

# 条件記述の解釈に基づく確率・期待値問題の自動解答

岩間 純輝 佐藤 理史 宮田 玲 小川 浩平  
名古屋大学大学院工学研究科  
iwama.junki.i0@s.mail.nagoya-u.ac.jp

## 概要

本稿では、高校数学レベルの確率・期待値問題に自動解答するための方法を提示する。確率・期待値問題を解く上では、文章題の中で普遍的に現れる条件記述を解釈することが重要であり、それを実行可能なコード列に変換することが、中心的課題となる。オンラインソースから広く集めた問題で作成したデータセットを用いて、正答率を評価する実験を行ったところ、約 87%の問題を解くことができた。

## 1 はじめに

確率の文章題を自動解答する取り組みは、1971年の Gelb の研究 [1] まで遡る。この研究では、事前に用意した木構造のパターンリストを使用して動詞や試行回数などの解答に必要な情報を抽出し、自動解答を実現した。Dries ら [2] は、限量子によりモデル化した問題文を、ベイジアンネットワークに変換して条件付き確率問題として解く end-to-end の確率問題ソルバを開発した。

日本語で書かれた問題文に自動解答する取り組みとしては「ロボットは東大に入れるか」プロジェクト [3] がある。この中で、数学の問題に対しては、論理式で記述した数式から、限量子を QE 法で取り除くことによって解を求める手法が主に採用されたが、この手法が通用しない確率・期待値問題の自動解答の研究 [4, 5] は、あまり進まなかった。

確率・期待値問題には、条件記述（事象を規定する条件の記述）が広く登場する。条件記述は、次のような特徴を持つ。

1. 試行結果である標本空間が得られて、初めて解釈できる。（それ単体では解釈できない。）
2. 記述の内容は、かなり複雑になり得る。

本稿では、条件記述の解釈に焦点を当て、高校数学レベルの確率・期待値問題に自動解答するための方法を提示する。この方法の特徴は、条件記述の構文

2つのサイコロを同時に振る。	(s0)
出た目の和が 10 以上となる確率を求めよ。	(s1)
出た目の和の期待値を求めよ。	(s2)
出た目の和を $S$ とする。	(s31)
$S$ の期待値を求めよ。	(s32)
得点を次のように定める。	(s41)
a. 目の和が偶数のとき、 大きい方の目を得点とする。	(s42)
b. 目の和が奇数のとき、 小さい方の目を得点とする。	(s43)
得点の期待値を求めよ。	(s44)

図 1 確率・期待値の文章題の典型例

解析を、組み立てるべきコード（関数）の型情報に基づいて行う点にある。

## 2 確率・期待値問題の構造

図 1 に、確率・期待値の文章題の典型例を示す。

確率・期待値の文章題では、まず冒頭で、標本空間（起こりうる全ての結果を要素とする集合）の定義が示される。サイコロの問題では、次のような試行文と試行繰り返し文によって、標本空間が定義される。

**試行文** 〈個数〉のサイコロを（同時に）振る

**試行文** サイコロを〈回数〉振る

**試行繰り返し文** これを〈回数〉繰り返す

図 1 では、試行文 (s0) のみが示されているので、個数は 2 個、回数は 1 回となり、標本空間は (1, 1), (1, 2), ..., (6, 6) の 36 個の結果から構成される集合となる。

確率の計算は、次のような形式の**確率計算指示文**で指示される。

**確率計算指示文** 〈条件記述〉確率を求めよ

この文は、ある特定の事象（= 標本空間の部分集合）が生起する確率の計算を要求する文で、その事象は〈条件記述〉によって規定される。文 (s1) における〈条件記述〉は「出た目の和が 10 以上となる」である。

求める確率  $P(A)$  は、以下の式により計算できる。

$$P(A) = \frac{|A|}{|\Omega|}$$

ここで、 $\Omega$  は標本空間、 $A$  は事象、 $|\cdot|$  は集合の要素数を表す。すなわち、標本空間の全要素に対して、〈条件記述〉の成否を判定できれば、確率を求めることができる。

一方、期待値の計算は、次のような形式の期待値計算指示文で指示される。

**期待値計算指示文** 〈演算記述〉の期待値を求めよ

この文の〈演算記述〉は、標本空間を定義域とする関数の計算法に対応する。文 (s2) における〈演算記述〉は「出た目の和」である。

求める期待値  $E$  は、以下の式により計算できる。

$$E = \frac{1}{|\Omega|} \sum_{r_i \in \Omega} s(r_i)$$

ここで、 $r$  は標本空間の要素 (結果)、 $s$  は数値を返す関数を表す。すなわち、標本空間の全要素に対して関数  $s$  の値が計算できれば、期待値を求めることができる。〈演算記述〉は、この関数  $s$  の記述である。

確率・期待値問題は、上で述べたような文以外に、名付けを含む文 (s31) や場合分けの記述 (s41, s42, s43) を含む場合がある。これらの要素も、〈条件記述〉と〈演算記述〉から構成されるため、この2種類の記述を正しく解釈して計算可能とすることが、問題を解くための中心的課題となる。

### 3 確率・期待値問題ソルバ

作成した確率・期待値の文章題を解くソルバの入力は、確率・期待値の文章題 (テキスト) であり、出力は、解答となる確率または期待値である。

文章題テキストの冒頭部の試行文に対しては、簡単なパターンマッチングを用いて、その問題の標本空間を表すインスタンスを生成する。確率・期待値計算指示文に対しては、その文に含まれる〈条件記述〉および〈演算記述〉を抽出し、それらを実行可能なコード列に変換し、そのコードを標本空間インスタンスの各要素 (結果インスタンス) に適用することによって答を得る。〈演算記述〉は〈条件記述〉の一部として含まれるため、以下では、〈条件記述〉をコード列に変換する方法と、そのコード列を実行する方法を説明する。

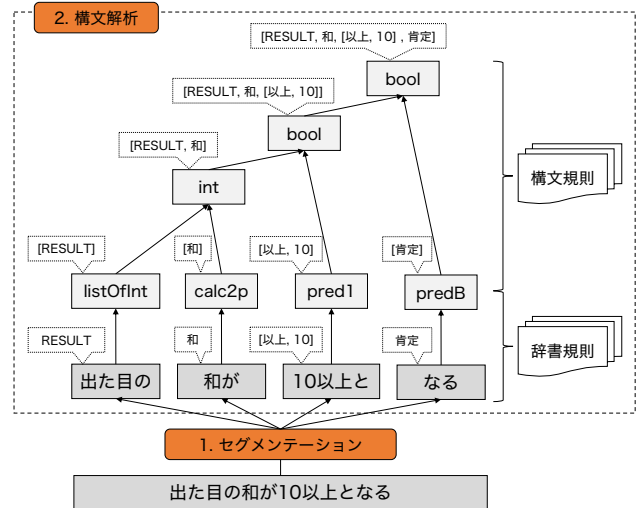


図2 条件記述をコード列に変換する手順

### 3.1 データ構造

標本空間の要素である結果や、条件記述を解釈する過程で発生する演算結果を表現するために、次のような計8種類のデータ構造を用意する。

1. 基本的 (プリミティブ) なデータを保持する— Int (整数), Str (文字列), Bool (真偽値) の3種
2. 確率問題を解くためのドメイン固有なデータを保持する— Object, List, Sequence の3種 [6]
3. 特殊なデータを保持する— Or, And の2種

これらのうち、次の4種類は、子要素として複数のデータ構造を保持する。

- List — 順序なしリスト
- Sequence — 順序ありリスト (列)
- Or — 複数の候補を OR として保持する
- And — 複数の候補を AND として保持する

これらのデータ構造  $X$  が、データ構造  $Y$  を子要素として持つとき、そのデータ構造を、単に ' $X$ ' ではなく、' $X$ Of $Y$ ' と区別して表す。たとえば、試行文 (s0) 「2つのサイコロを同時に振る」から得られる結果のデータ構造は、ListOfInt である。

### 3.2 条件記述のコード列への変換

〈条件記述〉をコード列に変換する手順を、図2に示す。この変換は、以下の3ステップで構成される。

#### 3.2.1 セグメンテーション

まず、日本語で書かれた条件記述を、基本要素に基づいて分解する。この基本要素の一覧を、付録に

示す。たとえば、文 ( $s_1$ ) の (条件記述) は、次のようなセグメントの列に分解される。

(1) [出た目の、和が、10 以上と、なる]

それぞれの基本要素 (ただし、属性値と属性名を除く) には、その内容に対応するコード列 (組み込み関数名) が定義されている。この組み込み関数には、具体的な計算法以外に、入力データの構造と出力データの構造が定義されている。たとえば、コード「偶数」は、Int (整数) を受け取って、それが偶数であるかどうかを判定し、その結果を Bool (真偽値) として出力する。

基本要素は、6 種類の大分類、16 種類の中分類に分類されている。この分類は、組み込み関数の、入力 (引数) のデータの構造の種類と数、出力のデータの構造の種類に基づいて区分されている。たとえば、大分類「数値演算」は Int を受け取って Int を出力するコードを持つ基本要素の集まりであり、引数が 1 個であれば calc1、2 個以上であれば calc2p に下位分類される。

### 3.2.2 構文解析 (1) – 辞書規則の適用

次に、分割された各セグメントを終端記号と見なし、それぞれに辞書規則を適用して前終端記号とそれが保持するコード列を得る。この前終端記号は、そのセグメントが含まれる基本要素の中分類名である。ただし、中分類が「属性名」である場合は、結果を表すデータ構造とする。保持するコード列は、終端記号のコードを要素とするコード列である。

たとえば、セグメントの列 (1) に辞書規則を適用すると、次のような前終端記号の列が得られる。

(2) [ListOfInt, calc2p, pred1, predB]

「出た目」は中分類「属性名」に属するため、このセグメントに対する前終端記号は、結果のデータ構造である ListOfInt (複数の Int を要素に持つ List) となる。この仕組みにより、標本空間の結果の型を考慮した解析を実現する<sup>1)</sup>。

### 3.2.3 構文解析 (2) – 構文規則の適用

その後、構文規則を順次適用して、開始記号まで組み上げる。開始記号は、(条件記述) の場合は Bool、(演算記述) の場合は Int である。

たとえば、次の構文規則は、「ListOfInt (出た目)」

1) 「2 個のサイコロを同時に振る」場合の「出た目」は複数の目 (ListOfInt) を意味するが、「1 個のサイコロを振る」場合の「出た目」は単一の目 (Int) を意味する。

と「calc2p (和が)」をまとめて、「Int (出た目の和が)」を構成する際に使われる。

Int  $\rightarrow$  ListOfInt calc2p

この規則は、複数の整数 (ListOfInt) に対して 2 個以上の整数を引数とする関数 (calc2p) を適用すると、その結果として整数 (Int) が得られることを意味している。構文規則は全部で 120 件ほど存在する。

同時に、構文規則は、コード列を組み上げる機能を持つ。上記の規則の場合、右辺の第 1 要素のコード列 [RESULT] と第 2 要素のコード列 [和] を連結したものが、左辺のコード列 [RESULT, 和] となる。

このような構文規則の適用により、入力開始記号まで組み上がると、入力に対するコード列が得られる。「出た目の和が 10 以上となる」に対しては、最終的に次のようなコード列が得られる。

(3) [RESULT, 和, [以上, 10], 肯定]

## 3.3 コード列の実行

得られたコード列の各要素は、データ構造に対して定義されている関数呼び出しに対応する。そのため、このコード列を結果インスタンスに対して適用すれば、求める値が得られる。

たとえば、上記のコード列では、まず、結果インスタンス  $r$  (ListOfInt) に対して RESULT が呼ばれ、それ自身  $r$  が得られる。次に、 $r$  に対して関数「和」が呼ばれ、Int インスタンス  $i$  が得られる。さらに、 $i$  に対して関数「以上 (10)」が呼ばれ、Bool インスタンス  $b_1$  が得られる。最後に、 $b_1$  に対して関数「肯定 — true かどうかを調べる」が呼ばれ、Bool インスタンス  $b_2$  が得られる。こうして得られた最終的な  $b_2$  が true であれば、その結果インスタンス  $r$  は条件を満たすと判定される。

## 3.4 特殊なデータ構造の使用

特殊なデータ構造である Or や And は、次のような問題を解く際に必要となる。

(4) a. サイコロを 3 個振る  
b. いずれか 2 つの目の和が偶数となる (確率)

(条件記述) (4b) には、以下の構文規則が順に適用され、その結果、コード列 (5) が得られる。

OrOfListOfInt  $\rightarrow$  anyN ListOfInt

OrOfInt  $\rightarrow$  OrOfListOfInt calc2p

Bool  $\rightarrow$  OrOfInt pred1

表1 実験結果 (正答数)

	基本	名付け	場合分け	計
サイコロ	524/589	90/109	2/5	616/703
玉	78/88	21/27	0/2	99/117
計	602/677	111/136	2/7	715/820

Bool → Bool predB

(5) [RESULT, [いずれか, 2], 和, 偶数, 肯定]

考え方は、次の通りである — 「いずれか2つの目」という表現は、3つの目  $a, b, c$  のうち、任意の2つ  $(a, b), (b, c), (c, a)$  のいずれかを指し示す表現であり、これを OrOfListOfInt というデータ構造で表現する。これに「和が (calc2p)」を適用すると、 $a + b, b + c, c + a$  を計算し、それを OrOfInt というデータ構造でまとめて出力する。これに「偶数と (pred1)」を適用すると、Or の子要素それぞれが偶数であるかどうかを調べ、1つでも偶数であれば true、すべて奇数であれば false (つまり Bool) を返す。

このような仕組みを導入することにより、複数の候補を指し示す表現を解釈可能としている。

## 4 実験

3節で述べたソルバを用いて、確率・期待値文章題の正答率を評価した。

### 4.1 実験設定

実験では、Q&A サイトなどのオンラインソースから広く収集して構築した、確率・期待値文章題 820 問から構成されるデータセットを用いた。このデータセットは、ソルバ開発で参照しなかった。

収集した問題は、次の条件を満たす問題である。

1. 題材が、サイコロもしくは玉であるもの。
2. 図表の絡まないもの。
3. 標本空間が有限であるもの。

なお、データセットの作成に当たっては、数字表記の統一、句読点の統一、明らかな誤りの修正を行うとともに、枝間が存在する問題は、枝間毎に分割して独立した問題とした。

### 4.2 実験結果と分析

実験結果を、表1に示す。用意したデータセットの約 87% の問題を解くことができた。なお、玉を題材とした問題は、標本空間の定義をテキストから自動抽出せず、手入力を与えた。

#### 4.2.1 正解できた問題

正解できた問題の具体例を示す。

- (6) a. 少なくとも1つは素数の目が出る (確率)  
 b. 出た目の積または和が3の倍数となる (確率)  
 c. 1回目に偶数の目が出て、2回目に6の約数の目が出る (確率)

事前に用意した基本要素の範囲を超えない範囲で記述されたものであれば、上で示したような複雑な (条件記述) であっても、正しく解釈し、問題に正解することができた。

#### 4.2.2 正解できなかった問題とその原因

正解できなかった問題の原因は、そのすべてが、(条件記述) または (演算記述) を適切に解釈 (コード列に変換) できなかったことによるものであった。その原因は、おおよそ、以下の3種類に大別できる。

**a. 共参照を含む場合** 作成したソルバは、共参照を解析する機能を持たないため、共参照を含む (条件記述) を解釈できない。

- (7) 1または2の目が4回、それ以外の目が2回出る (確率)

ただし、「それ (以外)」を「1または2 (以外)」と明記すれば解釈可能で、正解を出力する。

**b. 複雑な状況を端的に表す表現を含む場合** 問題文では、「交互に」や「この順に」のように、複雑な状況を端的に表す表現が用いられることがある。

- (8) 偶数の目と奇数の目が交互に出る (確率)

「交互に」は、「1回目に偶数の目が出て、2回目に奇数の目が出て、3回目に偶数の目が出て、4回目に奇数の…」という状況を表す。このような表現を一般的に扱うのは難しく、個別に組み込み関数を用意する必要がある。

**c. 対象外の問題** 現在のソルバが対象とする問題は、同一のサイコロを複数個振る (ひとつの袋から玉をいくつか取り出す) ことを繰り返す問題である。形状が異なる複数個のサイコロを振る (複数の袋からそれぞれ玉を取り出す) 問題は対象外であるため、次のような (条件記述) は解釈できない。

- (9) 大きいサイコロの目が偶数で、小さいサイコロの目が1である (確率)

このような問題は、積事象の問題として扱う必要がある。

## 謝辞

本研究は JSPS 科研費 JP22K19811 の助成を受けたものです。

## 参考文献

- [1] Jack P Gelb. Experiments with a natural language problem-solving system. 1971.
- [2] Anton Dries, Angelika Kimmig, Jesse Davis, Vaishak Belle, and Luc De Raedt. Solving probability problems in natural language. 2017.
- [3] 新井紀子, 東中竜一郎 (編). 人工知能プロジェクト「ロボットは東大に入れるか」. 東京大学出版会, 2018.
- [4] 神谷翼, 松崎拓也, 佐藤理史. 数学確率文章題の自動解答システムの開発. 言語処理学会 第 21 回年次大会 発表論文集, pp. 365–368, 2015.
- [5] 神谷翼, 松崎拓也, 佐藤理史. 題材となるオブジェクトの抽象化による確率文章題の自動解答. 人工知能学会 第 30 回全国大会 論文集, pp. 4B1–2, 2016.
- [6] 岩間純輝, 佐藤理史, 小川浩平, 宮田玲. 数学・確率問題を対象とした条件記述の自動解釈. 言語処理学会 第 27 回年次大会 発表論文集, pp. 947–951, 2021.

## 付録：基本要素の一覧

種類		データ構造	基本要素 (主要な日本語表現例)
大分類	中分類 (記号)	入力 → 出力	
属性値	数値 (Int)	(なし) → Int	整数 ( $n$ )
	文字列 (Str)	(なし) → Str	色 (赤, 青, 黄, 緑, 白, 黒)
属性名	数値	(なし) → $R$	整数 (出た目, 番号)
	文字列	(なし) → $R$	色 (取り出した色)
述語	数値 1 引数 (pred1)	Int → Bool XOfInt → XOfBool	同じ ( $n$ である) 異なる ( $n$ でない) 以上 ( $n$ 以上, $n$ より大きい) 以下 ( $n$ 以下, $n$ より小さい, $n$ 未満) 偶数, 奇数, 素数, 正, 負, 整数
	数値 2 引数 (pred2)	Int <sub>1</sub> , Int <sub>2</sub> → Bool	倍数, 約数
	文字列 1 引数 (predS)	Str → Bool XOfStr → XOfBool	同じ ( $str$ である) 異なる ( $str$ でない)
	2+引数 (pred2p)	XOfInt → Bool XOfStr → Bool	同じ (すべて同じ) 異なる (すべて異なる)
	肯定・否定 (predB)	Bool → Bool	肯定 (である, 出る) 否定 (でない, 出ない)
数値演算	1 引数 (calc1)	Int → Int XOfInt → XOfInt	逆数 剰余 ( $n$ で割った余り)
	2+引数 (calc2p)	XOfInt → Int	和 (和, 総和, 合計) 差 (差, 差の絶対値) 積 最大 (最大値, 大きい方) 最小 (最小値, 小さい方) 公倍 (最小公倍数) 公約 (最大公約数)
制約表現	制約 (cons)	XOfBool → Bool	全部 (すべて, $n$ 個とも, 両方とも) 下限 ( $n$ 個以上, 少なくとも $n$ 個) 上限 ( $n$ 個以下, 多くとも $n$ 個) 皆無 (1 個も, 1 度も) 特定 (ちょうど $n$ 個だけ, $n$ 回目だけ) 登場 ( $n$ 回目に初めて, $n$ 回目に $m$ 度目) 連続 ( $n$ 回続けて, $n$ 回目から $m$ 回目まで)
	参照 (Nth)	XOfD → D	回目 ( $n$ 回目)
	いずれか (anyN)	XOfD → OrOfListOfD	いずれか (いずれか $n$ 個)
並列表現	論理積 (and)	D <sub>1</sub> , D <sub>2</sub> → AndOfD	論理積 (かつ)
	論理和 (or)	D <sub>1</sub> , D <sub>2</sub> → OrOfD	論理和 (または)

表中において、 $R$  は試行により得られた結果を表すデータ構造、 $X$  は子要素を保持できる任意のデータ構造、 $D$  は任意のデータ構造を意味する。