Language to logic mapper to logic model checker

Colin James III
info@cec-services.com

ESMC
Colorado Springs

What is automated reasoning.

- This goal is to obtain logical proof of sentences.
- The motivation is to validate the requirement documents of a product before its manufacture.
- The economic benefit avoids mistakes before manufacture, because defects are costly to fix.
- The approach is two-fold:
  - 1. Input from the language to logic mapper (LLM)
  - 2. Output from the logic model checker (LMC)

Language to logic mapper (LLM)

- What is natural language?
  - What fits in sentences is parts of speech (POS).
- POS are abstract groups of:
  - Noun, Verb, Modifier (NVM).
    - The modifier is an adjective and adverb.
- POS approach ignores grammatical distinctions
  - A subject, object, or direct object is still a “Noun”.

POS by stemmer lookup table (LUT)

- Public domain list of POS: 180K English words
  - POS are further grouped into three logical groups.
    - 3 Nouns as singular, plural, pronoun
    - 5 Verbs as in-transitive, participle, gerund, conjunction
    - 4 Modifiers as adjective, adverb, preposition, article
    - Ignored are nominatives and interjections.
- Uses *sequential access* to the word list:
  - Searches on average for ½ of the list;
  - Avoids overhead of a sorted list for binary search;
  - Prompts for a word not found, to add it to the list.

Some POS are ambiguous.

- The stemmer may be several POS.
  - “free” as such is noun, verb, adjective, and adverb.
- The *alias lemma* is defined as the unique POS sequence pattern from adjacent stemmers.
- The *pseudo lexeme* := *alias lemma* + stemmer.
  - Example 1: **Tango** is leaders and followers.
    - Pattern is **Tango** – verb - nouns, hence chunked as **N**vn.
  - Example 2: Leaders and followers **tango**.
    - Pattern is nouns - **tango**, hence chunked as n**V**.

Generic disambiguation of POS

- LET *m* = modifier; **N** = noun; **V** = verb
- Pattern format: *mm***N***mmmmm***V***mmmmm***N**.
- "*mm***N**" is adverb adjective Noun
  or adjective adjective Noun
- "**N***mm*" is Noun adjective adjective
  or Noun adverb adjective
- "*mm***V**" is adverb adverb Verb
- "**V***mm*" is Verb adverb adverb

The ambiguous pattern options

- “holidays very warm ended”: N*mm*_V
  - “very warm” as *adverb adjective* modifies the Noun
- “holidays very early begin”: N_*mm*V
  - “very early” as *adverb adverb* modifies the Verb
- “ended very early holidays”: V*mm*_N
  - “very early” as *adverb adverb* modifies the Verb
- “begin very warm holidays”: V_*mm*N
  - “very warm” as *adverb adjective* modifies the Noun

The ambiguous option choice

- *Adverb adjective* is before or after Nouns
  - *mm*N: *adverb adjective* Noun "*very warm* holidays"
  - N*mm*: Noun *adverb adjective* "holidays *very warm*"

- *Adverb adverb* is before or after Verbs
  - *mmV*: *adverb adverb* Verb "*very early* ended"
  - V*mm*: Verb *adverb adverb* "ended *very early*"

What is logic model checker (LMC).

- Jan Łukasiewicz invented logic Ł4 but did *not*:
  - Find all combinations of the logical models; or
  - Find academic acceptance of his 4-valued system.
- Garry Goodwin (UK) and I (USA) fixed Ł4:
  - Variant Ł4 (V Ł4) uses 5 logical models; and
  - A 4-valued logic system of {F,N,C,T} and {U,I,P,E}.
- Meth8 logic model checker implements this.
  - We claim this *recent advance* in mathematical logic.

Example 1 of Meth8

Gödel-Löb Axiom

Variant Ł4 (VŁ4)

- Uses two sets of 4-valued logic for 5 models
- Validated when all models **T**rue and **E**valuated

C**T**C**T**	U**E**U**E**	**EEEE**	P**E**P**E**	I**E**I**E**
Model 1	Model 2.1	**Model 2.2**	Model 2.3.1	Model 2.3.2

- **□(□ p →p) → □p** The Gödel-Löb axiom (GL)

  The necessity of *choice*, as always implying *a choice*, implies always *a choice*.

  - This is suspicious with only one valid model of five.
  - If GL fails, then so does Zermelo - Fraenkel set theory and axiom of choice (ZFC) as the basis of mathematics.

What GL wished it was in words.

- **□(□~p→p) ↔ □p** [1]

  The necessity of *no choice*, as always implying *a choice*, is equivalent to always *a choice*.

- **□(□p → ~p) ↔ □~p** [2]

  The necessity of *choice*, as always implying *no choice*, is equivalent to always *no choice*.

- **□(□p → p) ↔ □(p ᵛ ~p)** [3]

  The necessity of *choice*, as always implying *a choice*, is equivalent to always *a choice* or *no choice*.

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

Example 2 of Meth8

Appendix

Karl Popper's proof of God

How to map POS to logical symbols.

- Nouns are literals for
  - Propositions lower case {p,q,r,s} in 16 table-values
  - Theorems upper case {A,B,C,D} in 256 table-values
- Verbs are connectives assigned {&+-<=>@\} for
  - { and, or, nor, not imply, equivalent, imply, xor, nand}
- Modifiers are operators assigned as { ~#% } for negation and modal necessity / possibility:
  - Adjectives as { not, necessary, possible }
  - Adverbs as { never, necessarily, possibly }

How to prove a sentence.

- A sentence is a proof table of logical values, eg:
  - "A floor of the factory has robots and computers."
  - "[*Possibly]* a floor [*and necessarily*] of the factory [*is*] robots and computers."
  - **(%p & #q) = (r & s)**.

Model 1	Model 2.1	Model 2.2	Model 2.3.1	Model 2.3.2
TTTC TTCC	EEEU EEEU	**EEEE EEEE**	EEEP EEEP	EEEI EEEI
TTTC FFFN	EEEU UUUE	**EEEE** UUUU	EEEP UUUI	EEEI UUUP

- Valid: **TTTT** in Model 1 and **EEEE** in Models 2.
  - Which models above are validated?

How to fix a sentence to prove it.

- "[*Possibly]* a floor [*and necessarily*] of the factory [*is*] robots and computers." **(%p & #q) = (r & s)**. [1]
- Rewrite logical expression: "The facts of necessity of the factory and possibility of a floor implying both possibly a floor and necessarily the factory, which combined with the facts of robots and computers implying both robots and computers, implies that *possibility of a floor and necessity of the factory is equivalent to robots and computers*."
- ((((#q=#q)&(%p=%p))>(%p&#q))&(((r=r)&(s=s))>(r&s))) > (**(%p & #q) = (r & s)**), for valid all models. [2]

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

Example 3 of Meth8

How to fix a sentence:

Proof for God is time

Conjecture for God is time

If God knows that past, present, and future are true
[and that past implies present, implies future],

then:

God as past implies God as present, implies past as present;

or
God as past implies God as future, implies past as future;

or
God as present implies God as future, implies present as future

{or past as present implies past as future, implies present as future}.

Proof for God is time

LET: p = God; q = past; r = present; and s = future
[also, t = time = q & r & s].

In Meth8 logic model checker script:

```
(p  &  ((((q=q)&(s=s))&(r=r))
   [&  (((q=q)>(s=s))>(r=r)))] )
>
(  ((((p=q)>(p=r))>(q=r))
 +  (((p=q)>(p=s))>(q=s)))
 +  (((p=r)>(p=s))>(r=s)))
{+((q=r)&(q=s))&(r=s))} );  Valid all models
```


Tips to fix a sentence to prove it.

- The technique of completeness & satisfiability
  - Rewrite the logical expression to recheck mapping
- The technique of expansion
  - Make sentences more descriptive
  - Divide sentences into simpler descriptive parts
- The technique of contraction
  - Abstract sentences into general, generic content
  - Build sentences with higher informational content
- Technical writing aid from proposals and grants

Sentences (S) to paragraphs (P)

- S1: nouns A, B, verb “is” =; S1 means (A=B).
- S2: nouns A, B, verb “and” &; S2 means(A&B).
  - P1 is S1 then S2, P1: S1>S2 means (A=B)>(A&B). The implication connective “>” is inserted between S1 and S2 based on the reason that S2 follows S1 in logical sequence.
  - In words, “S1 implies S2”, that is, “If S1, then S2.”
  - (“S1 as True implies S2 as False” is not allowed.)
  - **Sentence order is important.**
- Subsequent Pn imply requirements documents.

Summary

- Automated reasoning is achieved by mapping language to logic and by checking logic models.
  - Sentences use a word LUT for POS of stemmers.
  - Disambiguate pseudo lexemes from alias lemmas.
  - Map POS groups to symbols by logical expression.
- Sentences are validated by five logical models.
- Consecutive sentences imply a valid paragraph.
- Sequential paragraphs imply a valid document.

Now ask me a difficult question I may not know.

Colin James III info@cec-services.com

Appendix

Karl Popper's proof of God

Meth8 on Karl Popper proof Ex(Gx)

Demarcation between science and metaphysics (1972)

- "Science is testable and falsifiable, but metaphysics is not." So, prove the *arch-metaphysical assertion* that "There is a personal spirit named God who is omnipresent, omnipotent, omniscient."
  - Once asserted it's not disprovable (Fischer P=1) per Carnap.
- If morality is non physicalistic, then not the moral Christian God.
- However, this counter example proves *morality is physicalistic*:

  When the existentialist utters "I ought to" conscience is invoked, and the moral imperative is asserted. Thus Ex(Gx) becomes a moral God.
  - What forms of monotheism exist other than Orthodox Christianity?
    - Baha'i, Judaism, Muhammadanism
  - By what reasons do they admit they are not truthful?
    - No avatar; Revelation ceased; Impersonal contradictory rules

Meth8 scripts: Popper predicates 1

Meth8 scripts for *a,b,c,d* **as** `p,q,r,s`	**Predicates**	**Descriptions**
`1: p&q;`	1: Pos(a,b)	1: *a occupies a position in region b*
`2: (p&q)>r;`	2: Put(a,b,c)	2: *a* can put thing *b* into position *c*
`3: p&q;`	3: Utt(a,b)	3: *a* makes the utterance *b*
`4: p&q;`	4: Ask(a,b)	4: *a* is asked the truth of *b*
`5: (%p&#q)>(p&#q);`	5: Opos(a)=((Ea)(b)Pos(a,b)>(b)Pos(a,b))	5: *a* is omnipresent
`6: ((%p&#q)>#r)>((p&#q)>#r);`	6: Oput(a)=((Ea)(b)(c)Put(a,b,c)>(b)(c) Put(a,b,c))	6: *a* is omnipotent
`7: (p&q)>(p&q);`	7: Th(a,b)=(Ask(a,b)>Utt(a,b))	7: *a* thinks *b*
`8: (p&%q)>(p&%q);`	8: Thp(a)=(Eb)Th(a,b)	8: *a* is a thinking person
`9: (((p&%q)>(p&%q))&~(p&#q)) V(p&#q);`	9: Sp(a)=(Thp(a)&((b)~Pos(a,b))VOpos(a))	9: *a* is a (personal) spirit
`10: (q&r)>((p&(q&r))>(p&(q&r)));`	10: Knpos(a,b,c)=(Pos(b,c)>Th(a,"Pos(b,c)")	10: *a* knows that *b* is in position *c*
`11: (q&r)>s)>((p&((q&r)>s)) >(p&((q&r)>s)));`	11: Knput(a,b,c,d)=(Put(b,c,d)> Th(a,"Put(b,c,d)")	11: *a* knows that *b* can put *c* into position *d*
`12: ((q&r)>(q&r))&((p&((q&r) >(q&r)))>(p&((q&r)>(q&r))));`	12: Knth(a,b,c)=(Th(b,c)&Th(a,"Th(b,c)"))	12: *a* knows that *b* thinks *c*

Meth8 scripts: Popper predicates 2

Meth8 scripts for *a,b,c,d* as `p,q,r,s`	**Predicates**	**Descriptions**
`13: (((p&q)>(p&q))&(p@r))&(~((r&q)>(r&q)))=~(((p&q)>(p&q))&((r&((p&q)>(p&q)))>(r&((p&q)>(p&q)))));`	13: Unkn(a)=Th(a,b)&(a≠c)&~Th(c,b))=~Knth(c,a,b))	13: *a* is unfathomable: *a* thinks b and *a* is not *c* and *c* does not think *b* is equivalent to *c* does not know that *a* thinks *b.*
`14: ((p&q)>(p&q))&(q=q);`	14: Kn(a,b)=Th(a,b)&T(b), where T(b) means b is true	14: *a* knows the fact *b*
`15: ((p&#q)>(p&#q))>(q=q);`	15: Verax(a) =((b)Th(a,b)>T(b))	15: *a* is truthful
`16: (#q=#q)>(((p&q)>(p&q))&(q=q);`	16: Okn(a)=(b)T(b)>Kn(a,b)	16: *a* is omniscient
`17: ((p&#q)&((p&#q)>#r)=(((#q=#q)>(((p&q)>(p&q))&(q=q)))&(((p&#q)>(p&#q))>(q=q));`	17: (Opos(a)&Oput(a))=(Okn(a)&Verax(a))	17: *a* as omnipresent and *a* as omnipotent is equivalent to *a* as omniscient and a as truthful
`18: ((((%p&#q)>(p&#q))&(((%p&#q)>#r)>((p&#q)>#r)))>((#q=#q)>(((p&q)>(p&q))&(q=q))))&((((((p&#q)>(p&#q))>(q=q))&(((p&%q)>(p&%q))&~(p&#q))V(p&#q)))&((((((p&q)>(p&q))&(p@r))&(~((r&q)>(r&q)))=~(((p&q)>(p&q))&((r&((p&q)>(p&q)))>(r&((p&q)>(p&q))))))));`	18: Ex(Gx)=(((Opos(a)&Oput(a))>Okn(a))&((Verax(a)&Unkn(a))&Sp(a)))	18: There exists a personal spirit named God whose omnipresence and omnipotence implies omniscience, and who is truthful and unfathomable.

Meth8 validation tables: Popper 3

Table fragments for two of the four rows (The designated truth values are **T** and **E**.)

Expression	Model 1	Model 2.1	Model 2.2	Model 2.3.1	Model 2.3.2
18-5. Validated	**TTTT TTTT**	**EEEE EEEE**	**EEEE EEEE**	**EEEE EEEE**	**EEEE EEEE**
4. (p&q);	FFFT FFFT	UUUE UUUE	UUUE UUUE	UUUE UUUE	UUUE UUUE
3. (p&q);	FFFT FFFT	UUUE UUUE	UUUE UUUE	UUUE UUUE	UUUE UUUE
2. (p&q)>r;	TTTF TTTF	EEEU EEEU	EEEU EEEU	EEEU EEEU	EEEU EEEU
1. (p&q);	FFFT FFFT	UUUE UUUE	UUUE UUUE	UUUE UUUE	UUUE UUUE

