氏名	布川博士（ぬのかわひろし）
授与学位	工学博士
学位授与年月日	昭和63年3月25日
学位授与の根拠法規	学位規則第5条第1項
研究科，専攻の名称	東北大学大学院工学研究科 （博士課程）情報工学専攻
学位論文題目	**関数型言語に関する研究**
指導教官	東北大学教授 野口 正一
論文審査委員	東北大学教授 野口 正一　東北大学教授 伊藤 貴康 東北大学教授 佐藤 雅彦　東北大学助教授 白鳥 則郎

論文内容要旨

本論文では，プログラミング言語に要求されていることは，(a)記述されたプログラムが読みやすいこと（理解容易性，Readability），(b)プログラミングの為の方法論があること(Methodology) (c)形式的意味が明確であること（Formal Semantics）の3点であるとの立場に立ち，形式的意味を乱すことなく理解容易性，プログラム作成方法論を導入した使いやすいプログラミング言語の設計及び作成を行なった。特に本論文では，(c)として関数型計算モデルであるλ計算を据え，その中に如何にして(a)(b)を導入するかを提示した。それら(a)，(b)の(c)による解釈の方法，実現方法について述べた。

具体的には，理解容易性を高める記法として，抽象的な計算系である項書き換え系をすえ，それをλ計算及びそのインタプリタであるCrescendo（Cresc.）によって解釈した。またさらに，図的表現を提案しそれを同様にλ計算及びCresc.で解釈した。これらにより(a)，(b)，(c)すべてを満足する関数型言語の設計，作成を行なった。

1章は序論である。

2章では，本論文で核言語として用いるλ計算に基づく純粋な関数型言語Cresc.を設計及び作成した。

λ計算は特に関数の計算過程に注目して作られた形式体系であり，関数型プログラミング言語とのセマンティックギャップが少なく，その数学的モデル，ベースロジックとして重要であり，理論的整備も進んでいる。

black－λ（λカリキュレータ）は，このλ計算の操作的意味に着目して作成されたλ計算に忠実に従う実現系である。λカリキュレータは単にα，β－リダクションを行うのみでなく，δ－リダクションを行うことができる特長を持つため，関数型言語用の汎用評価機構として用いることができる。また，λカリキュレータはその戦略としてCall－by－Needを用いており，従って，正規項が存在する場合に最も少ない回数で必ず求めることのできる，λ計算の正しくかつ最適な実現である。λカリキュレータはLispを用いて作成された。

Gresc.はλカリキュレータ上の言語の一例であり，λカリキュレータを評価機構として用いた関数型言語である。本章では，Gresc.のシンタックスであるS式をλカリキュレータのコードへ変換するコンパイラを定義した。従ってGresc.の意味はλ計算により完全に記述されている。このようなコンパイラを定義することにより，種々のλ計算によって意味が与えられた言語が作成され，それを直接λカリキュレータによって実行させることが出来る。ここでは，プログラミング例を通してGresc.の特長を述べた。これはλカリキュレータの特長でもある。本章で述べたGresc.は3章，4章の議論のために極めて有用であった。これはGresc.の理論的基礎が明確であることに依る。

3章では抽象的記述ができ，プログラムの設計のみでなく，実現までもトップダウンに行うことのできるプログラミング言語である項書き換え系（TRS）を，本論文の核言語であるGresc.へ変換する方法を示した。その変換の正しさの証明も行なっている。さらに実際に変換系を作成，本方式に対する評価を与えた。

TRSにおける書き換えを行うシステム（TRS処理系）の実現では，正しさ（正規化戦略の実現）と速度が重要な点である。この2つは，互いに排反する条件であり，速度向上のため正規化戦略を実現しない方法もある。本方式は，TRSプログラムを一度，Gresc.のプログラムへ変換し実行するコンパイル方式であり，正規化戦略の実現，速度の向上が得られる方法である。そのために，書き換え規則に対応する関数定義のみでなく，他の補助的関数を付加し，並行最外リダクションを実現，TRS処理系の正しい実現を行なった。

作成したシステムはLispへの変換系とGresc.への変換系である。得られたLispプログラムはさらにLispコンパイラによりコンパイルすることができるため，実行に際しては3通りの方法がある。

変換系作成の点からはGresc.への変換系が，Lispへの変換よりも作成が容易でありGresc.が本変換法のためのターゲット言語として極めて有用であり，TRSと親和性が高いことが確認された。

Gresc.はLispを用いて作成されているためそれ自信の速度は早くない。そのため，速度の比較は同じUti Lispで記述されているTRSインタプリタを行った。その結果Lispインタプリタ上での実行の比較より，本コンパイル方式は約2倍の速度の向上が得られた。さらに，ともにコンパイルしたもの同士での比較することにより，ともに達成可能な最大速度での比較結果が得られるが，その

結果，本方式は3～9倍の速度向上が認められた。特に項のサイズが大きくなる程その差が顕著であった。さらに，扱える項の大きさも2～4倍の向上がみられた。以上のように本方式に対して，作成のしやすさ，速度，容量とも満足のゆく結果を得た。

4章では，データの流れに着目してプログラムを理解することができる。図式をシンタックスに持つ関数型言語Diagramaric Crese.（D－Cresc.）を提案した。D－Cresc.は一般の関数型言語の図式表現法でもあり，関数をも含むデータのやりとりを図的に記述することにより，関数型言語によって書かれたプログラムの見易さを向上させている。この関数型言語も，核言語であるCresc.へ変換され実行される。

この章では図式（モジュール）のシンタックスを明確に定め，各モジュールに対して対応するλ計算の項を割り当てることによりセマンティクスを定めた。さらに，マウス等により入力されたモジュールを，形態を保存しかつ定めたセマンティクスと同じ意味をもつCrese.のプログラムへ変換する。同形性を有するコンパイラcompを示し，その正しさを証明した。

従って，従来非形式的に用いられていた，モジュール，入出力ポート，データの入出力点，データの送受信，データの流れといった事柄について，すべてλ計算によって解釈することができ，意味が与えられた。

さらに，D－Crese.でのモジュール化技法，プログラム作成方法論を示した。また，ここで述べた統一的な考え方に沿って，流れるデータを観測することによるプログラムのデバッグ，プログラミングの支援のためのツールに関しても述べている。このようにして本章では，ソフトウェアの部品として関数を用い，その結合でプログラムを作成するための，関数型言語の図的表現方法，その作成方法論，プログラミング支援環境の理論的基礎を与えた。

5章では，4章で述べたD－Cresc.でのプログラム作成のためのシステムについて述べている。D－Cresc.を単なる言語としてみるのではなく，プログラム作成のためのエディタ，その他プログラミングのための種々の機能，ツールを全て含めたD－Cresc.システムとしてとらえ，システム全体を設計，そのプロトタイプを作成した。

D－Cresc.システムの基本的な概念は4章と同じく図的表記による使いやすさの向上にある。すなわち，プログラムのみでなくプログラミング時も，図的に記述された対象を扱うことにより抽象的に取り扱い，生産性，理解容易性を向上させることにある。

本システムは，従来のシステムとは異なり，図的に表記されたプログラムに対して直接デバッグができる点，プログラムの実行状態を表現するために本来のプログラムの意味とは異なった。新たな図的計算モデルを設定する必要がない，という点で優れている。また，図的なプログラムの意味がλ計算およびそのインタプリタCresc.で表されていることにより，デバッグで行なうべきことや，モジュール間の結合がλ計算の何に対応するかが明確にされている。そのため，プログラミングのための環境を構成するツールが行うべきことがらについてもλ計算により明らかにされた。従って，プログラミングのためのデバッグの支援に関しても，形式体系λ計算を用いることが可能になり，

λ計算，λカリキュレータ上の言語Cresc.により統一的に扱える利点を有する。このように，D-Cresc.システムは，関数型言語でのプログラミング用の環境を図的表記を用いて統合化したものである。

本章ではD-Cresc.システムの要求定義仕様を作成し設計を行なった。従って，D-Cresc.システムのユーザーからみた動きは完全に定められた。さらに，本仕様の妥当性を確かめるためにプロトタイプを作成，評価を行なった。その結果使いやすさの点から十分に満足のゆく結果を得た。

6章は結論である。

以上のように本論文では統合的，実際的立場でλ計算に基づく使いやすい関数型言語の提案，実現を行っている。その結果，従来漠然と用いられていた，プログラム作成，特に関数型言語でのプログラム作成時の事柄に関して，λ計算を用いて明確にすることができた。また，そのインタプリタであるλカリキュレータにより直接実行することを可能とした。本論文の結果は関数型言語を用いてのソフトウェア作成にとってきわめて有意義なものである。

審 査 結 果 の 要 旨

コンピュータの機能は飛躍的に拡大してきたが，より高度な処理機能を実現し，扱い易いシステムとするためには，人間とコンピュータとの間で意思の疎通を円滑に行う新しいプログラミング言語の開発が不可欠である。本論文の著者は，この立場から関数型計算モデルをベースとして新しいプログラミング言語の研究，開発を行った。本論文はそれらの成果をまとめたもので，全編6章よりなる。

第1章は序論である。

第2章では，λ計算に基づく関数型言語Crescendoを設計し，この言語の持つ特徴について述べている。Crescendoは本研究における関数型言語開発の核言語として設計された重要なものであり，その基本計算メカニズムは，λ計算理論における3つの基本的なリダクションに基づいている。このため，この言語で記述されたプログラムの形式的な意味は，λ計算の上で正しく表現される。

第3章では，関数型言語の中でも有用な項書き換え系をCrescendoに変換する方法について詳細に研究している。項書き換え系からCrescendoへの変換アルゴリズムの正当性を証明し，更にこの変換システムをLISP上で試作して実験的評価も行い，有用な知見を得ている。

第4章では，プログラムの作成，見易さを向上させることを目的とし，図的記述が可能な言語D－Crescendoの設計について述べている。この言語系を用いれば，関数を含むデータの流れを含めてプログラムを図的に表現することが可能であり，言語系の意味もλ計算を用いて正確に解釈することができる。

第5章では，第4章で設計したD－Crescendoシステムの処理系の構成法について述べている。特に，図的記述を可能とするプログラミング支援環境の問題について調べ，デバックに必要なソフトウェアツールを考案し，その設計を行っている。

第6章は結論である。

以上要するに本論文は，今後のソフトウェア工学における1つの重要な分野である関数型言語について研究し，新しいプログラム言語の開発及びその理論的基礎付けを行ったもので，ソフトウェア工学の研究に新しい知見を加え，情報工学の発展に寄与するところが少なくない。

よって，本論文は工学博士の学位論文として合格と認める。