新 組み込みソフトへの数理的アプローチ

～形式仕様記述をどのように使うか～

第2回

推論の正しさの検証
――真理表を使いこなす(2)

藤倉 俊幸

はじめに

● 前回の復習：命題論理を使う

前回は，命題論理を使って推論の正しさを示す方法について説明した．推論と言うと堅苦しいが，「何かを説明して納得してもらう」説明の仕方のことである．説明の飛躍や条件の不足などを，説明文を論理式に変換して，説明全体の真理表を作ることで判定する方法を示した．また，必要な条件がすべてそろっていても説明の仕方が悪いと真理表が真にならないことも示した．

「真理表を作る」方法については，BASICなどで個別にプログラムを作る方法と，真理表を生成するPrologプログラムを利用する方法を紹介した．手作業ではなく，ツールで一括生成できるところがポイントである．この連載を読んでいる人は，ぜひ真理表を生成する手段を確保してほしい．

●「真理表が真になる」が意味すること

「真理表が真になる」というのは，論理式を評価した結果がすべて真になることである．くどい説明をすると，論理式を評価するというのは，論理式に現れる命題変数に真か偽を割り当てて論理値を計算することで，一つの割り当て方に対して一つの論理値が計算される．割り当て方は，何通りもある[注1]．そのすべての割り当て方を試した結果がすべて真になるということである．ちなみに，「すべて真にならない」というのは，偽になるものが一つはあるということである．偽になる組み合わせが，「説明の飛躍や条件の不足など」に対応する．

「論理式を作って判定する」というのは，推論を表す論理式の真理表を作ってすべての評価が真になるか調べることである．推論を表す論理式ができてしまえば，後はプログラムにかけるだけで，その論理式が常に真になるかどうかわかる．ちなみに，常に真になる論理式は，定理とかトートロジと呼ばれる．

論理式を使った判定を行う上で重要なのは，どうやって「推論を表す論理式」を作成するかだ．頭の中で考えていることや与えられた文章から命題を選び出し，その命題を考えている筋道に従って組み合わせることで，「推論を表す論理式」ができあがる．これが，形式化ということである．今回は，例題をいくつか取り上げて，この形式化について説明する．

1 前回の宿題

● タスクの例：まず，命題を決定する

「このタスクが動くと，この変数が破壊される．この変数が破壊された．よって，タスクが動いている．」この推論は正しいだろうか．というのが前回の宿題であった．「論理式を作って判定する」手順と解答を**図1**に，考え方を**図2**に示す．

まず，文章を読んで何を命題にするかを考える．命題は真偽を確認できる簡単な文章のことである．もとの文章から小さな文章を切り出す感じである．この場合は，「タスクが動く」と「変数が破壊される」を選んだ．

● 命題で表現する

次に，前提と結論の文章を選択した命題で表現する．前提1の「タスクが動くと，この変数が破壊される」という文章は，**図1**ではA→Bと表現されている．これは，「タスクが動く，ならば，変数が破壊される」という文章に対応する．この文章が，もとの文章と同じ意味かどうかを考える．命題として切り出した小さな文章は，変数として扱わ

注1：命題変数がn個あったら2^n通りある．

KEYWORD ―― 命題論理，真理表，論理式，推論

●もとの文章の構造
前提1　このタスクが動くと，この変数が破壊される．
前提2　この変数が破壊された．
結　論　よって，タスクが動いている．

●命題割り付け
A：タスクが動く
B：変数が破壊される

●文章単位の論理式
前提1　A→B
前提2　B
結　論　A

●推論全体の論理式
((A→B)∧B)→A

A	B	値
T	T	T
T	F	T
F	T	F
F	F	T

図1　宿題の手順と解答

●論理式の真理表
- 論理式に現れる命題に網羅的に真偽を割り当てて論理式を評価した結果を表にしたもの

●推論を表す論理式の真理表はすべて真になる
- 偽になる部分は推論のモレに対応する

●推論を表す論理式
- 推論自身は大きな条件式（ならば式）である
  - すべての前提をandで結合したものを前提として，結論を結論とする条件式
  - （前提1∧前提2∧…∧前提n）→結論

図2　宿題を解く考え方

れる．その変数と論理記号→∧∨¬と結合順を示す（ ）を使って，もとの文章を再構成できるかどうか確認する．命題論理では，時制は現在のみなので多少変な文章になる場合があるが，細かいことは気にしない．また，形容詞や副詞は取りあえず無視する．

前提2と結論は，命題そのものなのでそのまま使用する．前提2は過去形で，命題Bは現在形だが気にしない．自然言語を無理やり命題論理に押し込めた結果は，不思議の国のアリスのへんてこな文章になってしまう(1)．不思議の国の仕様書でも，意味が通じれば役に立つ．

● 論理式を作成する

必要な文章すべてを論理式に変換できたら，次に文章全体を論理式にする．この作業は機械的で，前提をすべてANDで接続して「ならば」で結論に接続すればよい．そして，推論全体を表す論理式ができたらその論理式の真理表を作る．前回紹介したツールをダウンロードするのが面倒な人や使い捨てプログラムを作りたくない人は，**図3**のように論理式の優先度に従って順に計算すればよい．繰り返しになるが，推敲過程で論理式は何度も変更しなければならない．その際に，手作業で計算していたのでは非効率で，必ずどこかで間違うので結果も信用できない．ツールの使用を検討するべきである．

● 真理表の評価

真理表ができたらそれを評価する．この場合，**図4**に示したようにFalseになるところがあるので，この推論は間違っていることになる．タスクが動かず，変数が破壊される場合があると，この推論は成立しない．だから，そのような場合がないことを示さなければ話が終わらない．つまり，問題にしている変数を書き換える別のタスクや割り込みハンドラが存在しないか，存在しても議論している条件下では動かないことを示す必要がある．

この手の推論間違いはよくある．「犯人ならばアリバイがない」は正しくても，アリバイがないだけで犯人と決めつけることはできないのと同じである．アリバイのない人は世界中に何人でもいる．言えることは，「アリバイがあれば犯人ではない」である．

(1) まず，A→Bの列を「ならば」の定義にしたがって作る

A	B	A→B	(A→B)∧B	((A→B)∧B)→A
T	T	T	T	T
T	F	F	F	T
F	T	T	T	F
F	F	T	F	T

(2) まずA→Bの列とBの列でANDを取る

(3) 最後に，再び(A→B)∧Bの列とAの列で「ならば」の定義にしたがって最終結果を作る

図3　((A → B) ∧ B) → A の真理表（手計算の場合）

2 別の例題

● 郵便物からの推論の例

もう一つ別の例題をやってみよう．この例題は，参考文献(2)に載っていたものだ．

A	B	((A→B)∧B)→A
T	T	T
T	F	T
F	T	F ← 問題あり
F	F	T

図4　((A → B) ∧ B) → A の真理表