Mapping natural language sentences to expressions for the Meth8 model checker

Process summary:

Sentences of natural language are assigned into parts of speech then mapped as theorems and connectives to be checked as models by Meth8. Noun phrases are theorems, and verb phrases are connectives. Adjectives and adverbs are the NOT operator and also the modal modifiers as of necessity and of possibility or necessarily and possibly. A sentence as composed of nouns, adjectives, verbs, and adverbs is mapped into logical symbols to form an expression. It is submitted as a model to be checked by Meth8. The result is to validate the expression as proved or not proved and by which models.

Logical symbols used:

Nouns (theorems A, B, C, D)
Verbs (connectives &+-<=>@\ for and, or, nor, not imply, equivalent, imply, xor, nand)
Adjectives (~#% for not, of necessity, of possibility)
Adverbs (~#% for not, necessarily, possibly)

Example 1. "The floor of the factory contains computers and robots."

1.1 Verb segment (connectives &+-<=>@\)

1.1.1 Verbs
1.1.1.1 contains: =
1.1.1.2 and: &

1.2 Noun segment (theorems A, B, C, D)

1.2.1 Nouns
1.2.1.1 floor: A
1.2.1.2 factory: B
1.2.1.3 computers: C
1.2.1.4 robots: D

1.2.2 Adjectives and adverbs (~#&)
1.2.2.1 (necessarily) of the factory: #B

1.2.3. Possessive verb (connectives &+-<=>@\)
1.2.3.1 the floor (and) necessarily of the factory: A & #B
1.2.3.2 computers and robots: C & D

Mapped result: (A & #B) = (D & C) ;
Output follows as partial proof for (A&#B)=(D&C) [only Model 2.2]:

Model 1	Model 2.1	Model 2.2	Model 2.3.1	Model 2.3.2
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII
TTTT TTTT TTCC TTCC	EEEE EPEP EEII EPIU	EEEE EEEE EEEE EEEE	EEEE EPEP EEEE EPEP	EEEE EEEE EEII EEII

FCNT is: False, Contingent, Non contingent, True. UIPE is: Unevaluated, Improper, Permissible, Evaluated.

Example 2: "FOR idx = 1 TO n; NEXT idx."

Rrewrite as DO-WHILE-LOOP for clarity as:

"LET idx = 1; DO WHILE idx < 1 + n; LET idx = 1 + idx; LOOP"

2.1 Verb segment (connectives &+-<=>@\)

2.1.1 Verbs
2.1.1.1 LET: =
2.1.1.2 DO: >
2.1.1.2 LOOP: >

2.2 Noun segment (theorems A, B, C, D)

2.2.1 Nouns
2.2.1.1 idx: A
2.2.1.2 1: B
2.2.1.3 n: C

2.2.2 Adjectives and adverbs (~#&)
2.2.2.1 WHILE: ~

2.2.3. Possessive verb (connectives &+-<=>@\)
2.2.3.1 LET idx = 1: (A = B)
2.2.3.2 1 + idx: (B + A)
2.2.3.3 LET idx= 1 + idx: (A = (B + A))
2.2.3.4 1 + n: (B + C)
2.2.3.5 idx < 1 + n: rewrite as Not(idx = 1 + n): ~(A = (B + C))
2.2.3.6 WHILE idx < 1 + n; rewrite as WHILE Not(idx = 1 + n): ~(A = (B + C))
2.2.3.7 LET idx = 1; DO WHILE idx < 1 + n: (A = B) & ~(A = (B + C))
2.2.3.8 IF (idx = 1 AND DO WHILE ~(idx = 1 + N)) THEN (idx = 1 + idx):

Mapped result: ((A = B) & ~(A = (B + C))) > (A = (B + A));
Output follows as a proof of correctness for: ((A=B)&~(A=(B+C)))>(A=(B+A))

Model 1	Model 2.1	Model 2.2	Model 2.3.1	Model 2.3.2
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE
TTTT TTTT TTTT TTTT	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE	EEEE EEEE EEEE EEEE

FCNT is: False, Contingent, Non contingent, True. UIPE is: Unevaluated, Improper, Permissible, Evaluated.

Conclusion:

The first example shows how one sentence of a requirements document can be verified and validated by proof in only one of the five models of Meth8. To achieve proof in the other models, the requirements sentence could be unfolded into a larger requirements document such as a longer sentence or a series of sentences with a collecting sentence.

The second example shows how a computer software program, in this case a FOR-NEXT loop unfolded into a DO-WHILE loop, can be verified and validated by proof in all five models of Meth8.

Both examples demonstrate the efficacy of Meth8 as a strategic asset in computing.