ヒューマンインタフェース 91－7
音 楽 情 報 科 学 37－7
（2000．10．13）

LaTeX ソースの解析による編集ツールの提供

吉田 慎，渕野 昌
北見工業大学 情報システム工学科

抄録

Emacs editor を拡張して，編集するテキストの構文解析を盛り込んだツールを用意しようとする場合，Emacs の内部でのテキスト・データの扱いや，Emacs の拡張言語である Emacs lisp の処理速度に起因する制限のため，Emacs lisp でのプログラミングのみに頼った実現は実行速度の面での限界がある．本研究では，時間のかかる構文解析の部分を flex/bison を用いて Emacs の外側に実現し，このプログラムを Emacs から呼びだすことにより，Emacsl lisp だけでは実用的な実行速度の実現の困難なツールを提供する，という方針の検討を行う．このようなツールの実現の例として LaTeX のソース・テキストの表示の色分けプログラムのプロトタイプを作成する．

On realization of LaTeX tools for Emacs with parsing of TeX source codes

Makoto YOSHIDA and Sakaé FUCHINO

Dept. of Computer Sciences
Kitami Institute of Technology, Kitami, Japan

Abstract.

Tools for Emacs editor including parsing of text files to be edited are difficult to realize inside Emacs due to the limited performance of Emacs lisp. We examine here the efficiency of development of such a tool by programming the time-consuming parsing part outside Emacs using flex and bison. As an example of such a tool, we consider highlighting of LaTeX source files in Emacs buffers displayed on the screen.

1 Emacs 上のツール

Emacs エディタは 各種のプログラム言語，テキスト整形言語などへの非常に優れた text based なインターフェイスを提供する．Emacs は Emacs lisp のインタプレータとして実現されており，Emacs lisp による関数を付加することにより，ほとんど任意の拡張が可能である．実際，Emacs lisp によって実現されたいくつもの優れたソフトウエア・パーッケージが知られており，その中には，スタンダードな構成要素として Emacs エディタの最近の Versions に取りいれているものもある（[5],[2], [6], [7] などを参照）．

一方，Emacs のバッファに取り込まれたファイルに対して複雑な解析を行なおうとすると，Emacs lisp で書かれたプログラムの処理速度の限界のため，期待通りの速さでの処理が行なえない場合が往往にして起こってくる：例えば，パフォーマンスのそれほど良くないマシン上で動いている Emacs では，Emacs 上の WWW ブラウザーである W3 ([7]) では，大きな html ファイルがブラウズされるまでに何分もかかることがあるし，ファイルの色分けプログラム [6] では本一冊分の TEX のソースファイルの色分け表示を行なおうとすると，数十分かかってしまうことさえある — 実際，[6] を，このテキストを書いているノートブック・コンピュータ上 [1] のソースファイルを読み込んだ buffer で 走らせてみたところ，[6] は 30 分たっても終了しなかった（他方，[6] は正規表現のマッチングによる色分けをしているにすぎず，たとえば入れ子になった表現に対応できるようにはなっていない）．

このような処理速度の問題を克服する方針の 1 として，Emacs lisp で実現されるプログラムには負荷の大きすぎる作業を一切行わないような仕様を設定しておく，という方針が考えられる．[5] はそのような設計方針で成功した例と言えるであろう — 実際，[5] では有限オートマトンとしてのモデルを固定し，その範囲で行なえることから綿密に動作の仕様が割り出されており，そのことがこのプログラムの成功につながっている．

一方，有限オートマトンの能力範囲を超える厳密な parsing を含むプログラムを望む場合には，そこでの処理速度の問題を克服する一つの方針として，flex/bison を用いて parsing を行うプログラムを Emacs の外部に作成し，このプログラムを Emacs lisp のプログラムからサブルーチンとして呼ぶ形でツール群を実現することが考えられる．本研究ではこのアイデアの有用性の検証のために，LATEX ソースファイルの表示の色分けプログラムを作成した．

2 プログラムの概要と構造

本研究で作成したシステムでは，編集中の LaTeX のソースを外部プログラムに渡し，解析した結果を，Emacs lisp で書かれた Emacs の内部のプログラムで処理することで，LATEX ソースファイル内のトークンのカテゴリーに対応する色による表示の色分けを実現している．外部プログラムに処理させるファイル名などのいくつかのパラメータを渡し，parsing を実行させ，実行結果を Emacs のバッファに読み込みこれを評価する．

2.1 構文解析プログラム

外部プログラムは flex/bison を用い C 言語でプログラムしている．プログラムは，とりあえず日本語と英語の混在するファイルに対応するようなものにしたが，このために次のような方法をとった：日本語コードは，TEX や LATEX のマクロ名には使われていないと仮定して単純にスキップするように処理している，全角文字であることを判別しやすくするため，Emacs での mule の機能を用いて，Emacs buffer の内容を日本語 EUC でコーディングしたものに変換してファイルにはき出し，外部プログラムはこのファイルを読むようにしている．

こうすることで、0xa0～0xff までのコードが二つ続いた場合，全角文字として認識することができ，ファイルごとのコーディング・システムの違いも吸収することができる．

2.2 Emacs での全角文字のコードの扱い

プログラミングを行なったマシン上の Emacs では全角文字は 3 バイトで表されていたが，これは処理系や Emacs のバージョンによってことなる可能性がある．一方，使用した C 言語のコンパイラではこれは 2 バイトとして扱っていた．このようなこの違いのため、文字の位置の情報を Emacs 内のプログラムと外部プログラムで受け渡す場合に作業環境に依存した値の変換が必要になる．ここでは，Emacs 内のプログラムが全角文字のサイズを調べ，その値をパラメータとして外部プログラムに渡すことで，この問題を解決している．

2.3 マクロや分割ファイルについて

プログラムをトリッキーなマクロ定義を含む LATEX ソースファイルや，プログラム言語としての TEX の機能を駆使しているようなファイルにも対応するようなものとするためには，ファイルの完全な parsing が必要となるが，現段階のプログラムでは，`\input` や `\include` などの命令に対応したファイルの展開などいくつかの機能のみを実現している．

LATEX のマクロの書式は，TEX でのものとは違うが，動作はほぼ同じなので，外部プログラムでは TEX の形式に変形して扱かうようにしている．

例えば、LATEX の

```
\newcommand[2]{$\alpha$}
```

は、TEX の

```
\def\hoge#1#2{$\alpha$}
```

と同等である。

2.4 色分けの実行

テキストの表示の色分けを実行するためには，色分けされるトークンとその位置の情報が必要になる．外部プログラムでの解析結果を返すためには，外部プログラムに次のような形式のテキストファイルを生成させ，これを Emacs lisp 側の処理で用いることにした：

```
((<keyword> ((start_pos end_pos nest_level)))
(<keyword> ... ))
...
```

keyword は、トークンのカテゴリーで，これに対応して表示の色を変える．start_pos はトークンの開始位置，end_pos\verb は終了位置を表す．nest_level は，トークンが括弧の場合のネストレベルを表す．これは、括弧のネスティングの深さに依存して表示の色を変える場合などに利用する．

実際のデータは、たとえば次のようなものとなる：

```
((KAKKO_BEG ((25 26 0)(48 49 0)...))
(KAKKO_END ((34 35 0)(72 73 0)...))
(ERROR ((2010 2023)(2029 2030)...))
(COMMENT ((77 80)(80 125)...))
(KEYWORD ((1 15)(37 48)...)))
```

Emacs-lisp 側に、トークンと色の対応を定義したリストを変数として用意しておく．この変数を変更することにより，ユーザーの好みの表示の色付けを定義することができる．

3 考察

LaTeX ソースファイルの（部分的な）構文解析の応用としては，ここで扱った，Emacs 上でのキーワードの色分け表示の他，TeX や LaTeX 特有の命令や数式モードの部分を除いた、素の文章に対する検索や置換，スペルチェック，また，TeX ファイルの作成の際に頻繁に起こるマクロ名の入力の際のバックスラッシュの書き忘れのチェックなどが考えられる．このような応用では TeX または LaTeX の数式モードの正しい判定が不可欠となるが，たとえば，

```
$\{a\in X\ :\ a\mbox{ is not a subset of $B$}\}$
```

などのような数式モードとテキストモードが入れ子になったような書き方が可能であるため，数式モードの正しい検出のためには正規表現のマッチングだけでは不十分である．ある程度厳密な parsing が不可欠となる所以である．

外部プログラムによる以上のようなツールの実現のデメリットとして，システムに依存して外部プログラムを変更ないしリコンパイルしなくてはならくなる点があげられよう．Emacs-lisp のみでツールを作成した場合にはほとんど修正がいらないことと比べると，移植性はかなり悪くなてしまうと言えよう．しかし，外部プログラムの作成に用いた gcc, flex, bison は Emacs の移植されているプラットフォームにはすべて同様に移植されていると考えてよいので，本質的な移植性の低下にはつながらないと考えられる．

LaTeX に限らず，他の言語でも文法解析を行うことによって得られる効果は大であろう．例えば [6] では LaTeX ファイルのみならず，他の多くの言語に対するファイルの表示の色分けを提供している．

我々のプログラムで用いたサブルーチンに対応する部品を汎用的な形で提供するライブラリを作成しておくことにより，外部のパーサーを用いる方法においても様々な言語への対応を比較的容易にすることができるのではないかと考えられる．

参考文献およびソフトウェア

[1] T. Bartoszyński and H. Judah, *Set Theory: on the structure of the real line*, A K Peters, (1995), i–ix,1–546.

[2] Andy Norman, **ange-ftp.el** — transparent FTP support for GNU Emacs.

[3] 渕野 昌, *Emacs Lisp* で作る,
bit, vol.29, no.9 (1997), 37–45.

[4] 渕野 昌, *Emacs lisp* と *TEX* で作る （仮題）, 日本評論社, in preparation.

[5] 佐藤 雅彦 (et al.), **SKK** (Simple Kana to Kanji conversion program).

[6] Jonathan Stigelman, **hilit19.el** (Release 2.19) – customizable highlighting for Emacs19.

[7] William M. Perry, **Emacs-W3**.