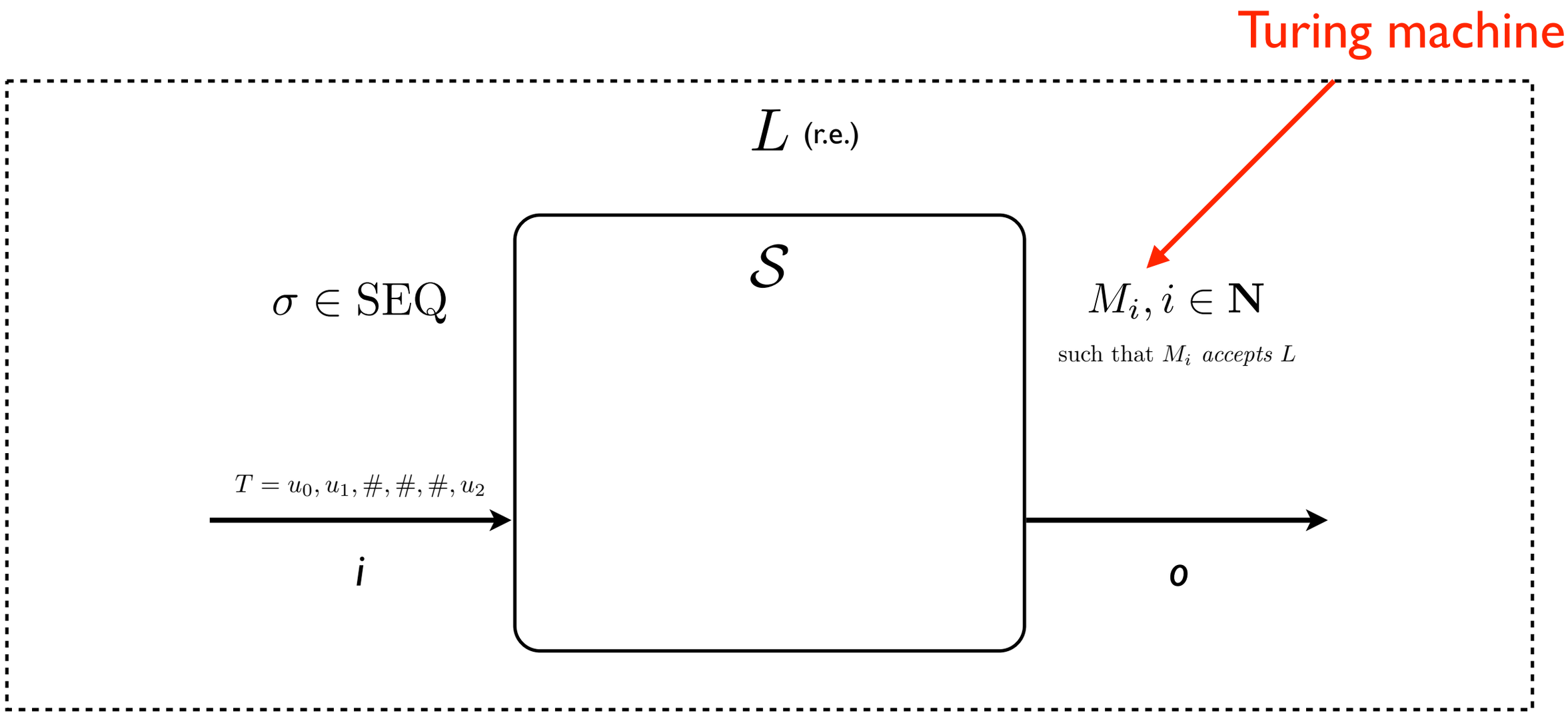
CLT-based Model of Language Learning



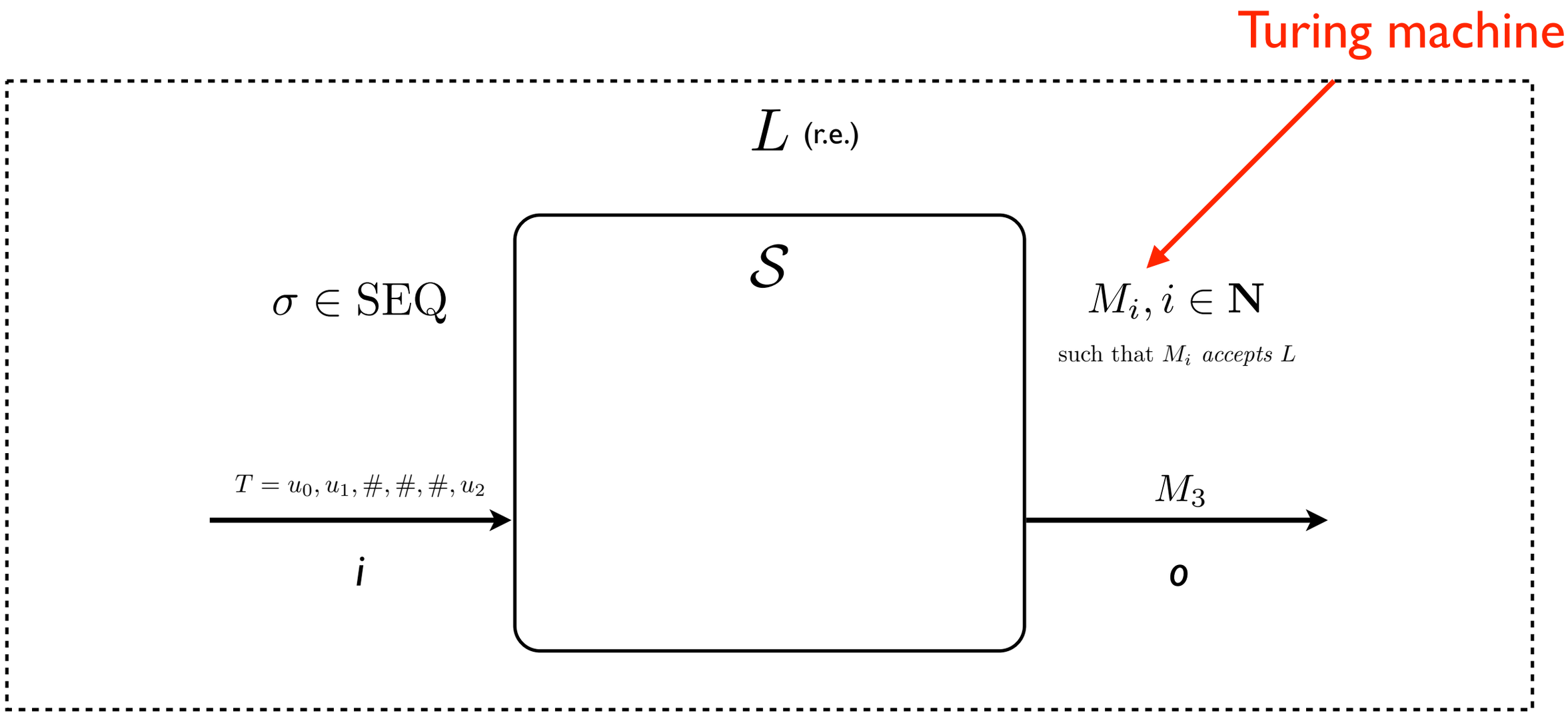
CLT-based Model of Language Learning



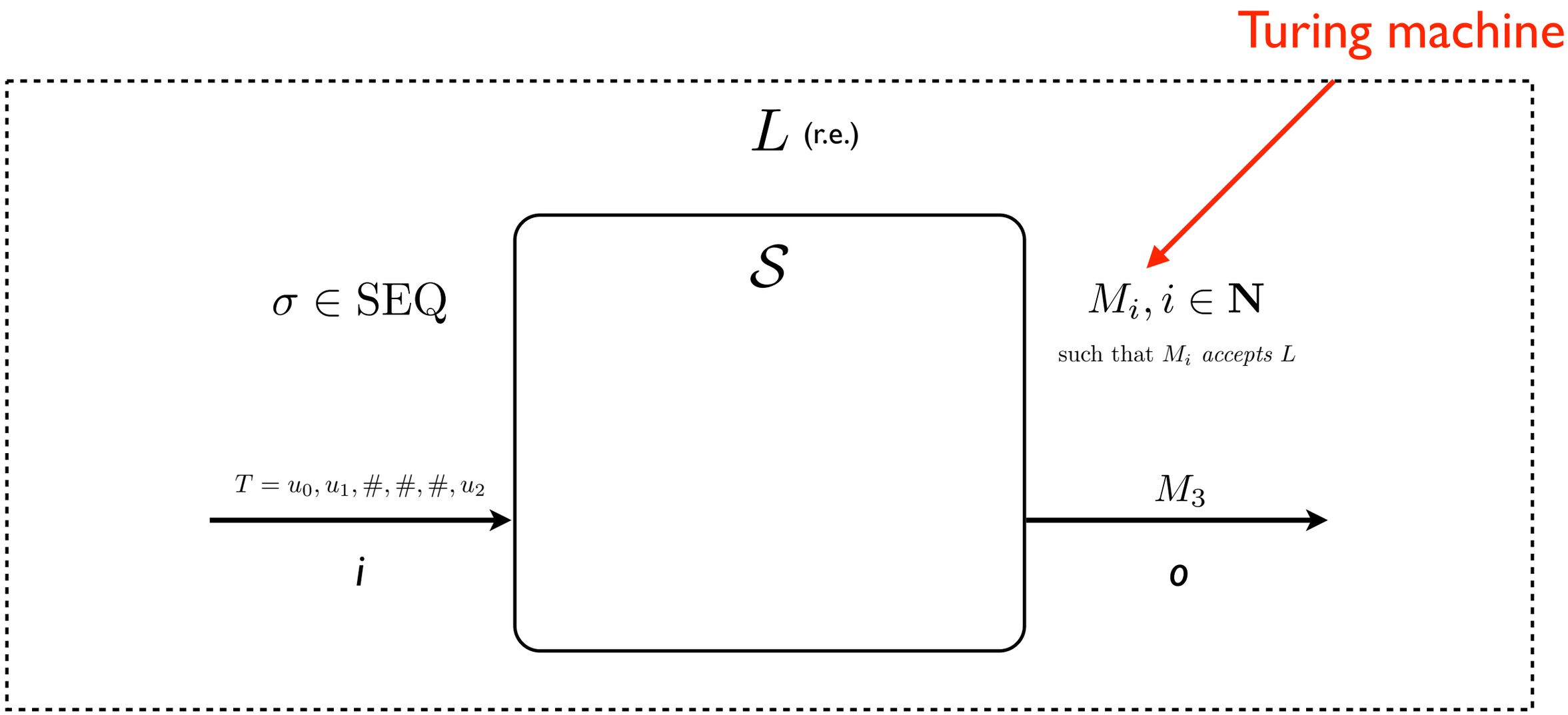
CLT-based Model of Language Learning



CLT-based Model of Language Learning

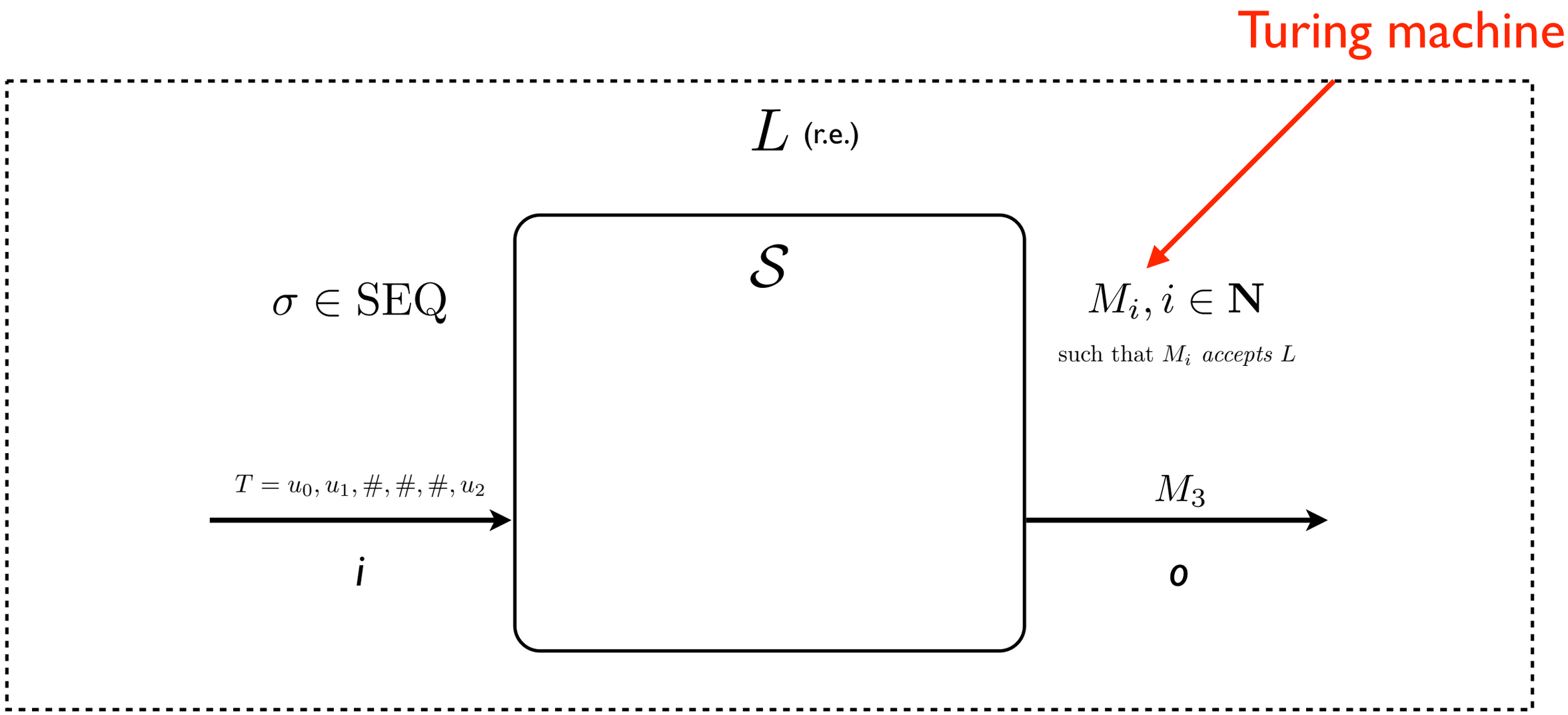


CLT-based Model of Language Learning



And when is identification of *L* achieved?

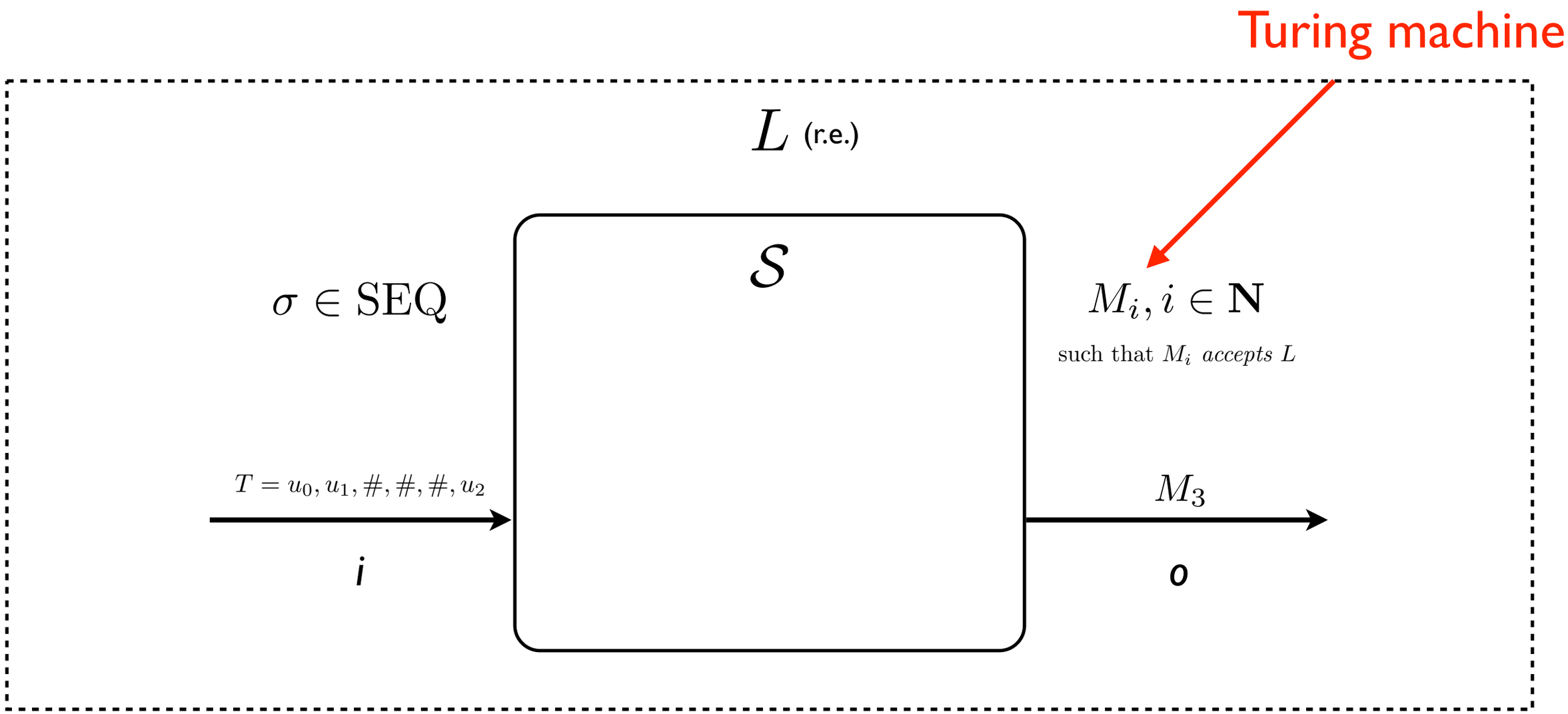
CLT-based Model of Language Learning



And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

CLT-based Model of Language Learning



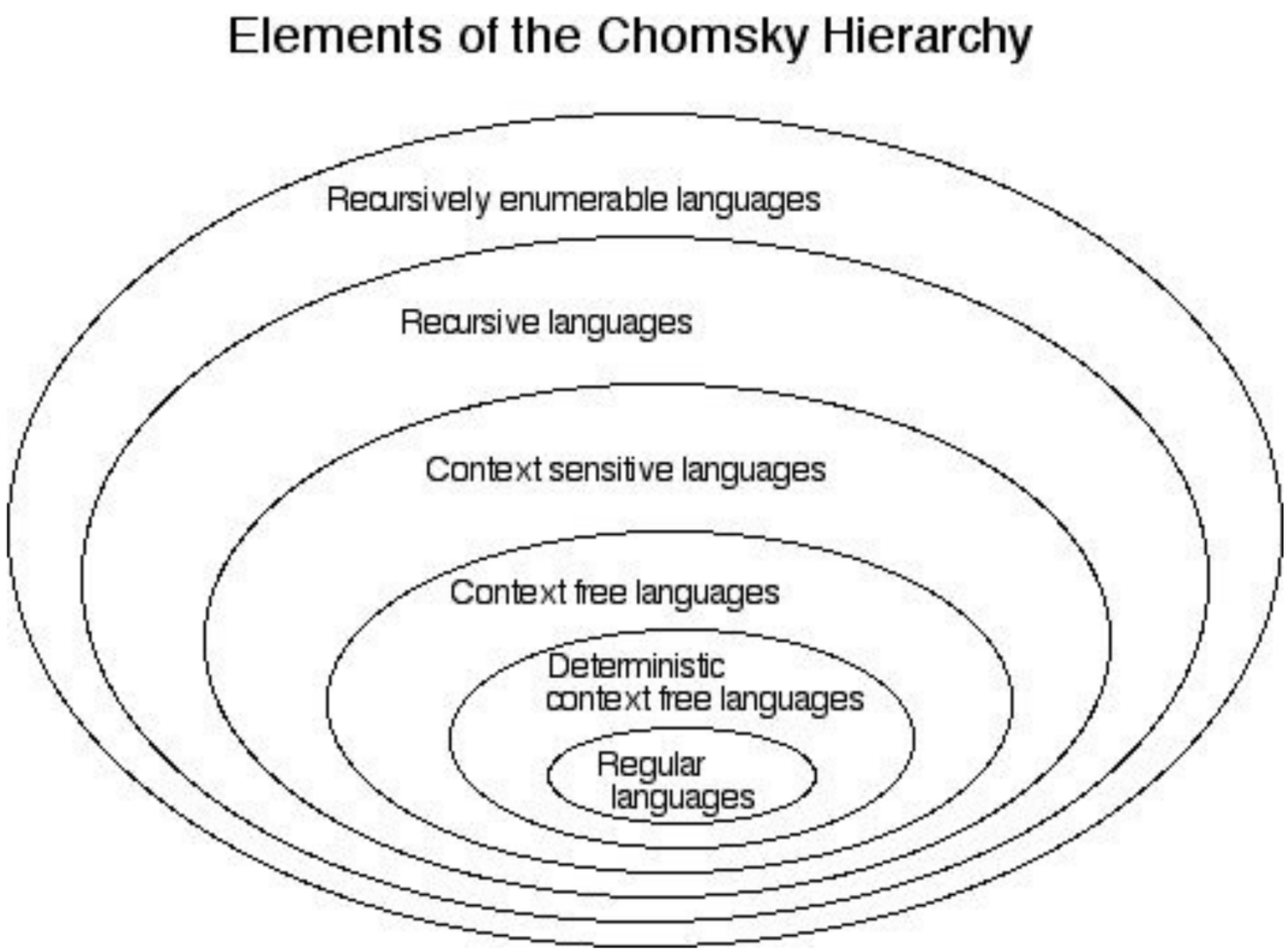
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

RAIR-Lab-relevant Target for Learning …

Chomsky Hierarchy of Languages



How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula}
\end{array}
$$

$$
\begin{array}{lcl}
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term})
\end{array}
$$

$$
\begin{array}{lcl}
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant}
\end{array}
$$

$$
\mathit{Connective} \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
$$

$$
\begin{array}{lcl}
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Sally likes Bill.

`(Likes sally bill)`

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

How About Learning the Grammar of the Pure Predicate Calculus?

Sally likes Bill.

`(Likes sally bill)`

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathtt{Likes}\ \mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathtt{Likes} & & \\
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathtt{Likes}\ \mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

Sally likes Bill's mother.

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\mathtt{Likes} & & \\
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

Sally likes Bill's mother.

Sally likes Bill only if Bill's mother is tall.

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\texttt{Likes} & & \\
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

Sally likes Bill's mother.

Sally likes Bill only if Bill's mother is tall.

Matilda is Bill's super-smart mother.

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\texttt{Likes} & & \\
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

Sally likes Bill's mother.

Sally likes Bill only if Bill's mother is tall.

Matilda is Bill's super-smart mother.

5 plus 5 equals the number 10.

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\texttt{Likes} & & \\
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

Sally likes Bill's mother.

Sally likes Bill only if Bill's mother is tall.

Matilda is Bill's super-smart mother.

5 plus 5 equals the number 10.

...

How About Learning the Grammar of the Pure Predicate Calculus?

$$
\begin{array}{lcl}
\mathit{Formula} & \Rightarrow & \mathit{AtomicFormula} \\
 & | & (\mathit{Formula}\ \mathit{Connective}\ \mathit{Formula}) \\
 & | & \neg\ \mathit{Formula} \\
\hline
\mathit{AtomicFormula} & \Rightarrow & (\mathit{Predicate}\ \mathit{Term}_1 \ldots \mathit{Term}_k) \\
 & | & (\mathit{Term}\ =\ \mathit{Term}) \\
\hline
\mathit{Term} & \Rightarrow & (\mathit{Function}\ \mathit{Term}_1\ \ldots\ \mathit{Term}_k) \\
 & | & \mathit{Constant} \\
\hline
\mathit{Connective} & \Rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\texttt{Likes} & & \\
\mathit{Predicate} & \Rightarrow & P_1 \mid P_2 \mid P_3\ \ldots \\
\mathit{Constant} & \Rightarrow & c_1 \mid c_2 \mid c_3\ \ldots \\
\mathit{Function} & \Rightarrow & f_1 \mid f_2 \mid f_3\ \ldots
\end{array}
$$

Lexicon

Sally likes Bill.

`(Likes sally bill)`

Sally likes Bill and Bill likes Sally.

Sally likes Bill's mother.

Sally likes Bill only if Bill's mother is tall.

Matilda is Bill's super-smart mother.

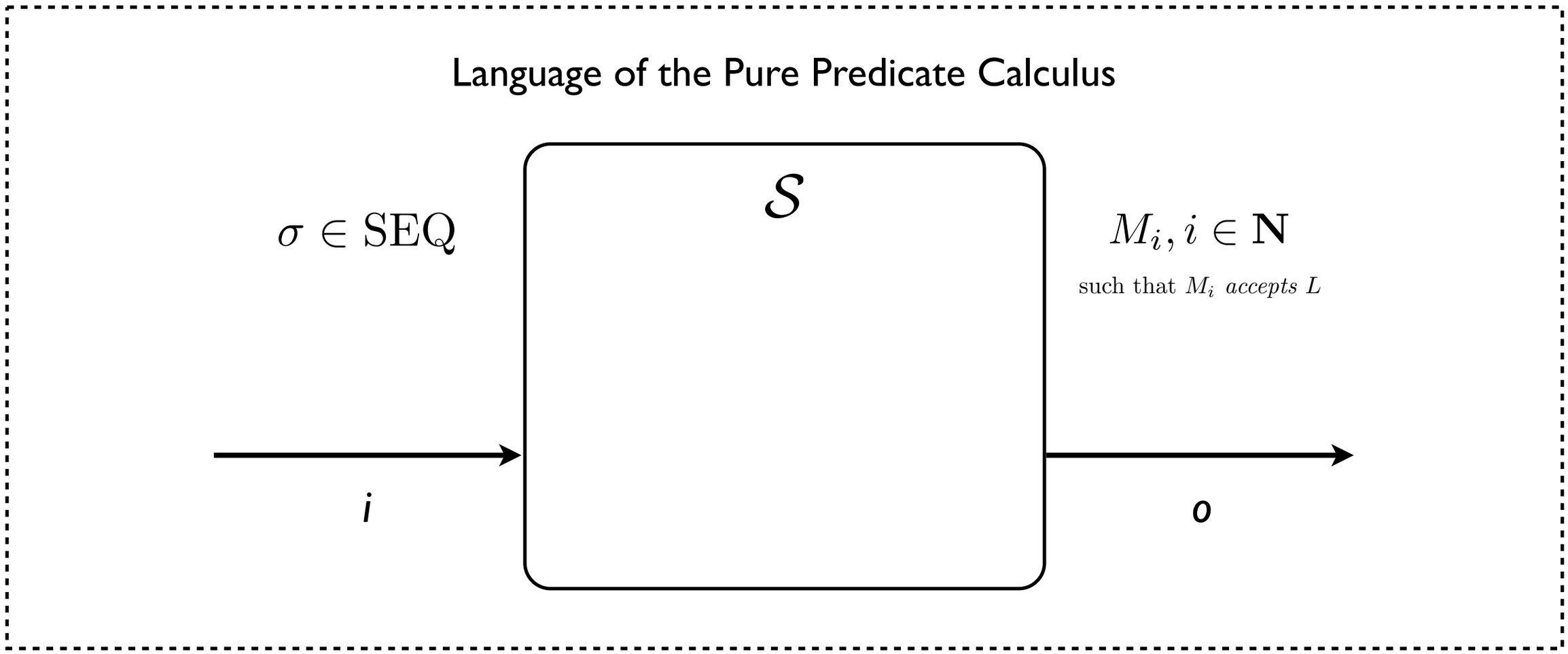
5 plus 5 equals the number 10.

...

Make sure you can simulate a machine that says "Yes that sentence is okay!" whenever it's conforms to this grammar!

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



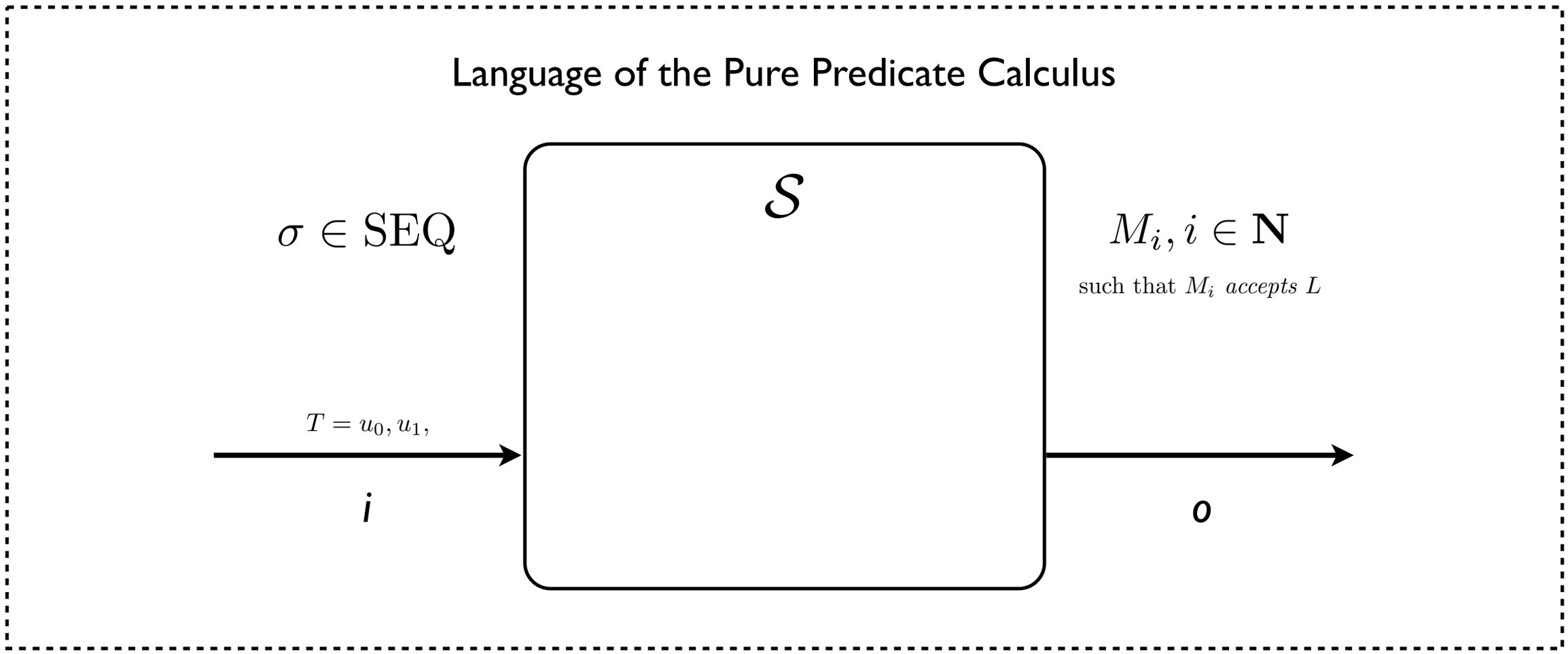
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



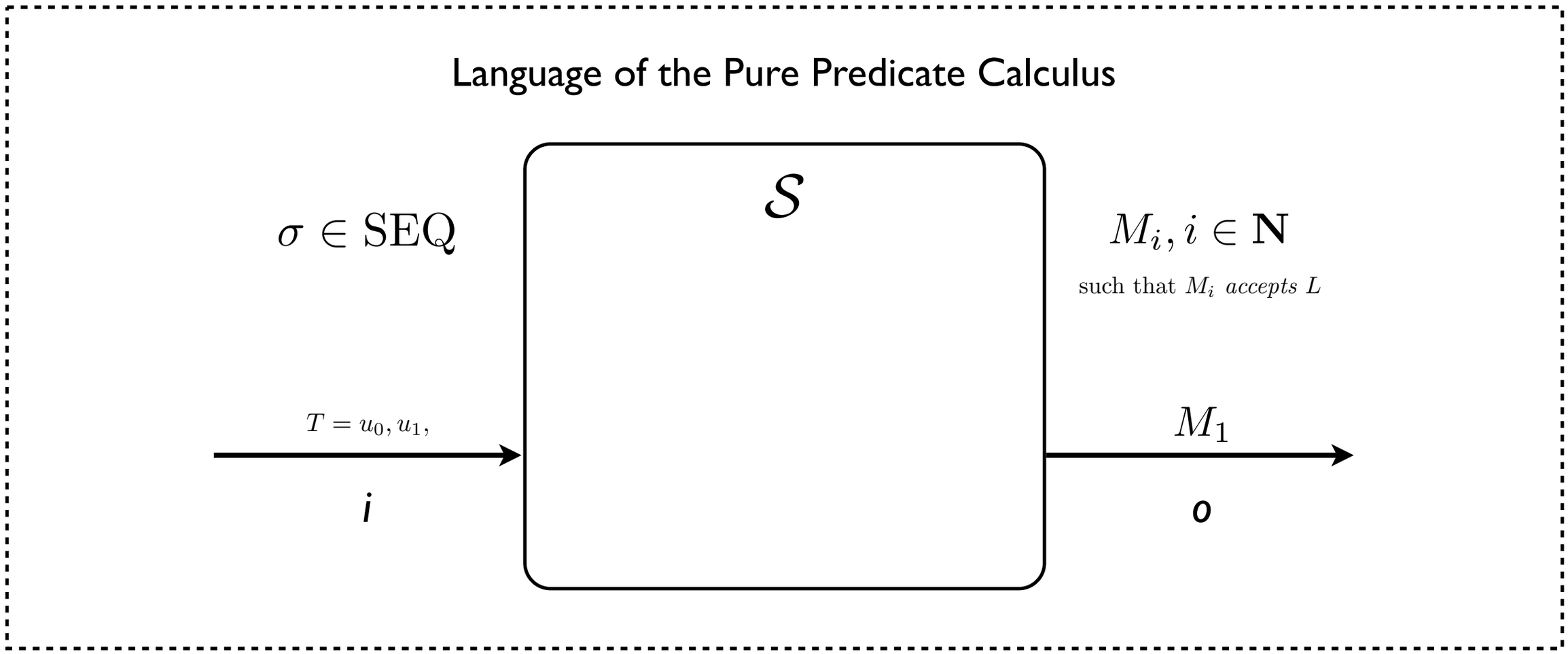
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



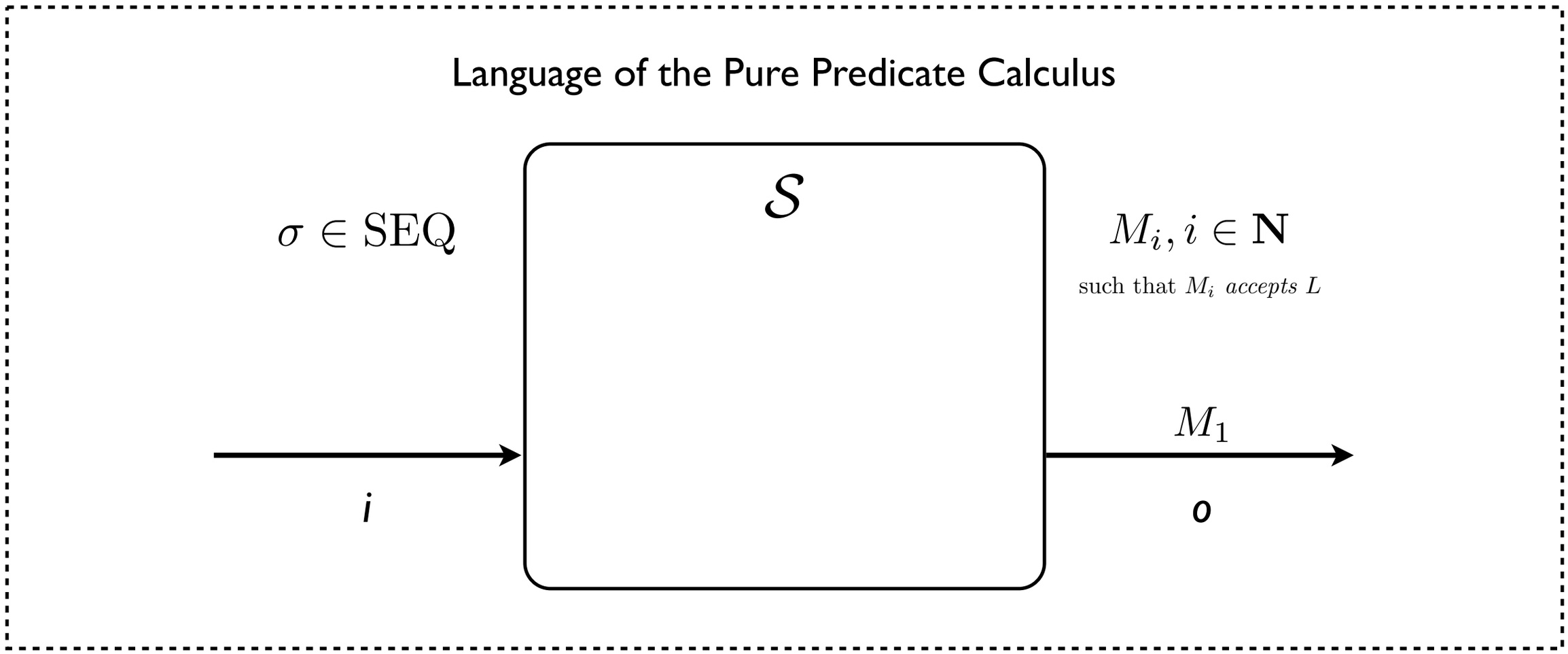
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



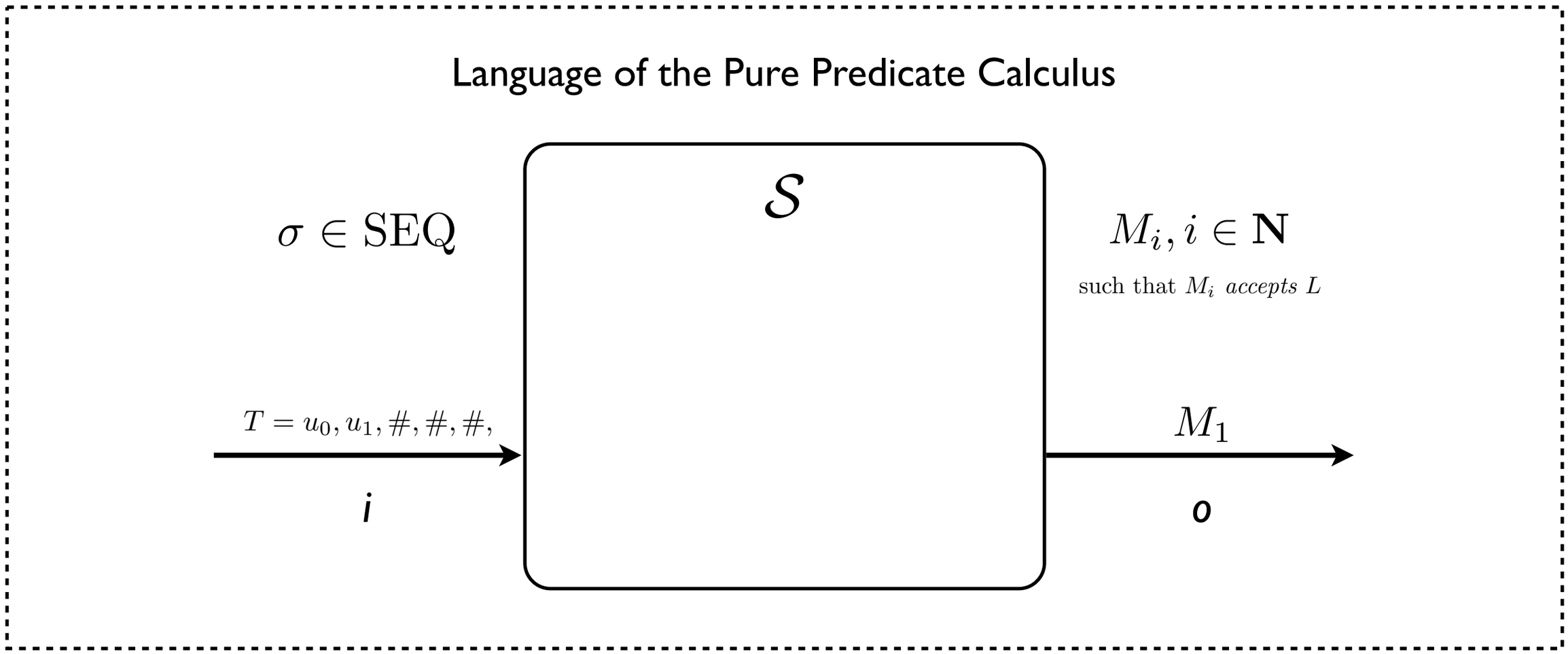
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



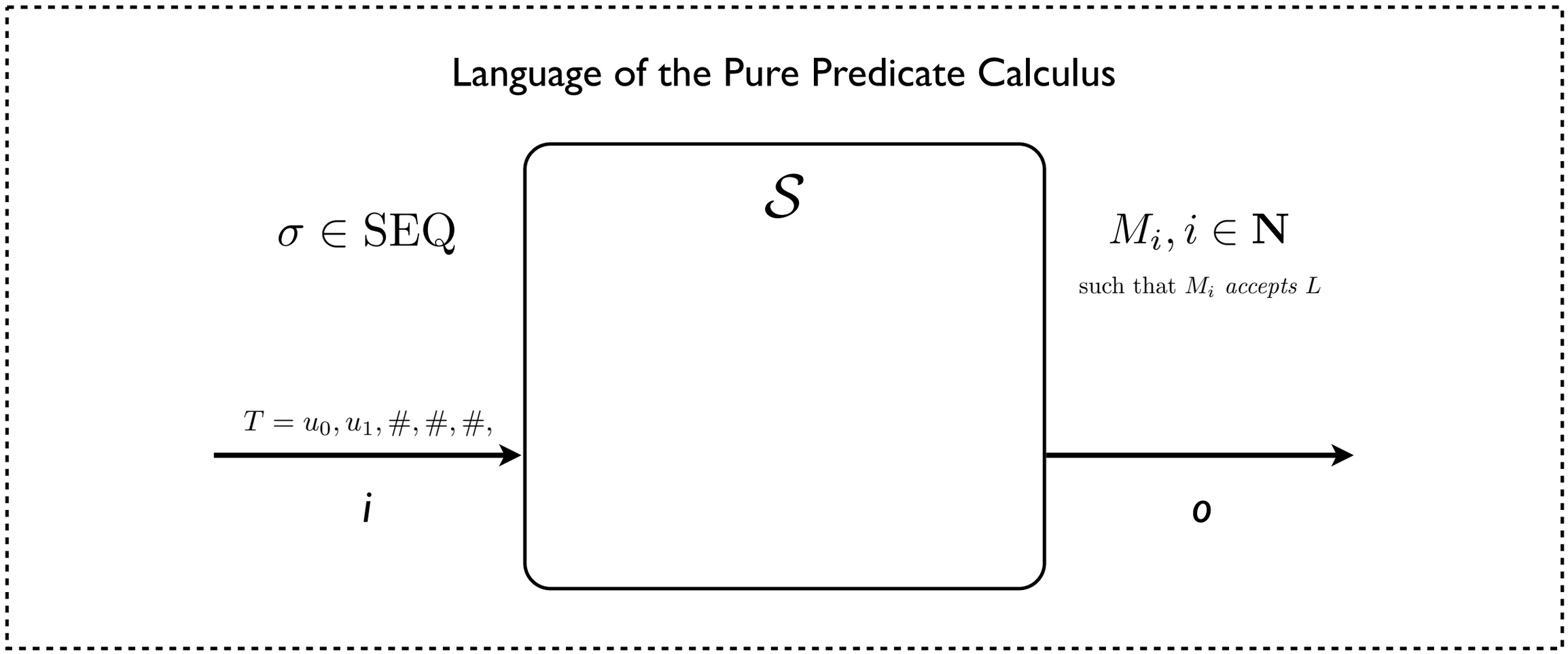
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



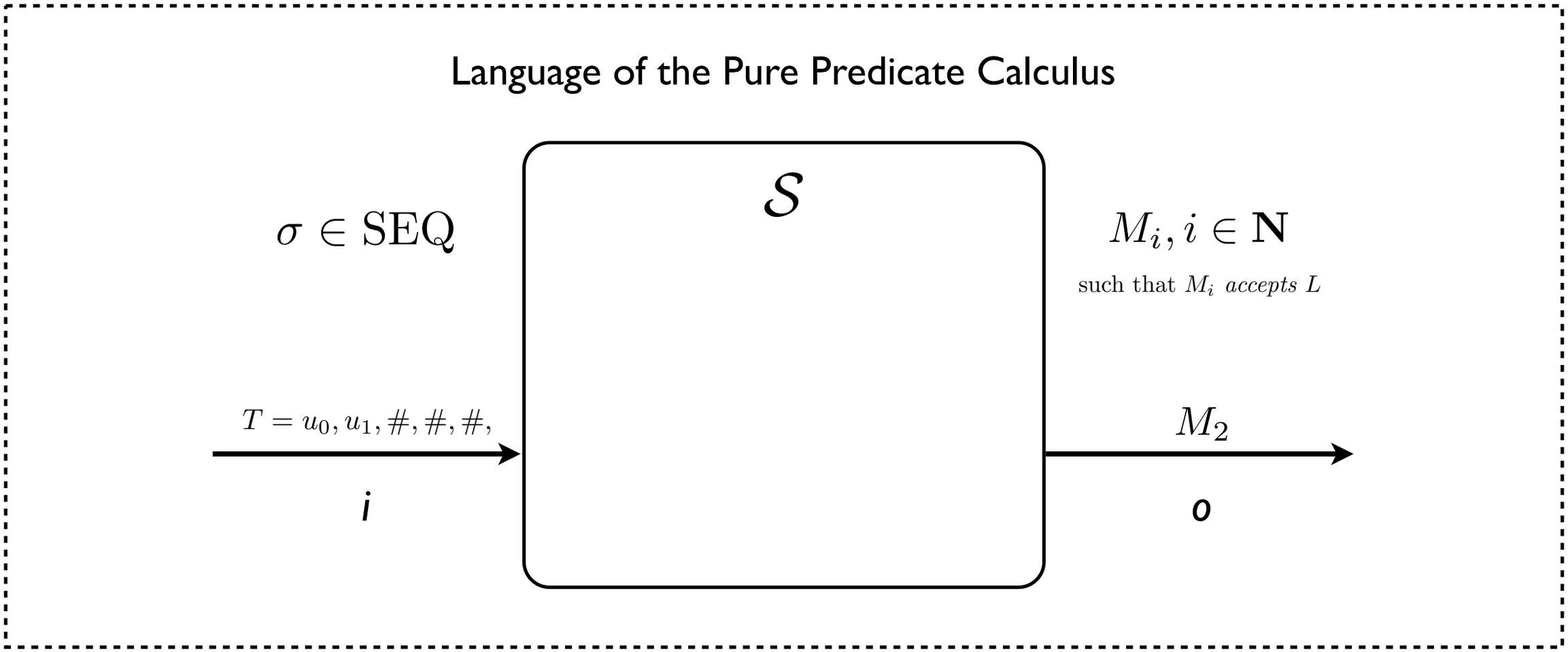
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



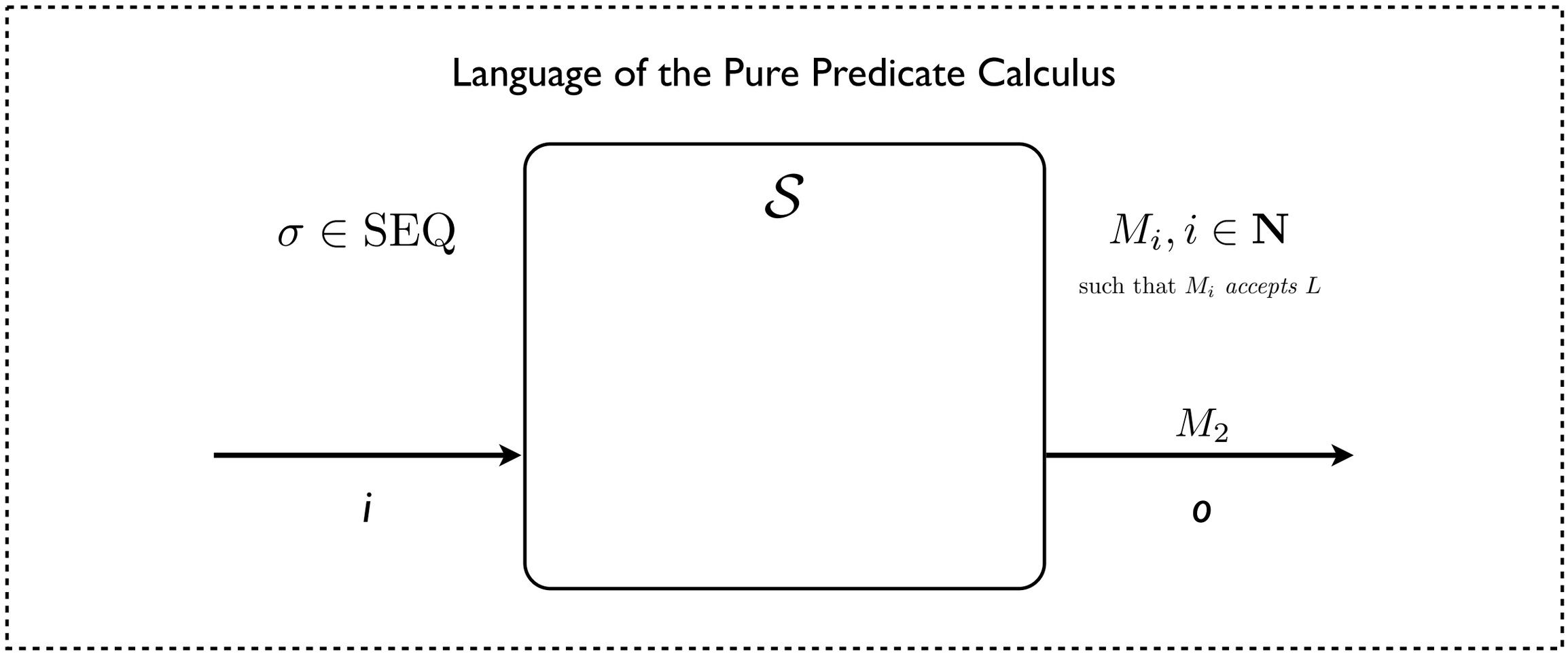
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



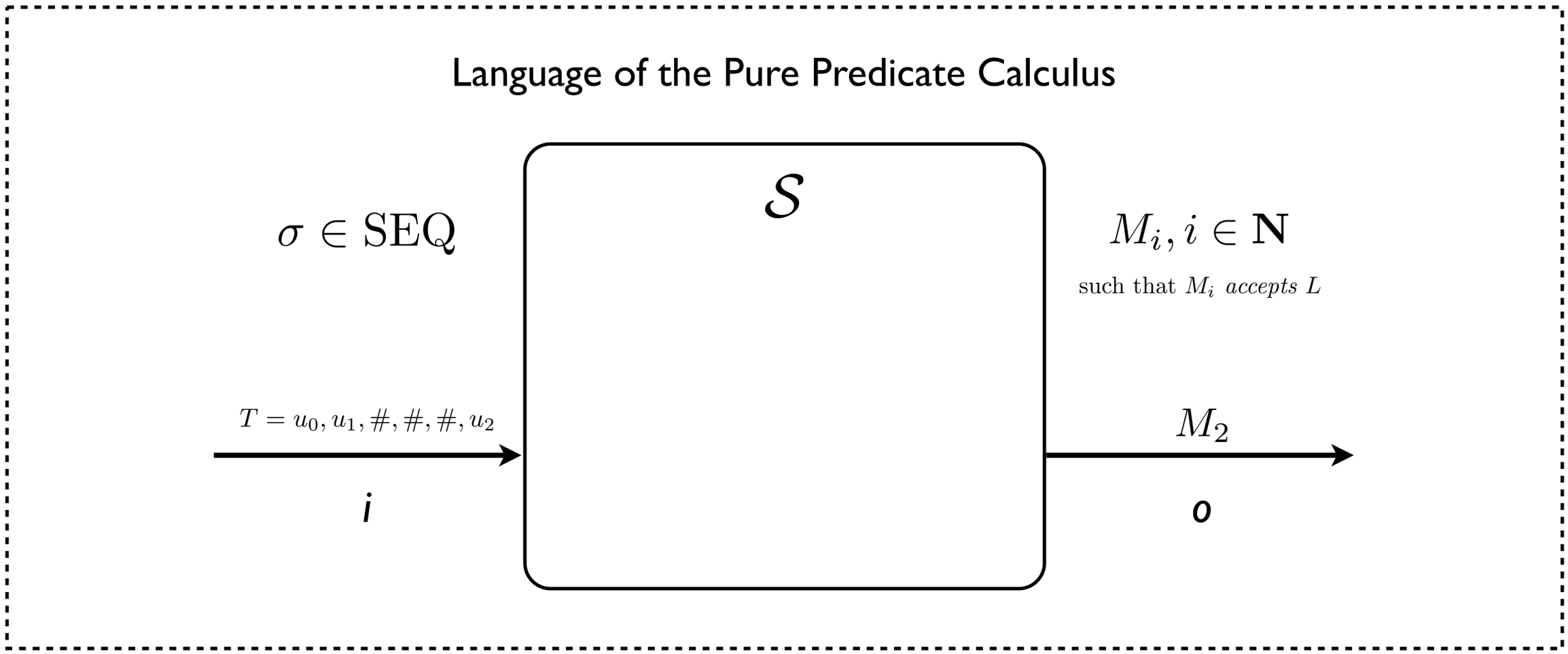
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



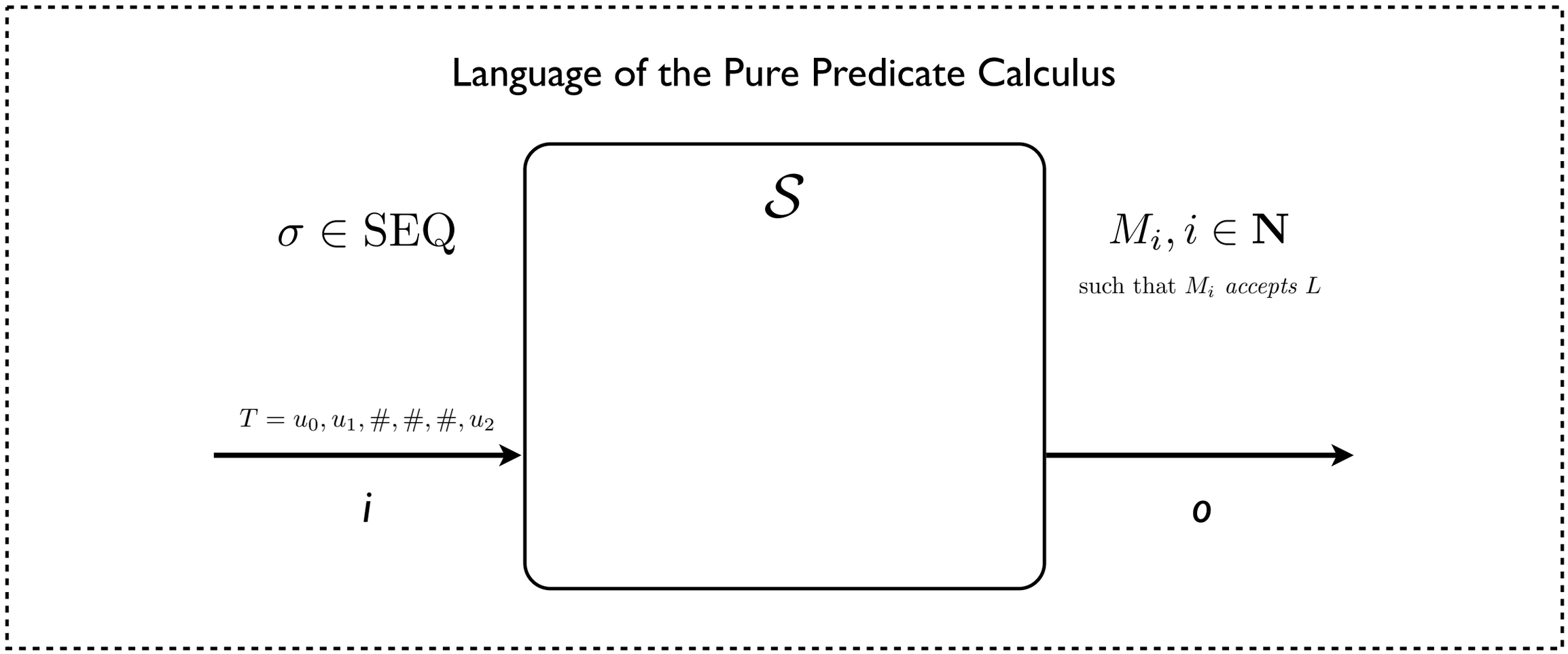
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



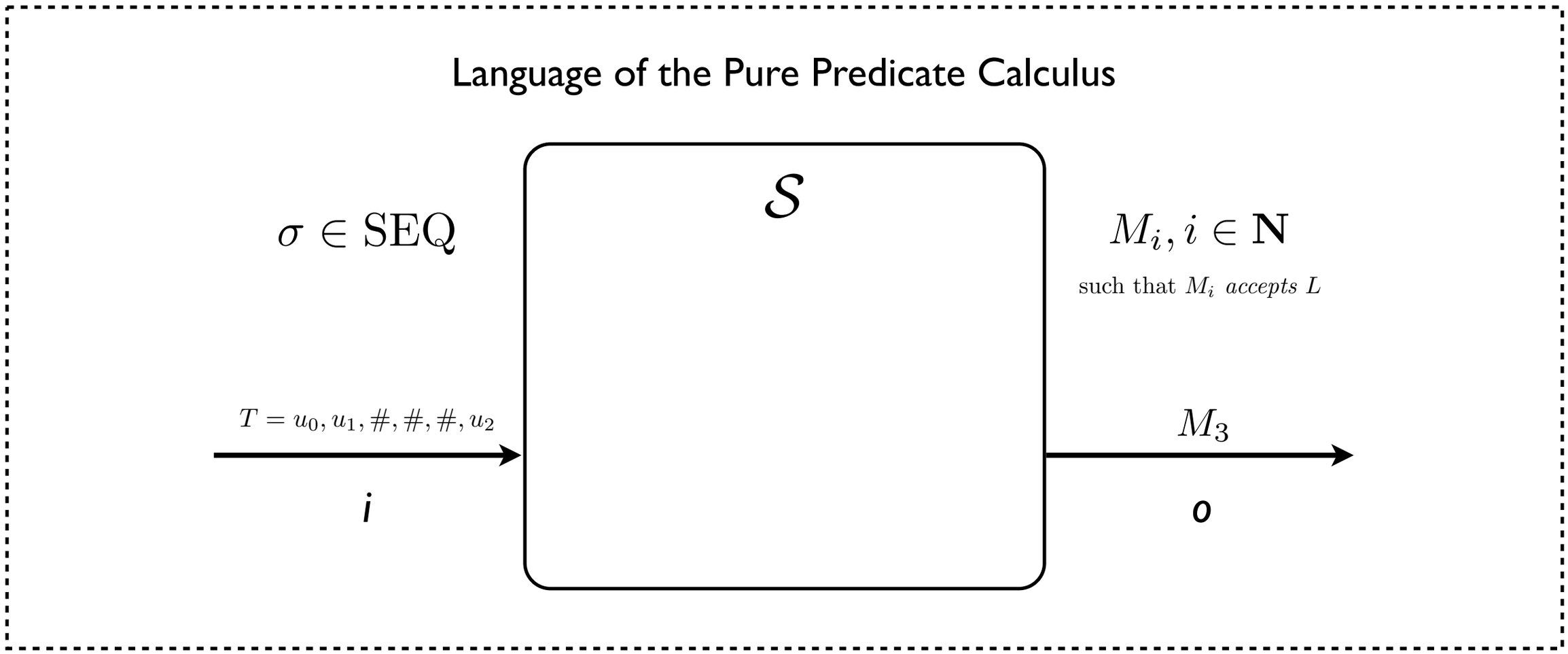
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



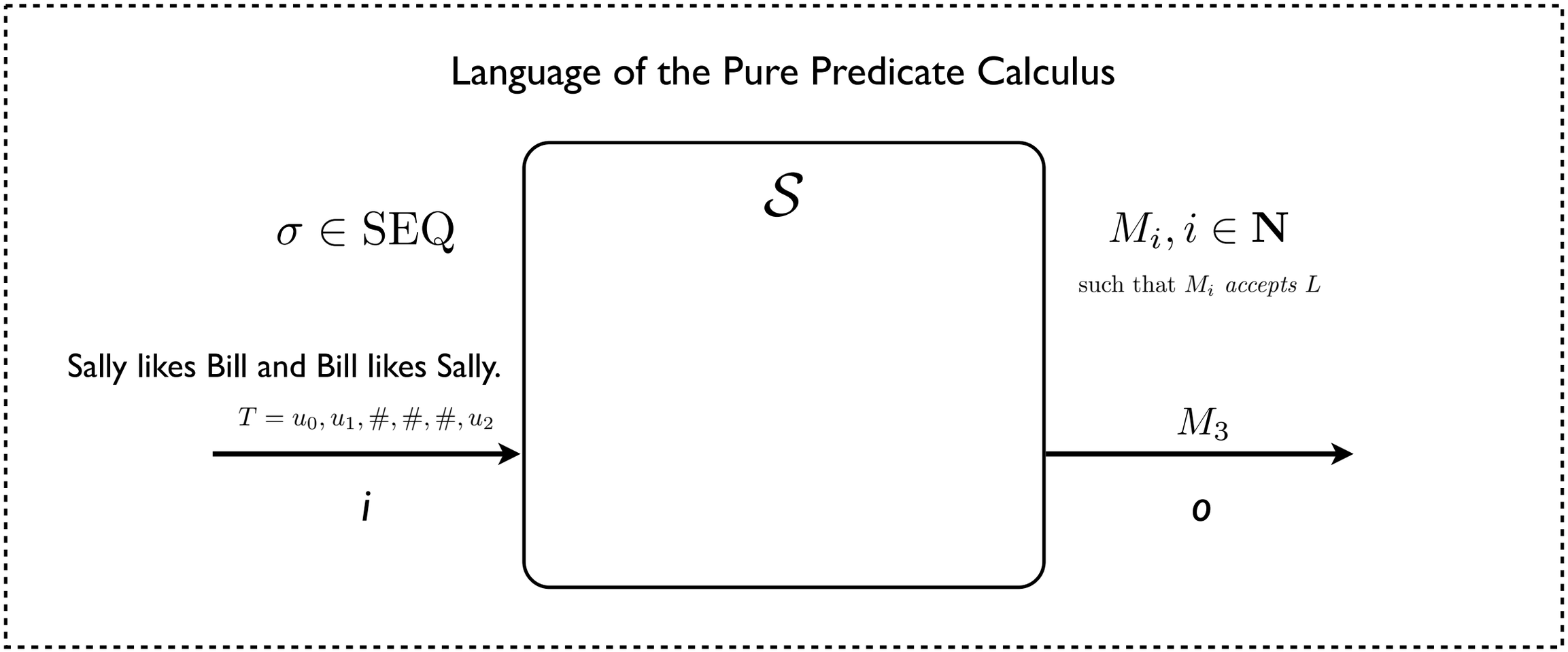
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



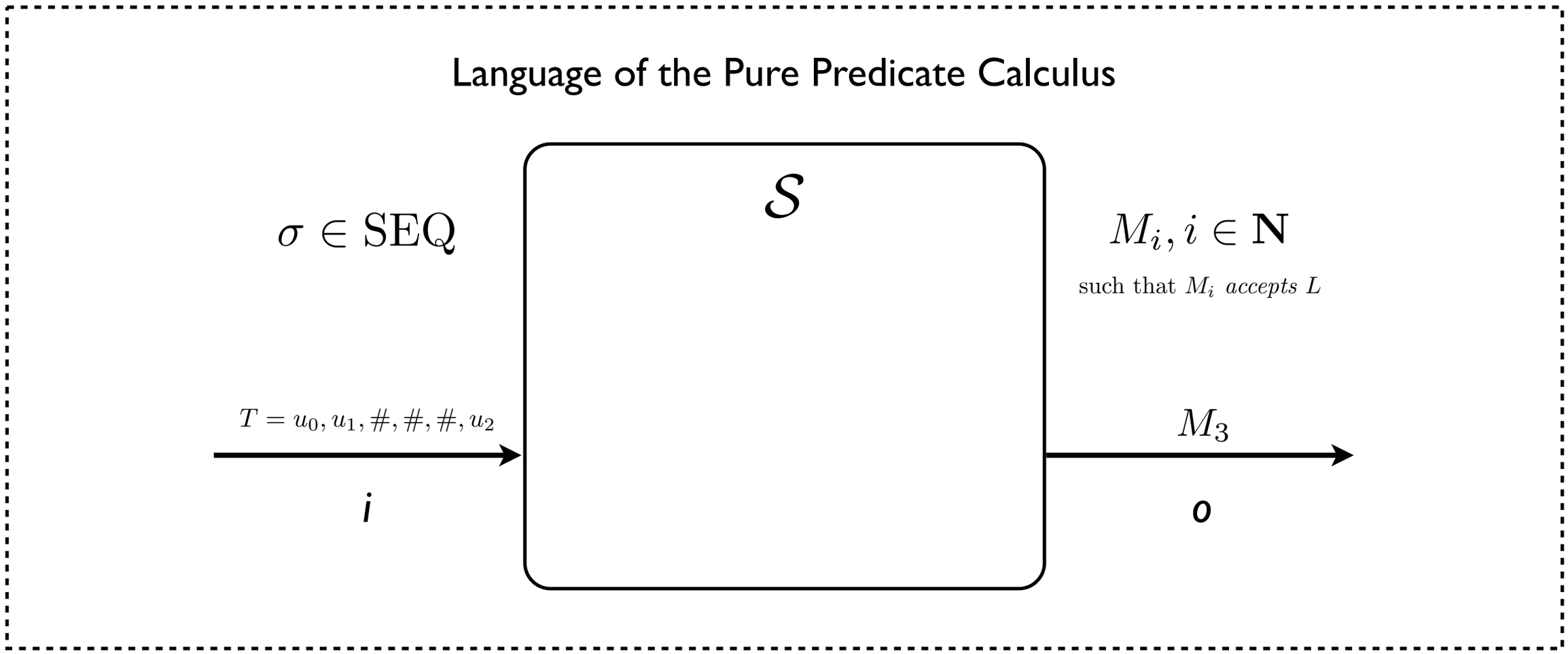
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated



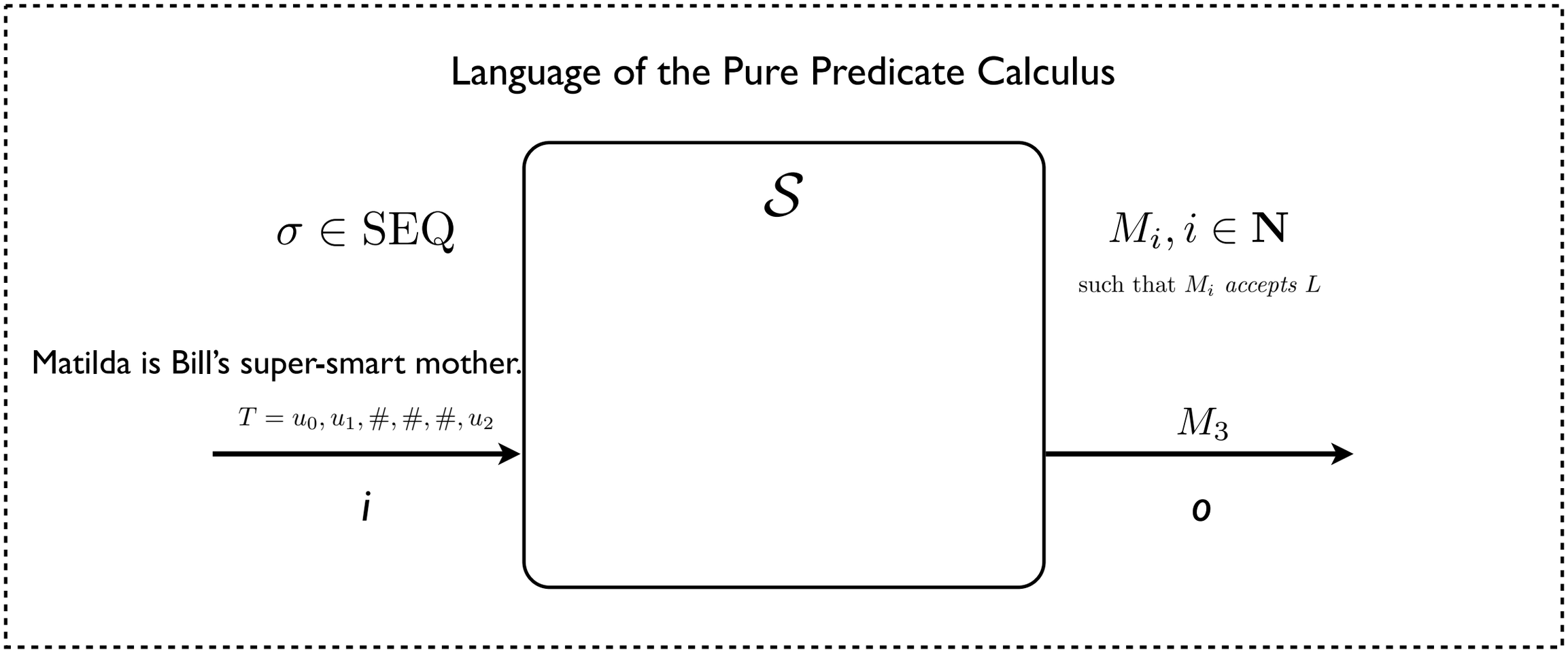
And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

[ ] Need modify because now the language is recursive!

CLT-based Model Instantiated

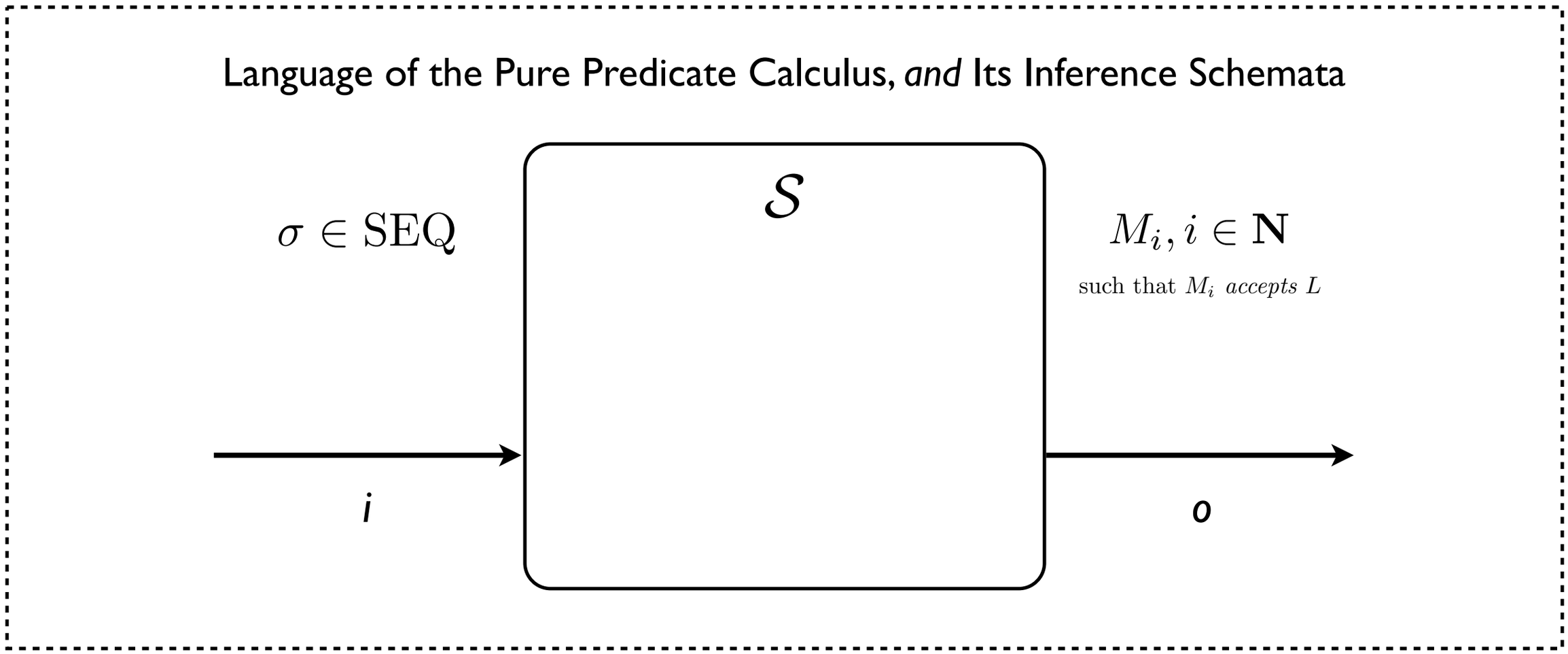


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

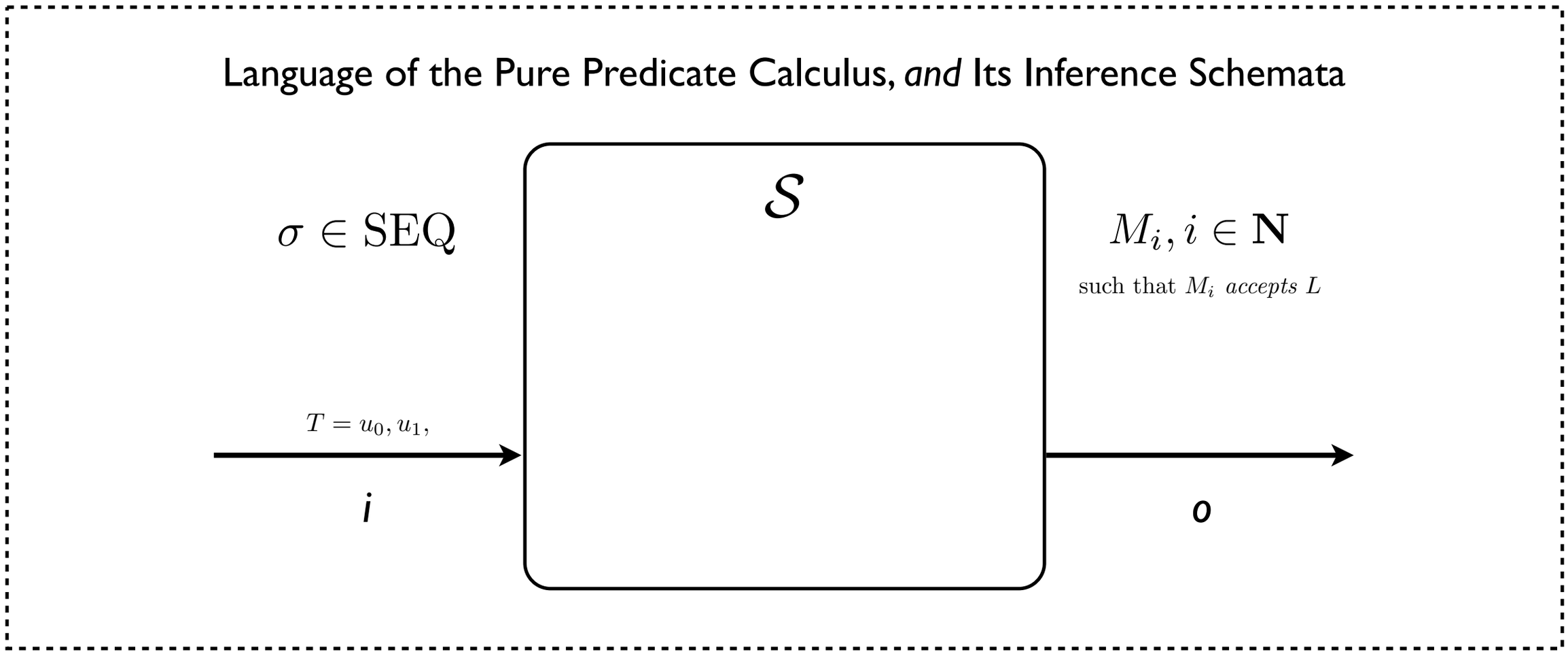


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

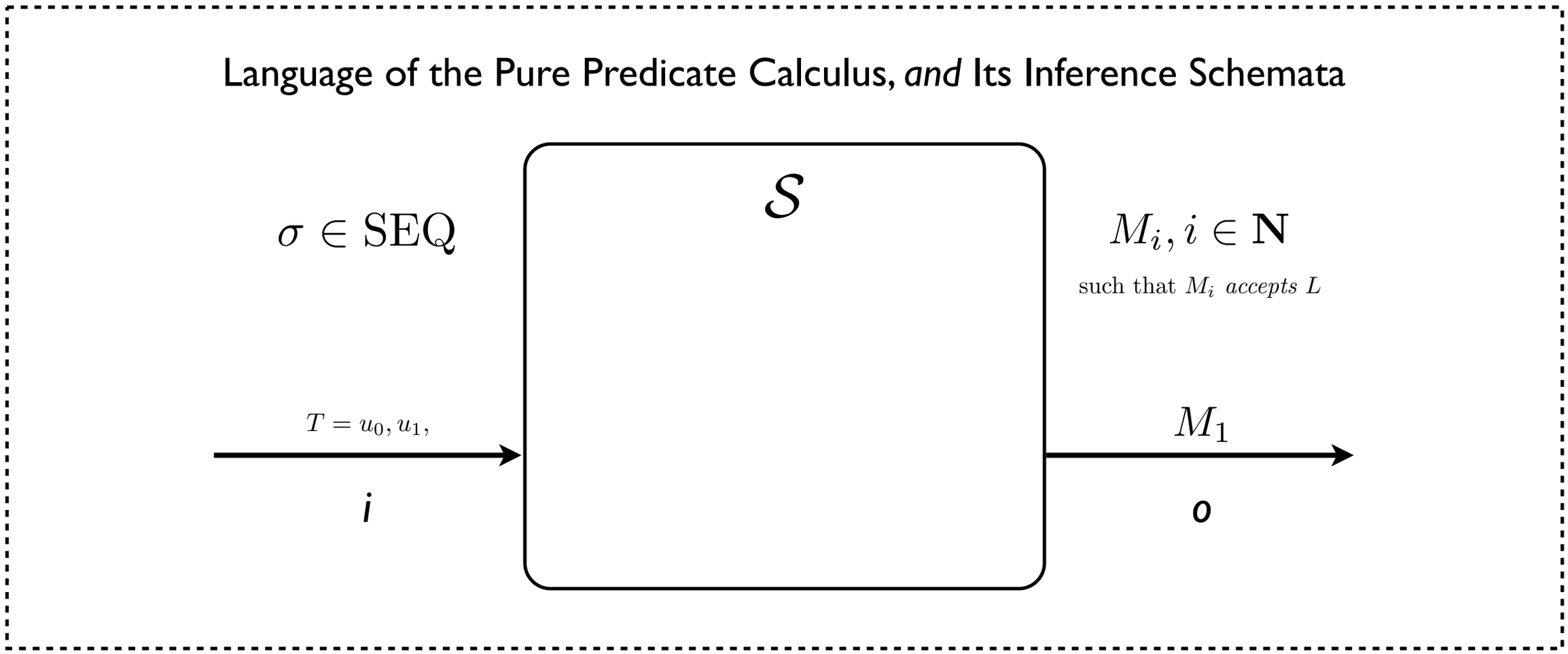


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

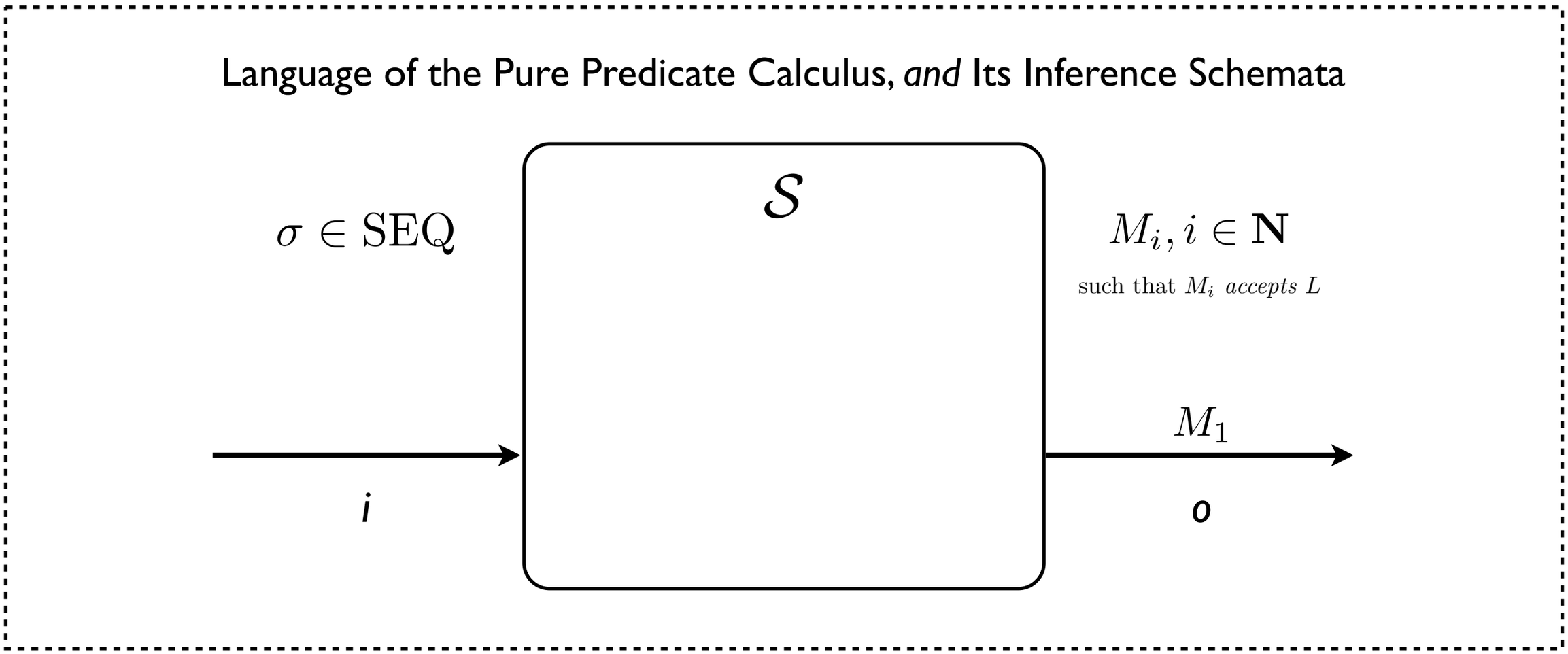


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

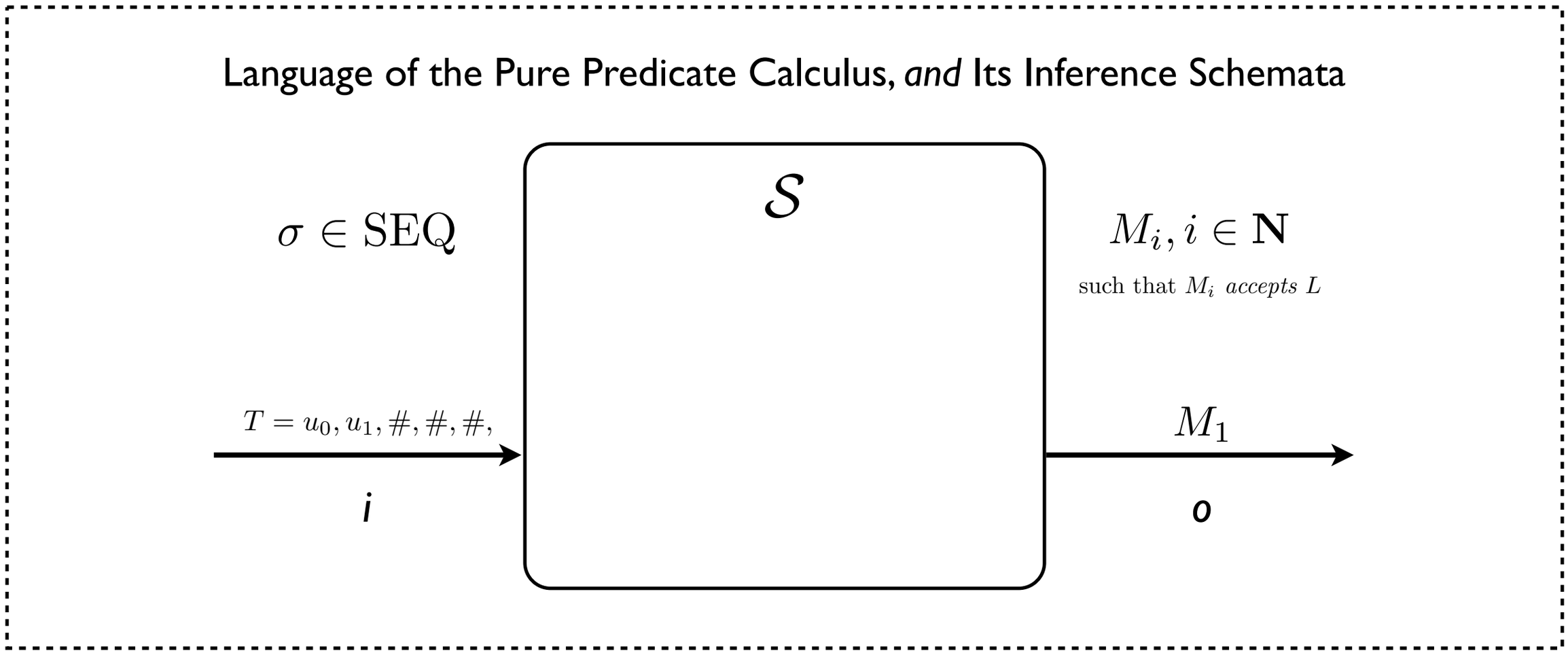


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

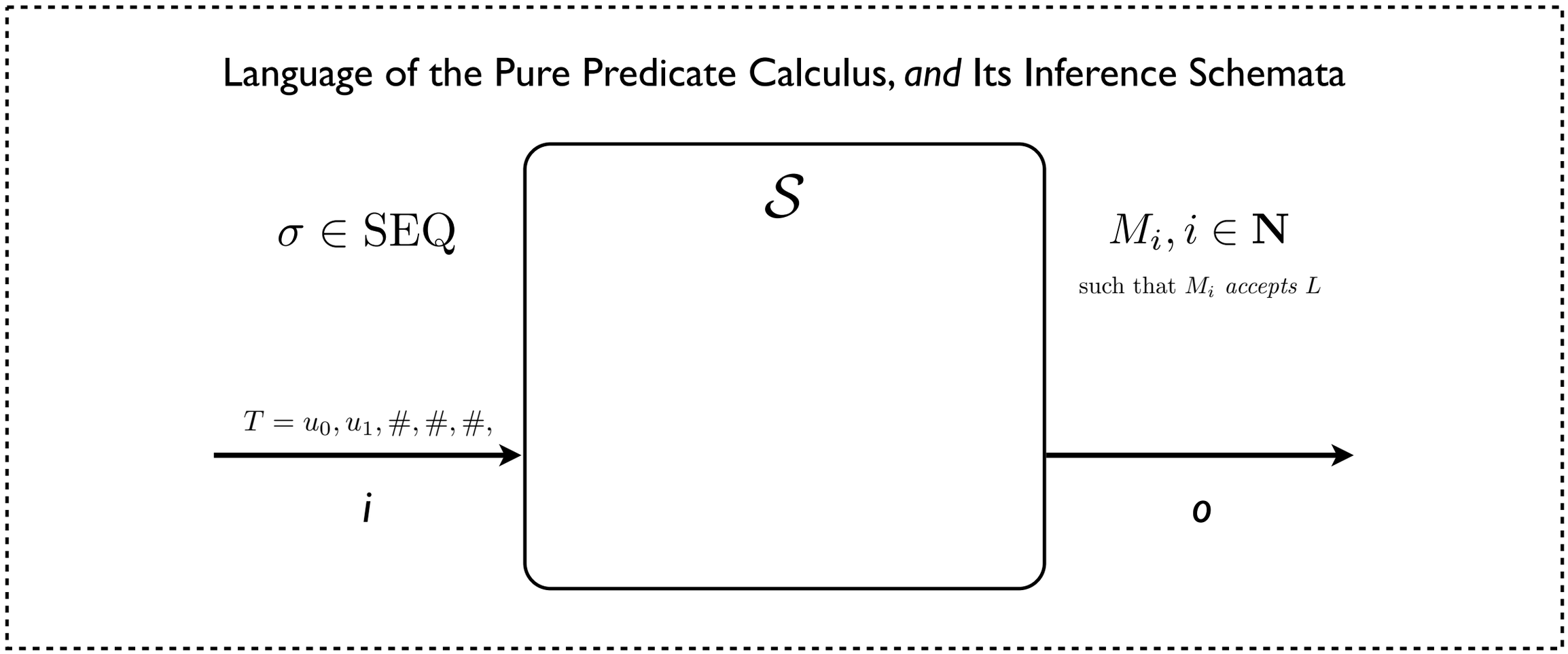


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

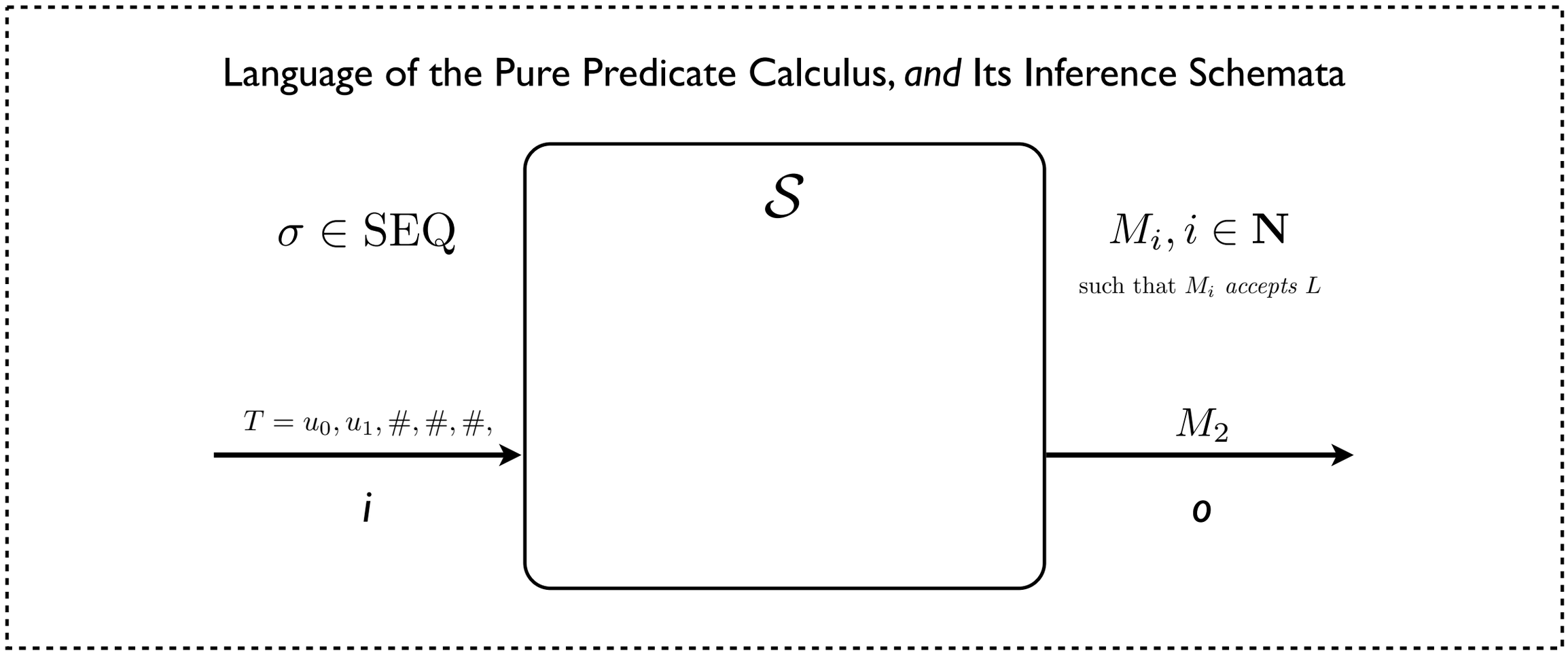


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

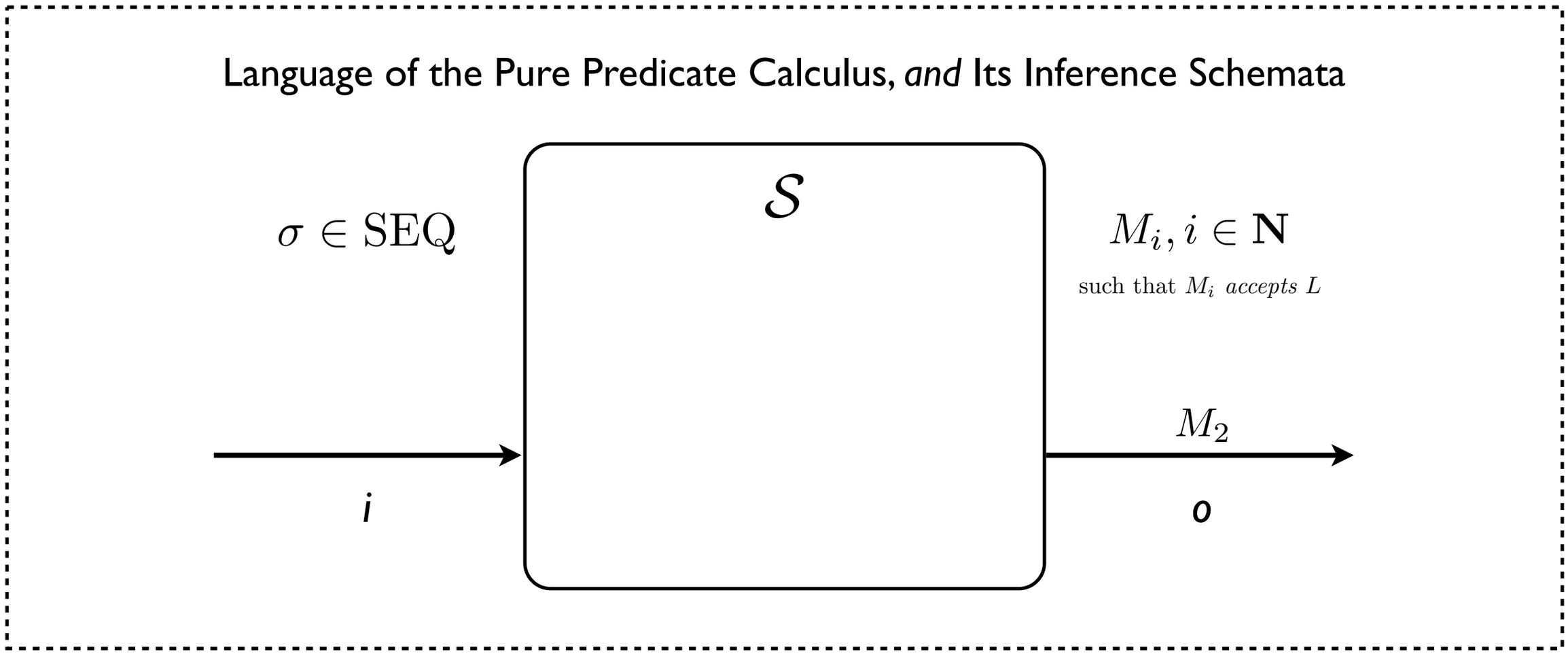


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

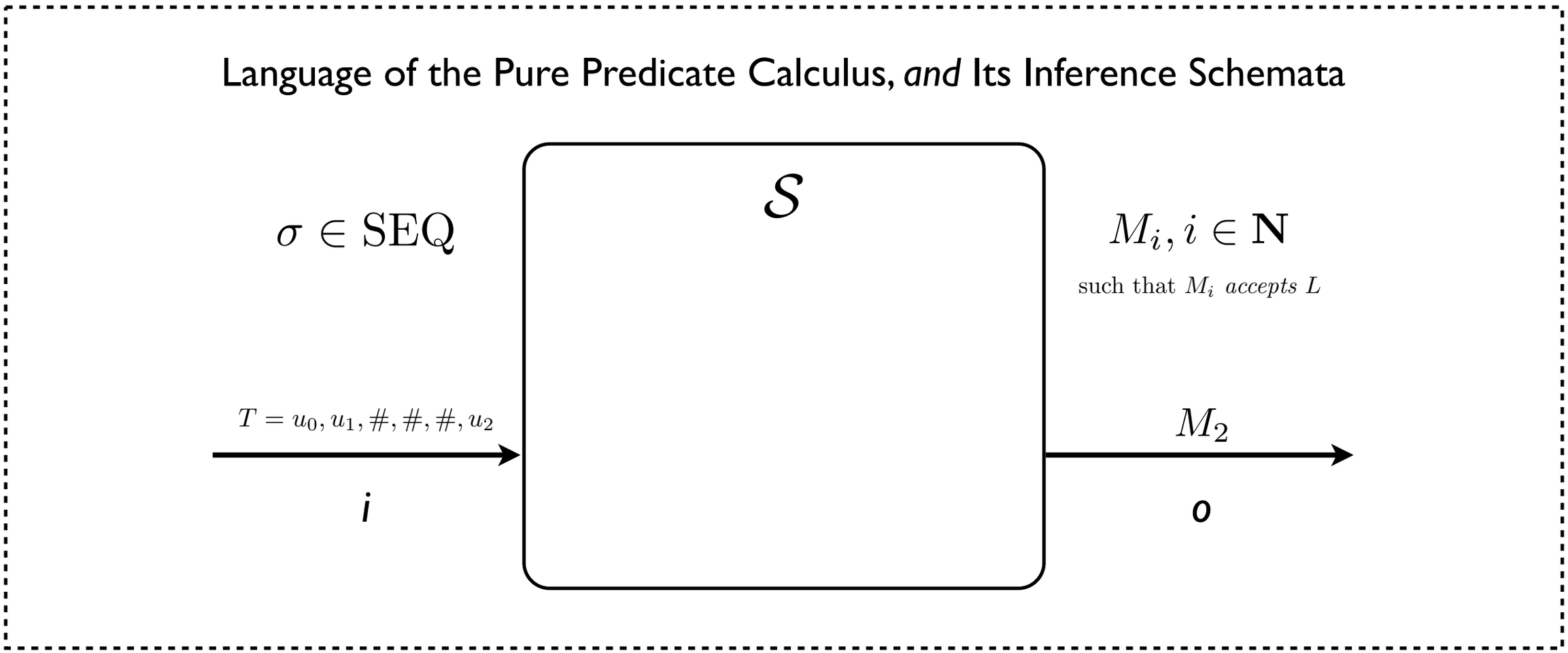


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

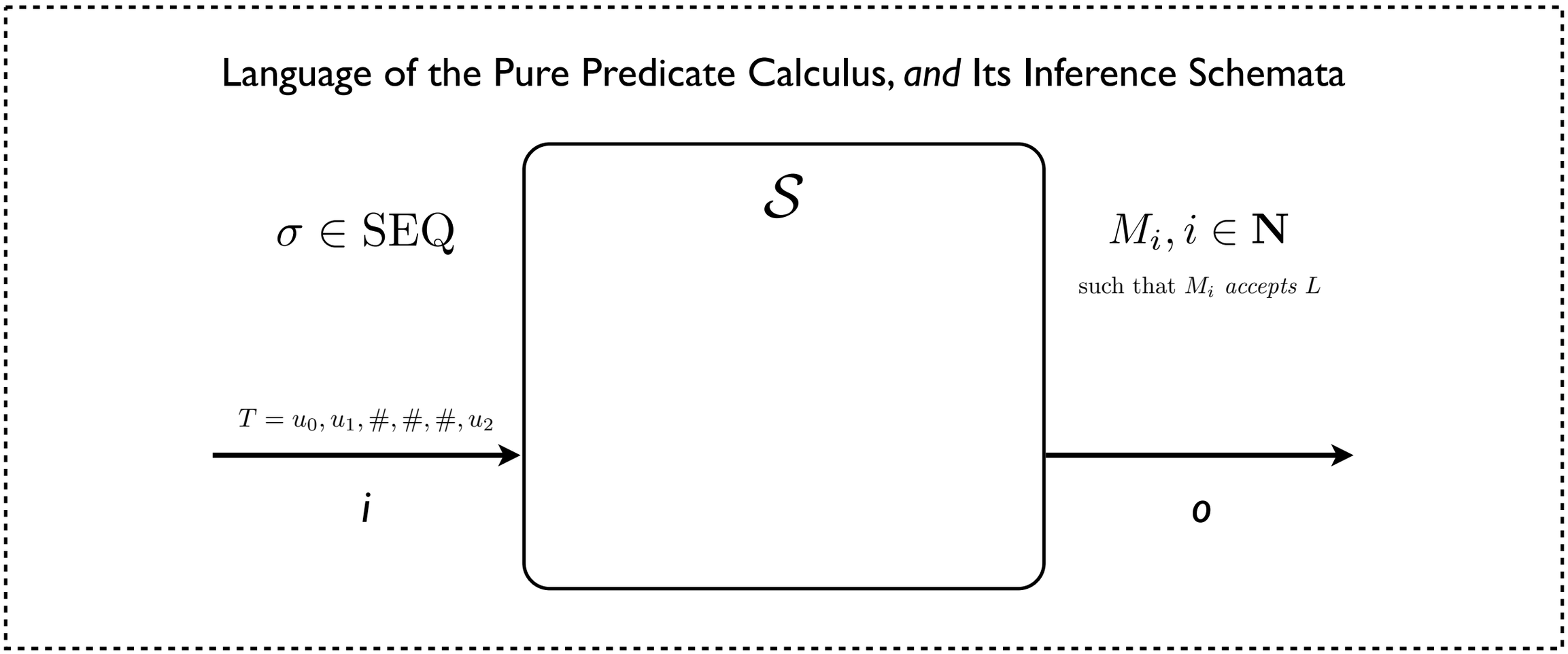


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

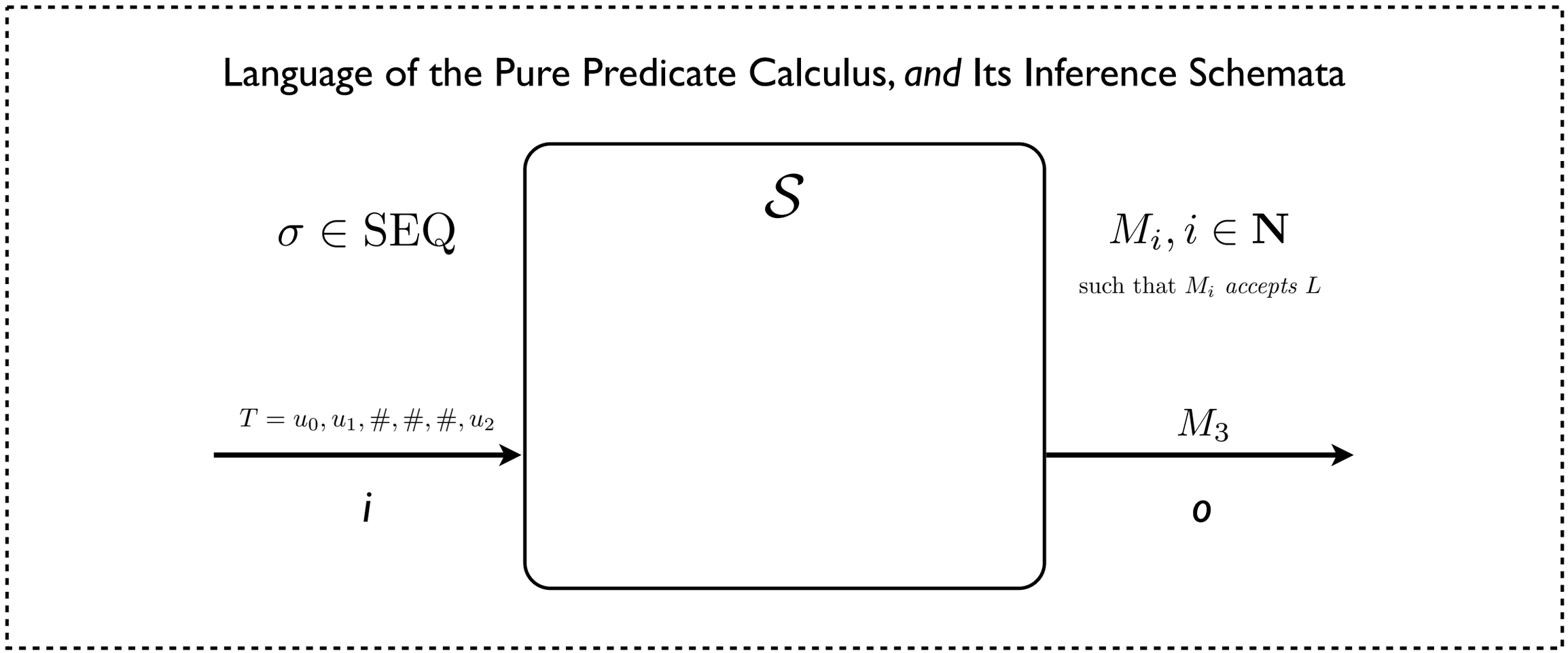


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

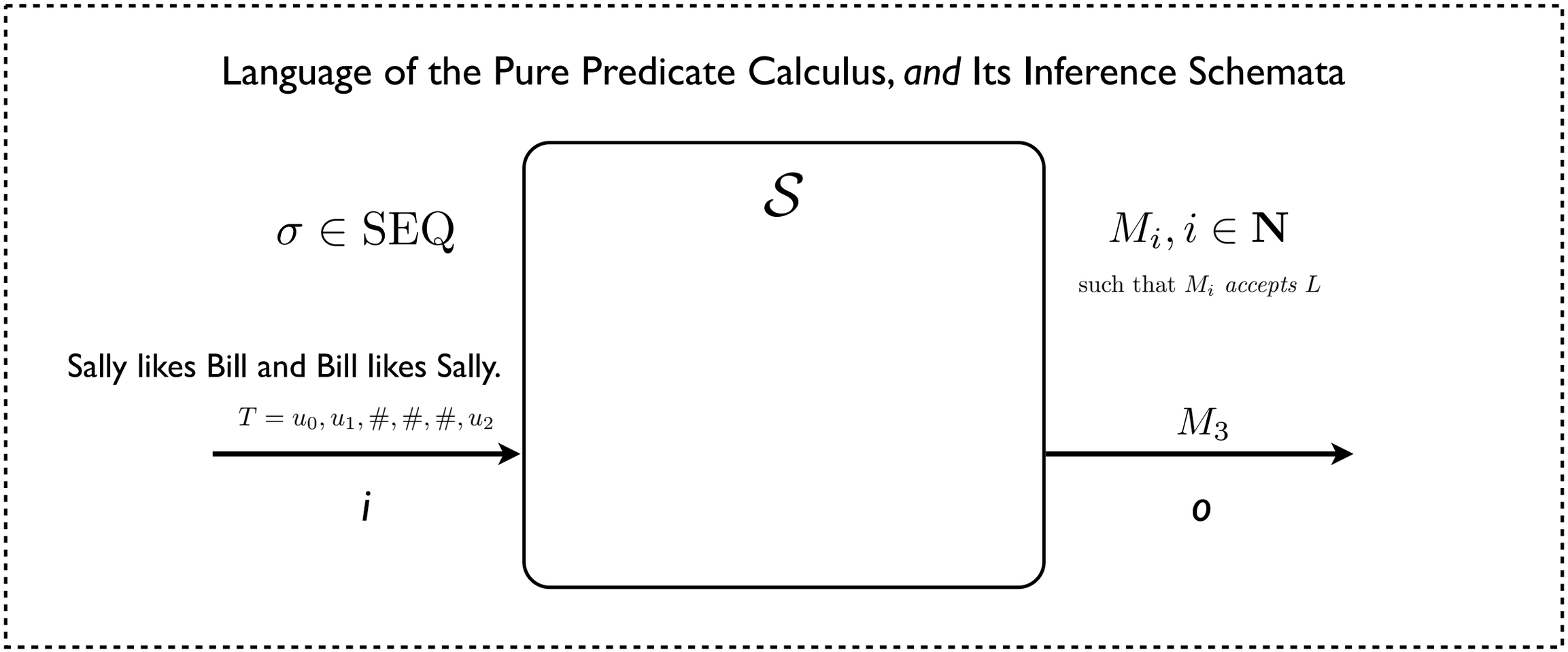


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

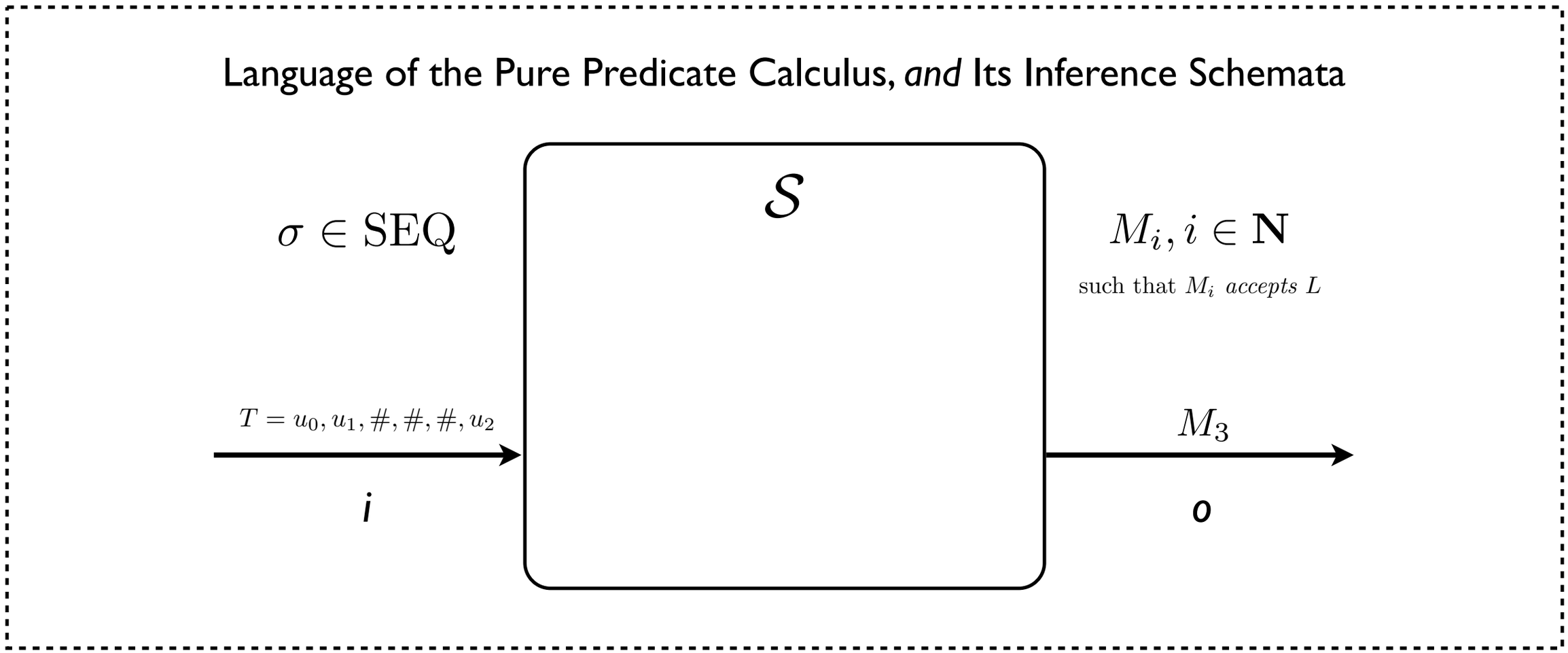


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

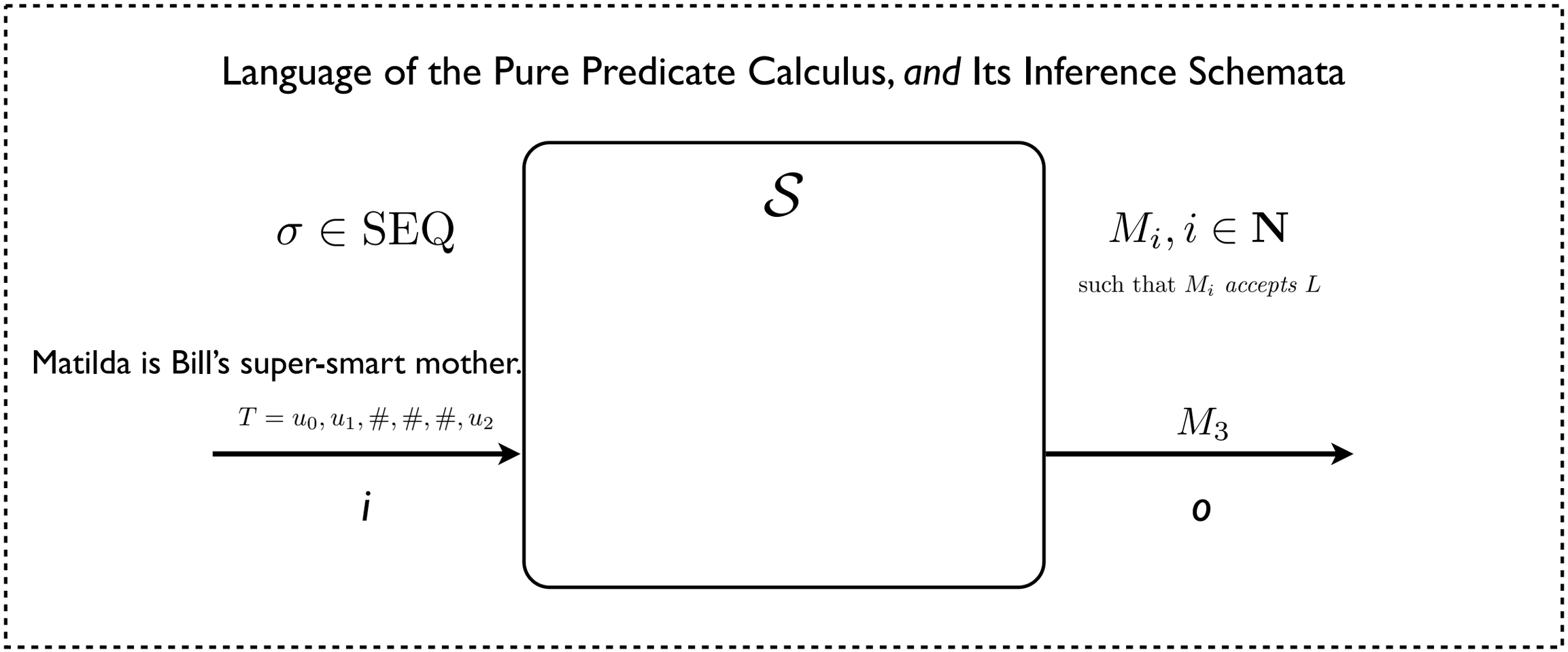


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi

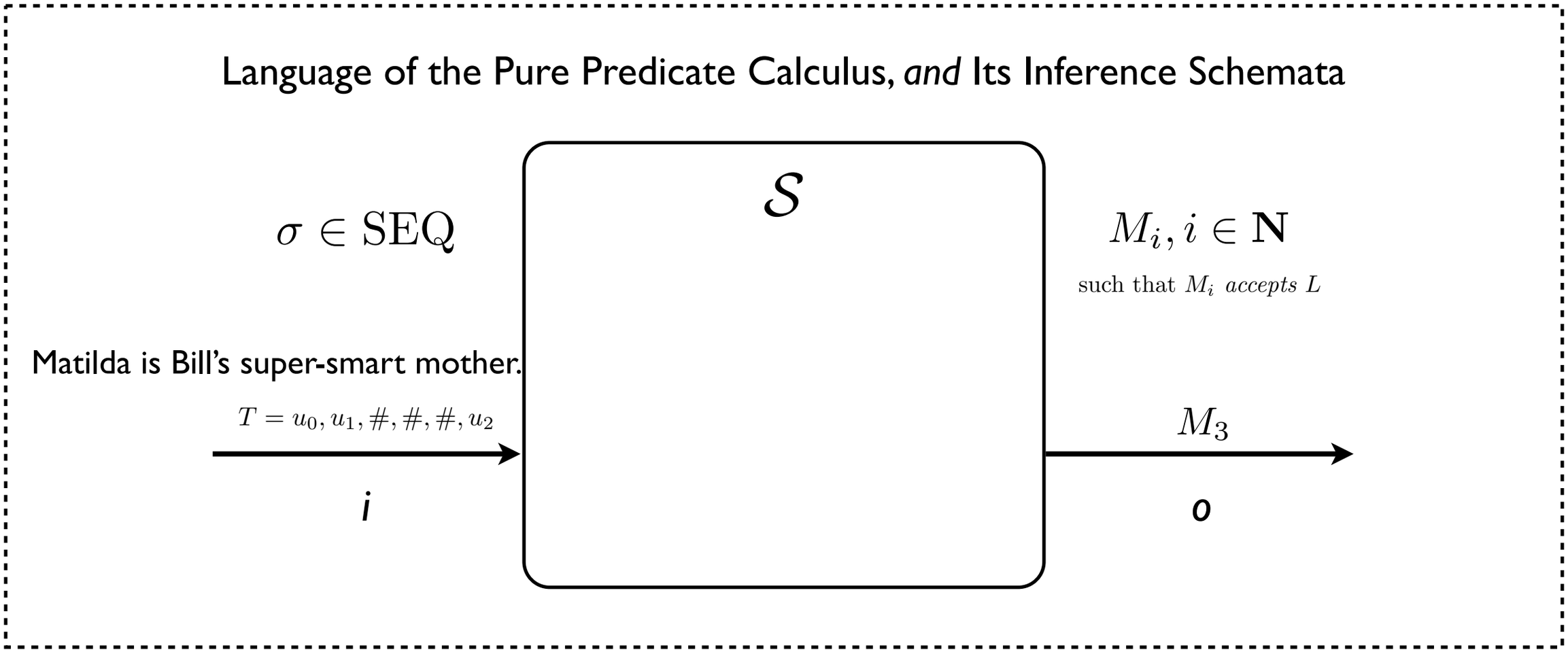


And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$\mathcal{S}$ identifies L iff $\mathcal{S}$ identifies every text for L

CLT-based Model Instantiated to Target Deductive Calculi



And when is identification of *L* achieved?

$$\forall^{\infty} n \in \mathbf{N}\ \mathcal{S}(T[n]) = M_k$$

$$\mathcal{S} \text{ identifies } L \text{ iff } \mathcal{S} \text{ identifies every text for } L$$

Sally likes Bill and Bill likes Sally.
Bill likes Sally.

$$\frac{Lsb \wedge Lbs}{Lbs}$$

RAIR Lab needs to work out the details and engineer an AI that can learn the language of the pure predicate calculus!

Future Work

- What about learning the language(s) of mathematics?
- What about learning to be self-conscious?