

Title	ナローイングの効率的実現に関する研究
Author(s)	松野, 吉宏
Citation	
Issue Date	1997-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1037
Rights	
Description	Supervisor:酒井 正彦, 情報科学研究科, 修士



ナローイングの効率的実現に関する研究

松野吉宏

平成9年 3月 19日

keyword: ベーシックナローイング、高速単一化、Tree パターンマッチング

E 単一化は、等式集合 E と項 s と t が与えられたとき E を法として $s\sigma = t\sigma$ をみたす代入 σ を求める問題であり、これを効果的に解くためにナローイングが導入された。ナローイングは項書換えの照合を単一化に拡張した書換えである。現在までに、Fay と Hullot は E が合流性と停止性をみたす項書換え系である場合には、ナローイングが E 単一化の完全な解の集合が求められること、すなわち、すべての解をカバーする一般的な解の集合が求められることを示している。

ナローイングは一般に規則の探索空間が広いため、これにたいして探索範囲を狭くする目的でベーシックナローイングが Hullot によって導入され、合流性と停止性をみたす項書換え系の上では完全であることが示されている。また、ベーシックナローイングが合流性のみをみたす項書換え系では完全ではなく、合流性をみたす右線形の項書換え系では完全であることが Aart によって示されている。ナローイングを用いて、論理プログラミングと関数プログラミングを融合させる研究も行なわれており、論理プログラミングの導出に最左最内ベーシックナローイングを組み込んだ ALF のような言語もいくつか提案されている。また、近年はレジー (Lazy) ナローイングと呼ばれるナローイングを用いて、論理プログラミングと関数プログラミングを融合させる研究も行なわれており、その一例として遅延ナローイング抽象機械 LANAM などがある。

しかし、ナローイングは書換え規則の選択とどの出現から書換え規則を適用するかに対して非決定性を持っているので一般に効率が悪い。そのため、探索範囲を狭めるための様々な戦略が提案されている。

本研究ではナローイングの効率的実現のため、単一化可能な出現と書換え規則を高速に見つけることでナローイングの高速化を図り、ナローイングの効率的実現を行うものとする。

一方で Tree パターンマッチングの高速な手法がこれまでに提案され、複数のパターンに対して、非常に効率良くパターンとマッチングする目的項の部分項が得られる。ナローイングを実現するためには複数のパターンに対して単一化可能な目的項の部分項を見つける必要があるため、この手法が単一化に応用できればナローイングの高速化が期待できる。

したがって本研究では、Tree パターンマッチングのための Tree オートマトンによる単一化に適用できるように改良し、ナローイングの効率化を図った。

実際に計算機実験により、本研究の単一化手法が、複雑なパターンに対して単一化可能な目的項を探す問題において、他の方法よりも高速であることが確かめられた。

しかしながら、本研究の単一化のための Tree オートマトンでは、パターンと目的項が共に線形である必要があるため、そのままではナローイングに適用が出来ない。

このため、まずパターンと目的項が線形であるとみなして本単一化アルゴリズムで単一化可能な候補となるパターンと目的項の部分項をあげたあとで、一般の非線形項に対応して単一化アルゴリズムを用いることで解決した。

実際に計算機実験により、この手法の有効性が確かめられた。

本論文では次の 2 点を目標とした。

- Tree マッチングオートマトンを改良し単一化のためのオートマトンの定義を与える。
これにより、単一化可能な出現と書換え規則の高速探索を可能にし、従来の線形時間の高速単一化アルゴリズムと比較し評価・検討する。
- 単一化のための Tree オートマトンをベーシックナローイングに適用し、ベーシックナローイングの効率の良い実現法の検討を行う。

ここで、単一化のための Tree オートマトンに入力する目的項とパターンは線形とする。

また、比較対象とする高速な単一化アルゴリズムは Alberto Martell and Ugo Montanari[1982] の実行時間が線形アルゴリズムとする。これを Standard ML of New Jersey で作成し Sun Sparc Station 5 において実験を行った。その結果以下のことが明らかになった。

1. 実行時間に関しては与えられたパターン集合の特性に影響されるが、本研究の手法は比較対象の単一化アルゴリズムより高速である。
2. 遷移表を作るための事前計算時間はパターン集合に影響されるが、繰り返し遷移表が使われる場合にはこのオーバーヘッドは問題にならない。