仮想リンクネットワークを用いた数式構文認識

江藤 裕子　笹井 真樹　鈴木 昌和

九州大学大学院数理学研究科
〒 812-8581 福岡市東区箱崎 6−10−1
E-mail:suzuki@math.kyushu-u.ac.jp

OCR を用いた数式認識において大きさの異なる類似文字や文字の誤認識があっても安定した数式構造解析を行うことができる手法を提案する。前年度発表した最小コスト全域木を用いる手法を改善し、初期のネットワークのコストで結果の全域木候補を複数出力し、全域木に対する大域的なコストで再評価することによって高い精度で正解を得ることができるようになった。ネットワークから全域木の上位候補を求める際には、ビームサーチを用いて処理を高速化している。また、数式領域中の文字・記号間の大きさと位置の関係を文字種ごとに測定した分布図を用いて判定することによって、添え字判定の精度を上げることができた。

OCR、数式認識、ネットワーク、全域木、ビームサーチ

Mathematical formula recognition using virtual link network

Yuko Eto　Masaki Sasai　Masakazu Suzuki

Graduate School of Mathematics,Kyushu University
6–10–1 Hakozaki, Higashi-ku, Fukuoka, 812-8581 Japan
E-mail:suzuki@math.kyushu-u.ac.jp

We propose a new method of mathematical formula recognition, robust against the recognition errors of characters and similar characters of different size. We improved the method which we proposed last year. The outline is as follows: We first try to classify all characters and symbols into some groups which consist of characters located on the same line. We next construct a network joining characters by possible links of relations with cost. Finally, we obtain the result of the recognition of mathematical formulas as the spanning tree of minimum cost of the network, reevaluated by the cost reflecting global structure.

OCR, Mathematical formula recognition, network, spanning tree, beam search

1　はじめに

近年、自然科学分野において学術雑誌の電子 Journal への移行が急速に進んでいるが、科学技術分野の文献では数式が含まれているため、電子化には多大なコストがかかる。科学的分野の論文誌、書籍の電子化に要するコストを軽減するために、数式を含む文章に対応できる OCR システムの開発の重要性が増している。

これまでにも数式認識の研究は行われているが、実行速度と認識精度の両方において実用化に適したアルゴリズムとして発表された最初のものは岡本 ([1][2]) の手法と、Fateman ([3]) によるものであると考えられる。特に、岡本による数式認識の研究は、多くの文献に引用され、実用化の可能性を持つ手法として国際的によく知られている。[4][5] による手法も、岡本や Fateman と同様にトップダウンの解析手法になっており、その流れをくむものである。しかしながら、これらの手法では文字の誤認識や大きさの異なる類似文字等による影響を受けやすく、局所的な誤認識が数式の全体の構造認識を大きく崩してしまうことが少なくない。

そこで、本論文で提案する手法では次のようなことを目指している。

- 多様な活字印刷に対応できる
- 文字認識の誤りがあっても数式構文認識できる
- 部分的な数式構文認識の誤りが全体に与える影響が少ない

OCR において文字認識の誤りは避けられない問題である。そこで、誤りを含んでいることを前提に数式構文の認識をすることが必要であると考えられる。さらに実用化を目指す上では、多様な印刷文書に対応できることと安定した認識結果を出力できることが重要である。

具体的には、数式中の文字や記号のあいだに親子関係を定めることができると考え、数式全体を向きを持った木として表現する。その実現方法として、文字や記号を頂点とし、可能性のあるすべての親子関係を有向辺として持つネットワークを構成する。ネットワークから数式構造として矛盾のない全域木を局所的なコストをもとに求める。ここで矛盾のない数式構文木とは、各矩形に対し認識結果がひとつに定まった連結な木で、矩形はそれぞれ各接続方向に高々１つの子を持ったものである。さらに、求めた全域木から数式構文木らしさをコストを与えて評価し、その最小コストの全域木が正しい認識結果になるようにする。この際、事前に与えられたネットワークのコストだけでなく、全体の構造の矛盾の度合いを示すコストを加えて評価することによって、局所的な誤認識に対して安定な結果を取得する。

数式構文解析に使うパラメータには、多くの印刷文書から実際に測定した値を用いて多様な印刷文書に対応できるようにする。また、ネットワークを作る際に、各頂点には複数候補を持たせることによって、文字認識の誤りをカバーする。

我々が昨年提案した手法 ([6]) では、ネットワーク構成時における局所的なコストのみを用いて最適な全域木を求めていた。今回は、ネットワーク構成時におけるコストにより矛盾のない数式構文木を複数もとめ、それらに数式構文木としての大域的なコストにより再評価することで安定した結果を得ることができるという点で大きく改善されている。

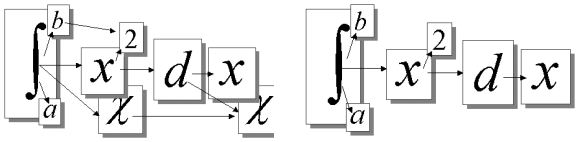


図 1: ネットワーク (左) と数式構文木 (右)

2　前提条件

数式構文認識部ではテキスト部分の文字サイズを情報として受け取り、数式領域の文字認識結果と各文字の外接矩形座標 (図 2) をもとに認識を行う。まず最初に、今回使用した一文字認識エンジンについて述べる。

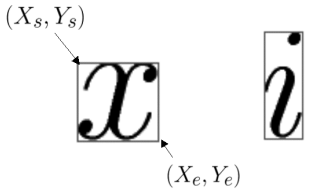


図 2: 外接矩形座標

2.1　一文字認識エンジン

本文字記号認識エンジンでは、400dpi〜600dpi の高品質画像で、英数字 (立体、イタリック体は識別する)、ギリシャ文字、$\oplus$,$\leq$ などの数学記号、約 450 種の文字を対象とする。今回、学習パターンとして日本語文書 11 冊、英文書 30 冊と TEX,Windouws,Macintosh のフォントより 400dpi で収集した約 450 文字種、合計 18 万パターンを使用した。

2.1.1　認識手法

数式中には、ベクトルなどの太字や添え字などが多く使用されるため、文字の太さ、大きさに対してロバストである必要がある。これらの影響を避けるため、3×3 の粗いメッシュ小領域を採用し、尚且つ、識別能力を落とさないために文字の輪郭線の方向線素特徴量に着目した。この特徴量は、まず文字を 3×3 のメッシュ小領域に分割し、その領域内で文字画像から、輪郭点の画素を取得し、正反対の向きを同一視した４方向 $(0°, 45°, 90°, 135°$

方向) により、輪郭点の列を方向成分の列に置き換える。このようにして各小領域内の各方向成分の個数を特徴量とする。また、高速性を実現するために多段階の大分類、絞り込み検索により候補文字を数十個に絞り込みを行い、詳細認識において、輪郭線方向線素特徴量に加え、さらに異なる性質を持つ二つの特徴量 (8×8 メッシュ濃度分布特徴量、8 分割した縦横方向からの第 1 次及び第 2 次ペリフェラル特徴量) を併用し、計三つの特徴量による認識結果を投票し、文字認識結果を決定した。

2.1.2 実験結果

1 ページ画像認識実験結果を英数字、ギリシャ文字、矢印、括弧、その他の記号に分類して、文字サイズ別 (通常文字、添え字) に集計した。実験に用いた画像は、英文書 8 冊より 600dpi でスキャンした 1 ページ画像 74 枚である。

文字種別の実験データ総数					
	英数字	ギリシャ	矢印	括弧	その他
通常文字	88997	2027	137	5398	6568
添え字	2973	694	42	237	941

以上の実験データの認識率を集計した。”S“と”s“,”1“と”l“などの同じ形状の文字は識別しないが、英数字の立体、イタリック体は識別することとする。

第 1 候補正解率　(単位 %)					
	英数字	ギリシャ	矢印	括弧	その他
通常文字	99.22	98.17	98.54	98.85	98.87
添え字	94.86	96.54	99.27	95.36	94.47

2.2 パラメータの定義

数式構文認識を行う際に使用する正規化サイズ、正規化中心を定義する。

正規化サイズ (NSize)

正規化サイズとは同一ライン上の文字が同じサイズを持つように大きさを補正したものである ([1])。ここでは、図 3 に示すアセンダー部分 (x) とディセンダー部分 (z) をあわせたサイズ $x+y+z$ とする。xyz の区別が明確でない記号については、文字の高さをそのまま正規化サイズとする。ただし、演算子については高さと幅の大きいほうを正規化サイズとする。

正規化中心 (NCenter)

正規化中心とは同一ライン上の文字が同じ高さの中心位置を持つように中心位置を補正したものである ([1])。ここでは、正規化した矩形の中心を正規化中心とした。

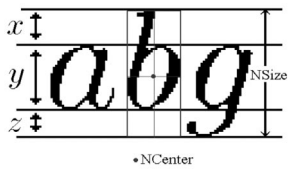


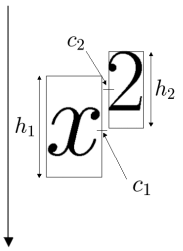
図 3: xyz 比と正規化サイズ・正規化中心

2.3 印刷文書からのデータの取得

あとで述べる同一ライン判定やネットワーク構成のための判定基準として 32 冊の数学の教科書、雑誌から実際のデータを取得した。xyz 比を $28:51:21$ として外接矩形から正規化サイズ・中心を求め、文字・記号間のサイズと位置の比較を行った。ここではその結果をまとめる。

2.3.1 正規化サイズ、正規化中心位置の比較

ページ画像から水平位置にある文字のペアと、上付き・下付き添え字の関係にある文字のペアについて正規化サイズ・正規化中心の関係を調べた。親文字の正規化サイズ、正規化中心 y 座標を $(1000, 0)$ としたときの子文字の正規化サイズ・正規化中心 y 座標 (H, D) を文字種ごとに分類してまとめた。H, D は図 4 に示すパラメータを使って次式で定義される。(c_1, c_2 は、それぞれの文字の正規化中心 y 座標を指す。)



$$H = \frac{h_2}{h_1} \times 1000$$

$$D = \frac{c_1 - c_2}{h_1} \times 1000$$

図 4: サイズと中心位置の比較

図 5 に実際に測定した値を散布図で表したものを示す。アルファベット類 (アルファベット、ギリシャ文字、数字) どうしの図において、添え字関係のデータの中で子文字のサイズが親文字のサイズ (1000) より大きくなっているのは、すべて第 1 添え字にさらに第 2 添え字がついている場合であった。ベースライン上の文字については、ほとんどの場合、水平、上付き、下付き添え字を明確に区別することができることがわかった。

親文字がアルファベット類で子文字が演算子の場合には、演算子の大きさが ”$*, +, -, \times$” のように小さいものから ”$\subsetneq, \nleq$” のように大きいものまでさまざまであり、本によっても形や大きさがが異なるため水平接続であっても広い範囲に分布している。また中心位置も本によってさまざまであった。

同様にインテグラルと、”$\sum, \prod, \bigcup$” のように上下限式を取り得る演算子を親文字として子文字にアルファベット類をとる場合についても調べたところ、ほとんどの場合は (H, D) を調べることによって添え字位置にあるのか水平位置にあるのかを区別できることがわかった。

これらの散布図をもとにデータが集中している部分を多角形で 2 段階に分けて囲み狭領域と広領域を定めておく。これらの領域を以後の判定の基準とする。

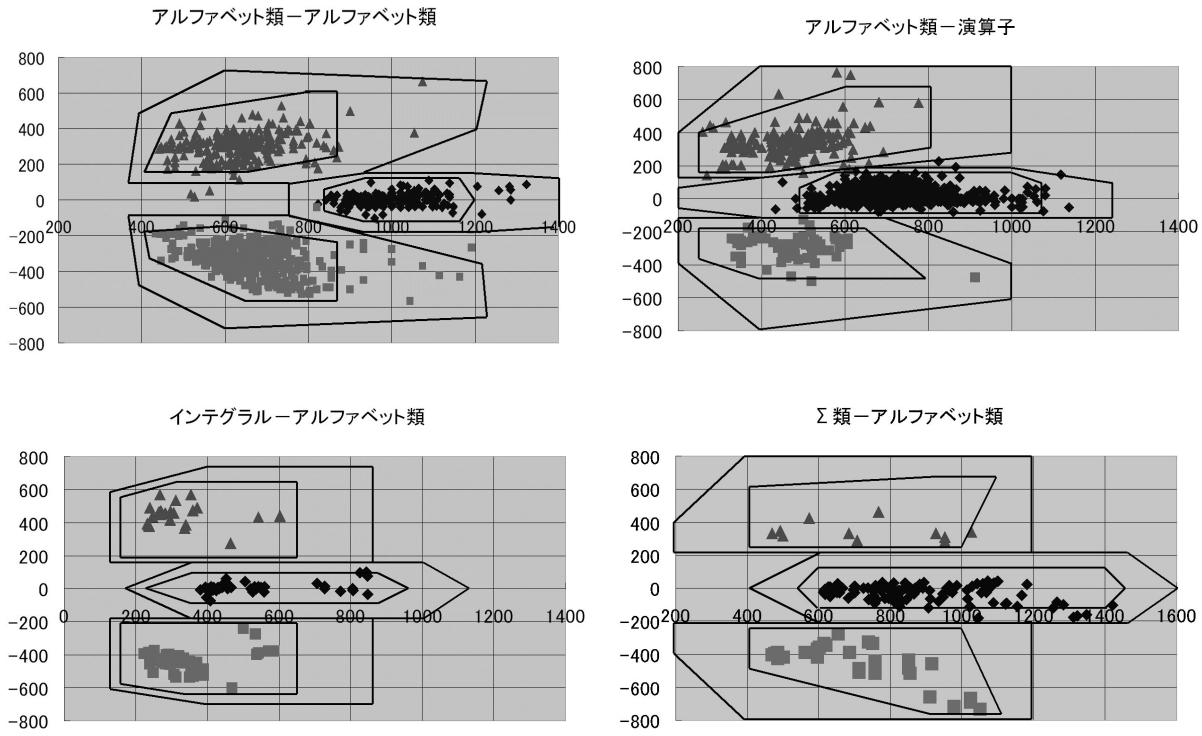


図 5: サイズと中心位置の散布図

3　ネットワークの構成

ここではネットワーク構成方法について述べる。分数線に対する分子・分母領域やルートの中の領域を判定する際の誤認識はあまり多くないため、今回の手法ではそれぞれの領域ごとに認識を行うことにする。また、アクセント記号・点類なども取り除いて認識し、最適な木構造が求めた後でアクセント記号がかかる領域や点類の挿入位置を決めることにする。

ネットワークのノードは各文字の外接矩形座標と認識結果の候補をもっており、リンクは (親候補, 子候補, リンクラベル, リンクコスト) の組を表すものとする。

今回使用した一文字認識エンジンでは、候補を最大で 10 個返し、各候補の類似度は 0 ～ 100 の値をとる。数式構文認識では文字認識結果の第 1 候補と、類似度が 50 以上の第 2 候補から第 5 候補を対象とした。この中から正規化サイズが異なるものを数式認識の候補としてノードに持たせる。さらに、”Ss,Cc,Oo” のように同形の文字どうしは、一文字認識エンジンでは識別不可能である。よって、必ず正規化サイズの異なる同形文字を数式認識候補に加えるようにしている。また、認識対象外文字としてリジェクトされている文字矩形については 4 通りの正規化サイズを考えるようにする。

3.1　同一ライン判定

数式領域においてさまざまな大きさと位置をもって並んでいる文字・記号を、同じサイズ・中心位置をもつと考えられる文字・記号のクラスに分類することを同一ライン判定と呼ぶ。ここでは、数式領域 (部分領域) 中のすべての認識候補を対象として同一ライン判定を行い、サイズとラインのラベルをつける。

同一ライン判定をおこなうことによって、外接矩形から正規化サイズの推定ができない演算子や記号類を同一ライン上のアルファベット類と同様に考えることができるようになる。

同一ラインクラスの判定方法

同一ラインクラスへの分類には前提条件で述べた水平の関係にある文字どうしの散布図を用いる。各同一ラインクラスには基準の正規化サイズと中心をもたせ、それらを $(1000, 0)$ として、判定しようとする候補文字の H, D を計算する (候補文字がインテグラル、$\sum$ 類のときには、候補文字を $(1000, 0)$ として H, D を計算する)。H, D が図 5 に示した**狭領域**に入るときに同じクラスとし、どのクラスにも入らないときに新しいクラスを作る。ただし、標準文字サイズより大きな括弧類・矢印・分数線に対しては、中心位置 (D) のみで判定をおこない、D の値が $-100 \sim 100$ のときに同一ラインと判定する。

判定は文字種ラベルの順におこなう。これは、アセンダー、ディセンダーの区別が明確なものを先に判定し、同一ラインクラスの基準に選ぶことで判定をより正確におこなうためである。

判定後、同じクラスの文字に同じラインラベルをつける。また、クラスごとの基準の正規化サイズの小さい順

にサイズラベルをつける。同一ラインクラスを基準の正規化サイズの小さい順に並べ替え、サイズラベルをつける。サイズラベルは異なるクラスであっても正規化サイズが近いクラスには同じラベルをつけるようにする。今回は、前のクラスの基準の正規化サイズより、5 ピクセル以上大きくなるときにサイズラベルを一つ大きくすることにした。

同一ライン判定についての考察

ここで述べた同一ライン判定では、アルファベット類については、添え字領域であってもほぼ正しくラインクラスへの分類をすることができる。しかしながら、分数線と演算子だけからなるラインのように、アルファベットを含まないラインについては正しくクラス分けできないことがある。また、複数候補を持っている場合に、正解と異なる候補が基準の正規化サイズを求めるときに平均を乱すことがある。この同一ライン判定において間違った判定をしても最終的な結果として必ずしも誤認識するわけではない。これらの判定の誤りはネットワークを用いる手法によって吸収される。なお、同一ライン判定の結果をもとに "$\lim, \sin$" など記述後を統合して以後、一文字として認識を行っている。この統合については、複数候補を持っている場合でも候補に統合したい文字を含んでいれば、ほぼ正しく統合できる。

3.2 ネットワークの構成

数式として可能性のある文字・記号間の親子関係をすべてリンクラベルとリンクコストを持った有向辺で結びネットワークを構成する。

接続の種類 (リンクラベル)

数式の構造をすべて、表 1 に示す 9 種類のリンクラベルであらわすことにする。ただし、アクセント記号は取り除いており、ルートの中の領域は分割して処理することにしているため、ここでは水平接続・添え字接続・上下の接続（分子分母への接続を除く）だけをネットワークとして構成することを考える。

接続可能性の判定

すべての候補文字に対して子になる可能性がある文字を探し、すべて結んでネットワークを構成する。水平接続と添え字接続の判定には前提条件で述べた散布図を使う。候補ごとに親文字の正規化サイズ・中心 y 座標を $(1000, 0)$ として、子文字の (H, D) を計算し、図 5 に示した領域に入るかどうか調べる。入る場合は、**狭領域**と**広領域**でコストに差をつけてネットワークの辺を構成する。同一ライン判定の際に付けたラインラベルとサイズラベルにより、同じラインラベルの文字どうしの水平接続の場合にはコストを低くし、サイズラベルの小さい文字から大きな文字への接続には大きなコストを与える。また、"$\leqq, \in, !$" などの演算子や記号が添え字領域の先頭にくる可能性や、添え字としての子を持つ可能性は低いので、これらの接続にも大きなコストを与えておく。今回は、これらのコストを以下のように設定した。

- 同一ラインクラスの水平接続の場合にはコストを 0 とする。
- 狭領域での接続コストを 30 とする。
- 広領域での接続コストを 70 とする。
- 添え字を取る可能性が低い文字 ("$\leqq, \in$" などの演算子、"?, !" などの記号類、開き括弧など) についての添え字接続にはコスト 100 を加える。
- 添え字位置にくる可能性が低い文字 ("$\leqq, \in$" 演算子、"?, !" などの記号類、閉じ括弧など) を子文字とする添え字接続にはコスト 100 を加える。
- 子文字のサイズラベルが親文字のサイズラベル以上の添え字接続にはコスト 100 を加える。

接続の制限

ネットワークを構成した後に、以下に述べる条件に従ってネットワークのリンクを削除しておく。これは、数式構文木を求める際に、分布以外の条件から明らかに数式として矛盾する辺を除いておくことでより正確な判定がおこなえるようにするためである。また、後の数式構文木取得の段階における組み合わせの数を減らすためにもリンクの数を減らすことは有効である。

- "$\sum, \lim$" などは、上下に接続をとる場合と添え字位置に接続をとる場合がある。しかし、それらの接続を同時にとることはない。よって、ネットワークを構成したあとで、記号の真上、真下の領域に子の候補文字があるときには、添え字の接続をすべて削除しておく。
- 下限式をとる記号・文字 ("$\sum, \lim$" など) の下接続領域を決める。これらの記号が 2 つ以上並んでいる場合には、間の下接続候補文字をどちらの領域に入れるかを決めることは難しい。また、2 つ以上並んでいない場合でも前後の文字の添え字領域と区別することが必要である。そこで、可能な限りすべての接続を探してネットワークを構成した後で、下接続領域を定めその領域の内外を結ぶ辺をすべて削除する。領域の決め方は、"$\sum$" などの下限式の親文字と上下に重なっている子の候補文字をすべて取り出し、それらの文字間のスペースの最大値を求める。その最大値をもとに下限式領域の接続ピッチを決め、ピッチを超えたところで下限式の領域を区切る。上限式も同様にする。
- ある矩形に対してネットワーク中に親を決める接続がひとつしかない場合には、その接続を必ず採用しなければ連結な数式構文木を得ることができない。また、各矩形は同じ接続方向の子を高々 1 つしか持たない。よって、親候補を一つしかもっていない矩形について、その親との接続をあらわす辺と同じ親で、同じリンクラベルを持つ辺はすべて削除しておく。

接続の種類	リンクラベル	接続の意味
水平	Horizontal	隣り合った文字間の左から右への接続
右 (左) 上添え字	R(L)SupScript	親文字から上添え字領域の先頭の文字への接続
右 (左) 下添え字	R(L)SubScript	親文字から下添え字領域の先頭の文字への接続
上	Upper	記号から上限式の先頭文字への接続 分数線から分子領域の先頭への接続 インテグラルの上にくる文字への接続
下	Under	記号から下限式の先頭文字への接続 分数線から分母領域の先頭への接続 インテグラルの下にくる文字への接続
ルート	InRoot	ルート記号からその中にある文字への接続
アクセント	InAccent	アクセント記号からその影響領域の先頭文字への接続

表 1: 接続の種類

4 無矛盾数式構文木の取得

構成したネットワークから数式構文木として矛盾のない全域木を取得する。その一つ一つを無矛盾数式構文木と呼ぶことにする。無矛盾数式構文木をネットワークのコストにより複数もとめるアルゴリズムについて述べる。このアルゴリズムには動的計画法で用いられるビームサーチを使用している。

4.1 無矛盾数式構文木

無矛盾数式構文木とは、以下の４つの条件を満たすネットワークの全域木である。

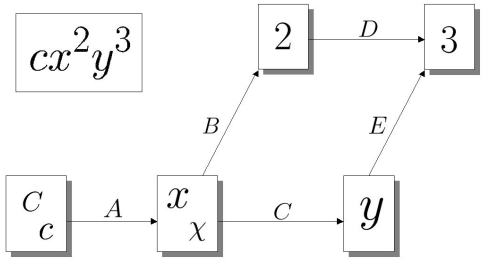
1. 各矩形が、同じ接続ラベルの子を高々一つしか持たない。
2. 各矩形が一つの全域木の中で選択する文字候補が１つである。
3. 各矩形の右添え字領域は水平方向の子よりも左側にある。
4. 各矩形 K の左添え字領域は K の親文字よりも右側にある。

4.2 無矛盾数式構文木の探索

ネットワークから数式構文木を求めていく。構成したネットワークの各リンクは

(親候補, 子候補, リンクラベル, リンクコスト)

のようにあらわされている (図 6)。このリンクを表す組は 4 つの成分のうち 1 つでも違えば異なるリンクであるとみなす。このように組にしたリンクの情報を子の矩形に持たせ、ネットワークをリンクのリストの形で書き表す。ここでリンクのリストはコストの小さい順に並べておく。ネットワークから木を構成するということは各矩形の親をひとつ決めるということである。よって、各矩形についてリンクリストから親へのリンクをひとつずつ選んだものがネットワークの全域木をあらわしている。図 7 は図 6 のネットワークを子のノードが親候補のリンクをリストで持つように置き換えたものである。ここで、各ノードのリンクリストからリンクを一つ選び、右端から左端へと結んだ一つのパスがネットワークの全域木をあらわしている。このようなパスと全域木が１対１に対応している。さらに、一つのパスがあらわす全域木が先に 4.1 で述べた４つの条件を満たしているときに矛盾のない数式構文木をあらわしていることになる。



A(c,x,Horizontal,10)
(C,χ,RSupScript,50)
(c,χ,Horizontal,100)
(C,x,Horizontal,100)
B(x,2,RSupScript,10)
(χ,2,RSupScript,50)
C(x,y,Horizontal,10)
(χ,y,Horizontal,100)
D(2,3,Horizontal,10)
E(y,3,RSupScript,10)

図 6: ネットワーク

数式領域中にノイズを含んでいたりして、正しくネットワークが構成されていない場合に数式構文木を得られないことがある。また、構文木のルートを求めるためにもすべての矩形のリンクリストにその候補自身が木のルートになるというリンクを持たせておく。これによって，連結でない場合も含めて数式として成り立つ構文木をすべて求めることができる。できる限り連結な構文木を出力するため、木のルートであるときのコストを大きく設定しておく。

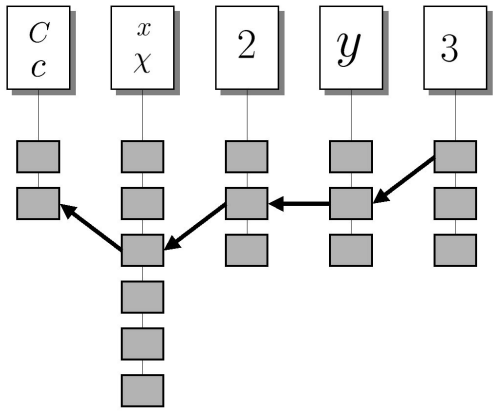


図 7: 全域木の探索

左添え字については前処理で判定し、左添え字の親矩形は左添え字を含んだ領域に拡大して考える。そうすることで、数式構文木を探索する際に上下限式を除いてリンクはすべて左の文字から右の文字へと結ばれていると考えることができる。

4.3 無矛盾数式構文木探索アルゴリズム

矛盾のない数式構文木を最大 S 個求めるアルゴリズムを以下に示す。ネットワークから数式構文木を求める際に、部分木において構文木としてのコストが小さいものから順に一定量 S だけを取り出すことにより、組み合わせ数の増大を抑えていくことにする。

準備

ネットワーク中の矩形数を N とし、外接矩形左上 x 座標 X_s の小さい順に並べる。i 番目の矩形を K_i $(i = 1,\ldots,N)$ とし、各矩形 K_i はリンクのリスト $l_{i1},\ldots,l_{im_i}$ を持つ。m_i はネットワークにおける K_i を子とするリンクの数を表す。また、i 番目以降の矩形 $(K_i,\ldots,K_N)$ の各リンクリストからリンクを一つずつ選んだときに、数式構文部分木として矛盾のないものを i-TreeList に格納することにする。

アルゴリズム

Step1 K_N の各リンク $l_{N1},\ldots,l_{Nm_N}$ をそれぞれ部分木として、N-TreeList にリンクコストのよいものから順に最大 S 個格納する。$i = N-1$ とおく。

Step2 $i = 0$ ならば 1-TreeList を結果として出力して終了。そうでないとき Step3 に進む。

Step3 $i+1$-TreeList の中の部分木 T と K_i のリンク l_{ij} のすべての組み合わせについて、これらをあわせた部分木が矛盾のない部分木かどうか、以下の 3 つの条件を調べる。

1. l_{ij} の親候補、子候補と矛盾する候補文字が T に存在しない。
2. l_{ij} の親候補とリンクラベルが一致するリンクが T に存在しない。
3. l_{ij} の親候補の矩形について、水平方向の子の正規化中心 x 座標より、添え字子部分木の領域の左端座標が左にある。

Step4 Step3 の条件をすべて満たすもののうち、リンクのコスト和が小さいものから順に最大 S 個を i-TreeList に格納する。$i \to i-1$ として Step2 へ戻る。

Step3 の条件 3 は部分木の情報をもとに判定される。よって、上のアルゴリズムのように右側から探索しながら、リンクを追加するごとに部分木の情報を更新し、リンクの親の文字に対応させて格納しておくことにする。そうすることで、i 番目の矩形に対するリンクに着目しているときには、そのリンクに関する矩形の情報だけに着目すればよいことになる。また、あとでおこなう数式構文木のコスト付けのために、部分木の最大サイズラベルを求めながら探索しておく。

矩形数を N とし、各矩形がもっているリンクの数が平均 w 個で最大値が m 個であるとする。現時点では i 番目のノードでの無矛盾判定 (Step 3) において $N-i$ に比例する計算量が必要なので、このアルゴリズムによる数式構文木探索の計算量の平均値は $O(wN^2)$ で、最悪の計算量は $O(mN^2)$ である。

5 大域的コストによる再評価

取得したすべての無矛盾数式構文木に対して以下のような数式構文木全体の構造を反映させた大域的なコスト付けによって再評価を行い、最小コストのものを最適な数式構文木とする。

1. ネットワーク構成時の各矩形のリンクのコスト和を木のコストとする。
2. 各矩形 K に対し、添え字領域の部分木の最大サイズラベルが K のサイズラベル以上のときコストを上げる。
3. ベースライン上の候補文字と同じラインラベルの文字が添え字領域にあるときコストを上げる。
4. ベースライン上のアルファベット類が異なるサイズラベルを持つときコストを上げる。

ただし、これらのコスト付けをおこなっても最小コストのものが複数存在する場合には文字認識結果の類似度の和を計算し、類似度がもっとも大きい数式構文木を最適な構文木とする。

各部分領域に対して最適な数式構文木を出力した後で、分数線に対して分子分母の領域をつなぎ、ルート記号に対してルートの中の領域をつないで、数式領域を一つの木構造で表す。

6 実験結果と考察

昨年と同じ 16 冊の本からとった数式行 161 行について認識実験を行った結果を報告する。完全正解数は 146 行であった。認識を誤った数式行のうち、lim や分数線など上下に接続を持つ文字が接触、または分離していて正

しく認識できていないものが 6 行、ネットワークの構成時に適当なリンクが得られなかったり、複数選んだネットワークの全域木に正解が含まれていなかったものが 9 行であった。各文字ついて親文字とリンクラベルがともに正解のものを数えると次のようになる。

総リンク数	正解数	正解率
5429	5402	99.5%

接触文字、分離文字、および "i" のような複数連結成分文字の統合誤り文字を切出し誤り文字と呼ぶ。また、文字認識の結果が "S" のように類似文字をもつ場合や、誤認識により正しい正規化サイズの文字が第 1 候補にない場合、またリジェクトされている場合の文字を不確定文字と呼ぶ。これらについての結果を次の表にまとめる。

	総数	正解数	正解率
切出し誤り文字	37	29	78.4%
不確定文字	806	788	97.8%

これらの文字に対しても、ある程度数式認識可能であることを示している。

認識結果の例をあげる。上が入力画像で下が認識結果である。

$$(5.3)\quad k_\alpha(\zeta,z)=\sum_{q=0}^{n-1}c_{n,\alpha,q}\frac{1}{(1-\bar{\zeta}\cdot z)^{n+\alpha-q}(1-\zeta\cdot\bar{z})^{q+1}(1-|a|^2)^n}$$
$$\times\Big[\big[(1-\bar{\zeta}\cdot z)P_{n-q-1}^{\alpha-1,-n}\bar{z}\cdot\mathrm{d}\zeta-(1-|z|^2)P_{n-q-1}^{\alpha,-n}\bar{\zeta}\cdot\mathrm{d}\zeta\big]\wedge(\mathrm{d}\bar{z}\cdot\mathrm{d}\zeta)^q$$
$$+qP_{n-q-1}^{\alpha,-n}\bar{z}\cdot\mathrm{d}\zeta\wedge(\mathrm{d}\bar{z}\cdot\mathrm{d}\zeta)^{q-1}\wedge\bar{\partial}|z|^2\wedge\bar{\zeta}\cdot\mathrm{d}\zeta\Big],$$

$$(5.3)\,k_\alpha((,z)=\sum_{q=O}^{n-l}c_{n,\alpha,q}\frac{l}{\left(l-\overline{\zeta}\cdot z\right)^{n+\alpha-q}(1-\zeta\cdot\overline{z})^{q+1}(l-|a|^2)^n}$$
$$\times[\Big[\left(l-\overline{\zeta}\cdot z\right)P_{n-q-l}^{\alpha-1,-n}\overline{z}\cdot d\zeta-\left(l-|z|^2\right)P_{n-q-1}^{\alpha,-n}\overline{\zeta}\cdot d\zeta\Big]\wedge(d\overline{z}\cdot d\zeta)^q$$
$$+qP_{n-q-l}^{\alpha,-n}\overline{z}\cdot d\zeta\wedge(d\overline{z}\cdot d\zeta)^{q-l}\wedge\overline{\partial}|z|^2\wedge\overline{\zeta}\cdot d\zeta],$$

この例では上下の添え字が重なっている部分もアクセント記号の判定についても正しく認識できている。最後の行の "$\wedge$" の判定を誤っているがそれ以外の部分への影響はない。従来手法では、

$$+qP_{n-q-l}^{\alpha,-n}\overline{z}\cdot d\zeta\wedge(d\overline{z}\cdot d\zeta)^{q-l\wedge\overline{\partial}|z|^2\wedge\overline{\zeta}\cdot d\zeta],}$$

と判定されることが多い。

$$|X_n|\leqq\left(|p|F^{\left(\beta-\frac{1}{p-1}\right)}\right)^n 1$$

この例では "p,L, 分数線" が接触している。よって "1" は接続先が見つからず木が連結にならなかったため最後に接続されている。

7 今後の課題

今後の課題として以下のようなことが挙げられる。

- 同じサイズタイプの文字に誤認識した場合の認識結果の補正は、本手法ではできない。何らかの新しい手法の導入が必要である。
- 構造を決めるインテグラルやルートなどの誤認識には現時点では対応できていない。

8 まとめ

本論文では、仮想リンクネットワークを用いた新しい数式構文認識の手法について提案した。ネットワーク構成時の局所的な関係から無矛盾数式構文木を複数取り出し、大域的な数式構文木としての構造から決まるコストによって再評価したことで、一部の誤認識の影響を受けずに安定した結果を得ることができることを示した。

また、複数の文字を数式構文認識の候補として考えることにより、一文字認識エンジンの誤認識や識別不可能な"Cc,Ss,Oo" などの類似文字にある程度対応できるようになった。さらに、パラメータを実際の印刷文書から測定した値をもとに設定したことにより、書籍・論文の種類によらず多様な活字印刷に対応できることを示した。今後、"a,α"、"l,1" のようにサイズが等しい類似文字を前後の情報や、数式としての意味などから判定することなどが課題として残されている。

参考文献

[1] 岡本 正行、トワキョンド ムサフィリ ハシム:「周辺分布特徴を用いた数式構造認識」、信学論、J78-D-II、No.2、pp366-370(1995-2)

[2] 岡本 正行、東 裕之「記号レイアウトに注目した数式構造認識」、信学論、J-78D-II、No.3、pp474-482(1995-3)

[3] R. J. Fateman, T. Tokuyasu, B. P. Berman and N.Mitchell Optical Character Recognition and Parsing of Typeset Mathematics, Journal of Visual Communication and Image Representation vol 7 no. 1 (March 1996), pages 2-15.

[4] K.Inoue、R.Miyazaki、M.Suzuki:「Optical recognition of printed mathmatical documents」、Proceedings of the Third Asian Technology Conference in Mathematics、pp280-289、(1998-8)

[5] 中山 優幸、福田亮治、鈴木昌和、玉利文和:「数学記号の特徴を用いた数式の水平分割による数式構造解析」、信学技報 (本号)

[6] 江藤 裕子、福田 亮治、鈴木 昌和:「最小コスト全域木探索を用いたオフライン数式構文認識」、信学技報、PRMU99 − 236、pp37-44(2000-2)