TECO (Text Editor and COrrector) メモ (第二版)

1980-03-19　　上田 和紀

- 作成の動機,目的

① OS7 上に，気に入ったテキスト・エディタがなかった。

② 字エディタ，及びマクロ機能を持ったエディタを使ったことがなかった。

③ PASCAL 8000 の

a. packed 型

b. 外部サブルーチンとのリンク機能

c. 会話型入出力

によって，効率が良く，しかもOSとの整合性の良いものが，気持良く書けるのではないかと考えた。

④ 簡単な会話型システム（行エディタ，データ入力システム，reformatter 等）の試作が容易になると考えた。

- 作成の経過

1979年 8月下旬　入出力部分（アセンブラ）のみ作成，PASCALから呼べることを確認。
その他，OSとのインターフェースについて調査検討

9月初　"insymbol" と簡単なコマンドを完成

9月上旬　マクロをつけて，自分自身の自分自身による edit を開始

9月中旬～10月上旬　残りの機能をほぼ整備

10月中旬　例外処理に関する拡張

10月下旬　Qレジスタの実現方式の全面変更

- プログラムの大きさ

PASCAL　1900行（予想の50%増）

アセンブラ　百数十行　① ファイル入出力　② クイットの処理
③ ポインタの大移動

＋(プラス) AMISPACK　① DTF, DLDTF, SET 等のコマンドの発行
② FFN (Find File Name)（ファイル実体名を与えて，それに対するDTFの有無を調べる）

• 仕様 —— Multics版に準拠（ただし，Multics版のきたない所まで忠実にまねることはしない。）
TENEX版，TV（DEC 20）も参考にした。

主要変更点

① R(eplace)，V(iew) コマンドの追加

② コマンド列の最後　#← ⟶ Θ← ⟶ △←

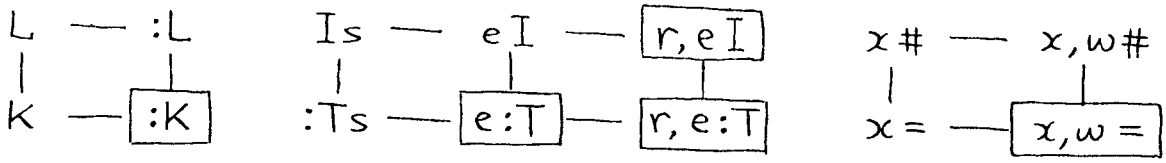
③ 引用文字列の記憶

Q" ⟶ Qα（α… delimiter）⟶ { Q/（探索文字列）; Q;（それ以外）

④ 入出力コマンド

EI(input) ⟶ ER(read)
EO(output) ⟶ { EW(write); EO(overwrite)

⑤ コマンドの整備拡張



⑥ O(goto)，"…:'…'（if…then…else…fi），<…>（loop）におけるスキップの方式

⑦ 例外処理機能

• 作成上の工夫

① 文字列探索　BM法 ⟶ BM法＋単純法

② Qレジスタ　つなぎ配置 ⟶ 順配置

③ MOVE命令の使用

④ 改行文字の取扱いの一般化

⑤ エラー発生箇所の表示，閉じていない引用文字列の記憶

⑥ その他（とるくだらない話）

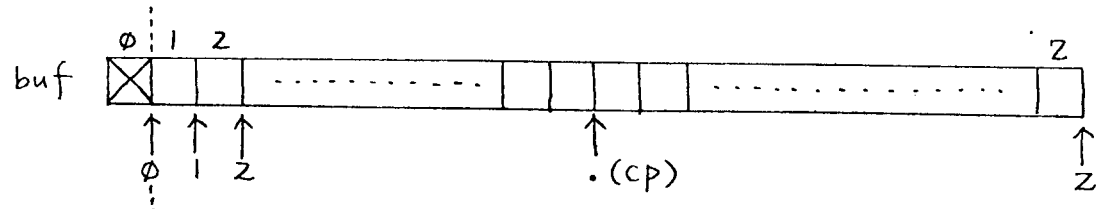
{ variable parameter　空白の扱い; procedure parameter　EOF問題; 変数宣言の順序　4バイト可変長レコード } etc.

• つけなかったもの　① 入力コマンド列の記憶，回復機能　② コンパイラ　etc.

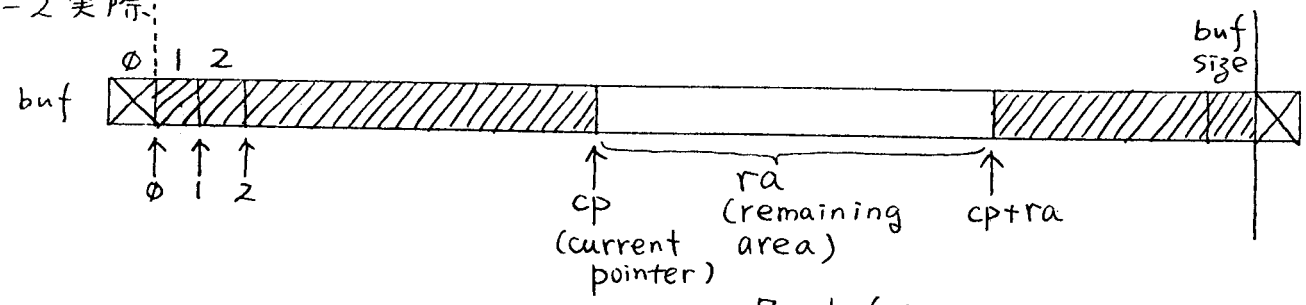
・データ構造

A. バッファ

A-1 概念上



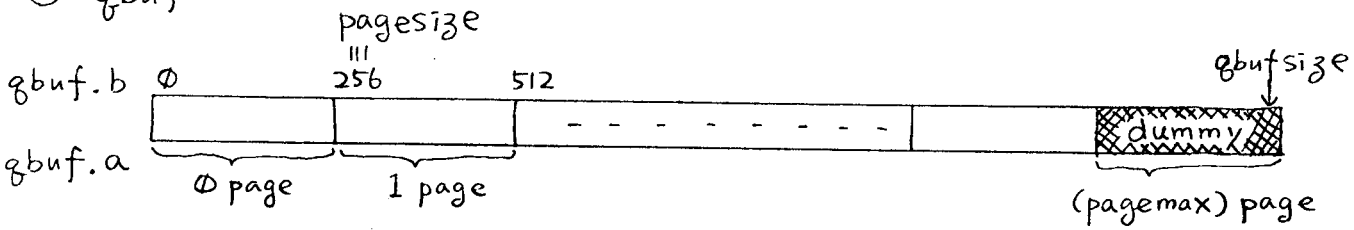
A-2 実際



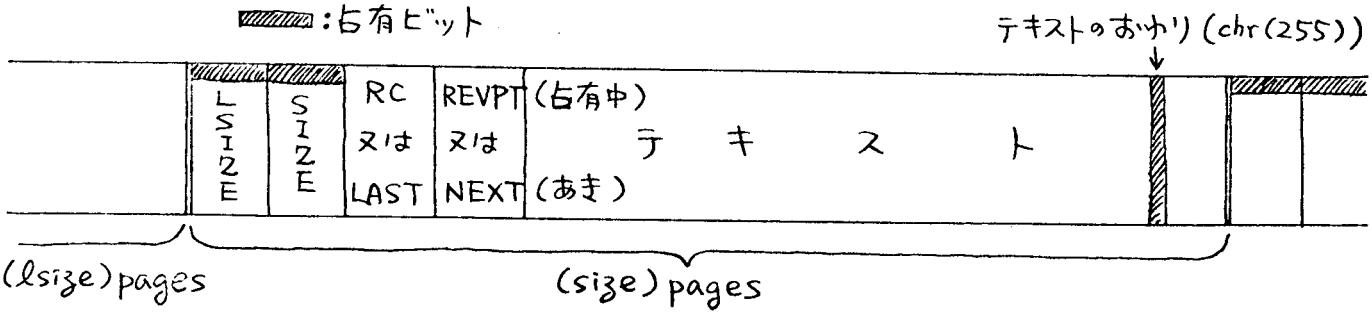
Z ≡ bufsize − ra

B. Qレジスタ・スタック関係

① qbuf



② 占有領域，あき領域 （page単位にわりつける）

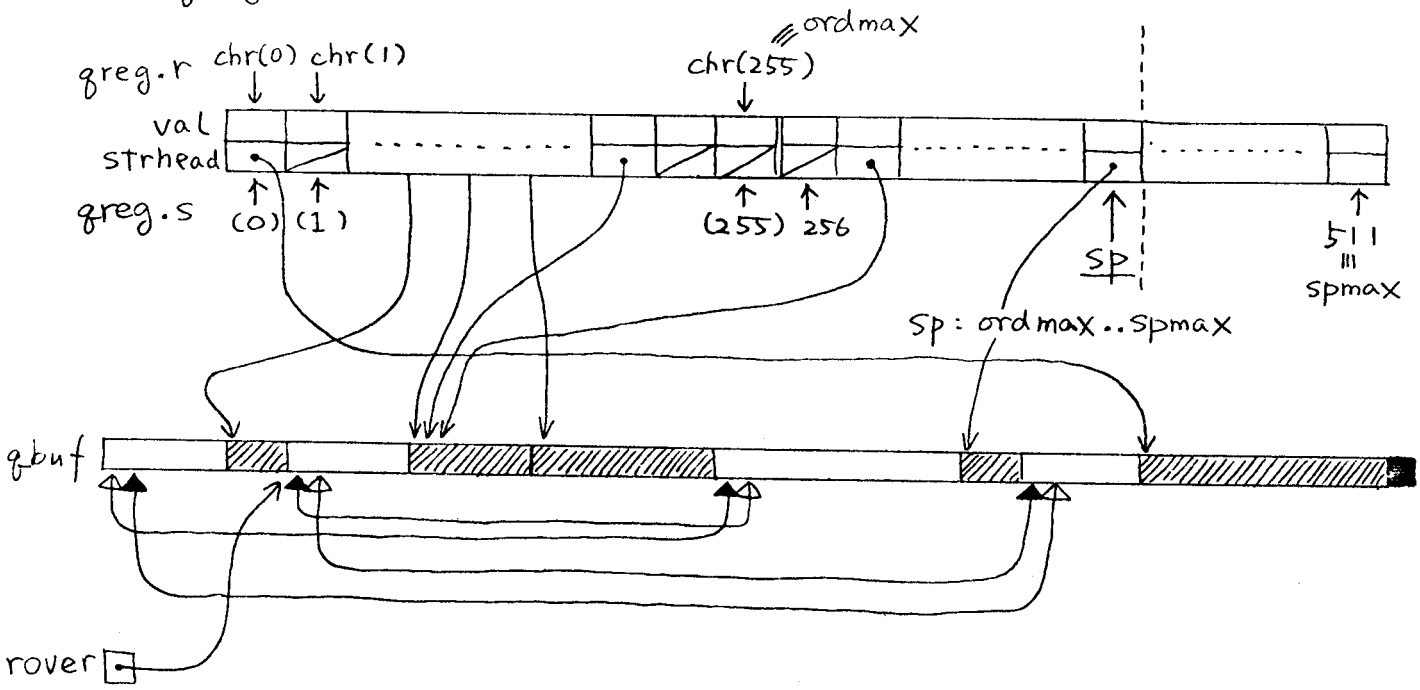


RC: 占有領域の reference counter (≧1)

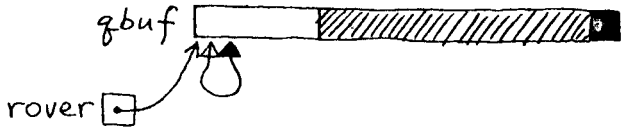
REVPT: compaction 時の reverse pointer (通常は nil ≡ −1)

LAST, NEXT: あき領域の doubly-linked circular list を構成

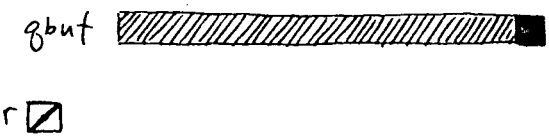
③ greg



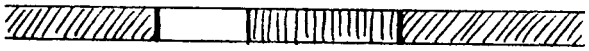
注) あき領域が1個のとき

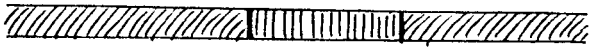


あき領域が0個のとき

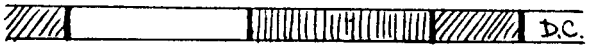


C. 領域の割当て (first fit)

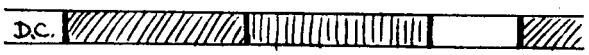
case 1 

case 2  あき領域をリストから削除

D. 領域の解放

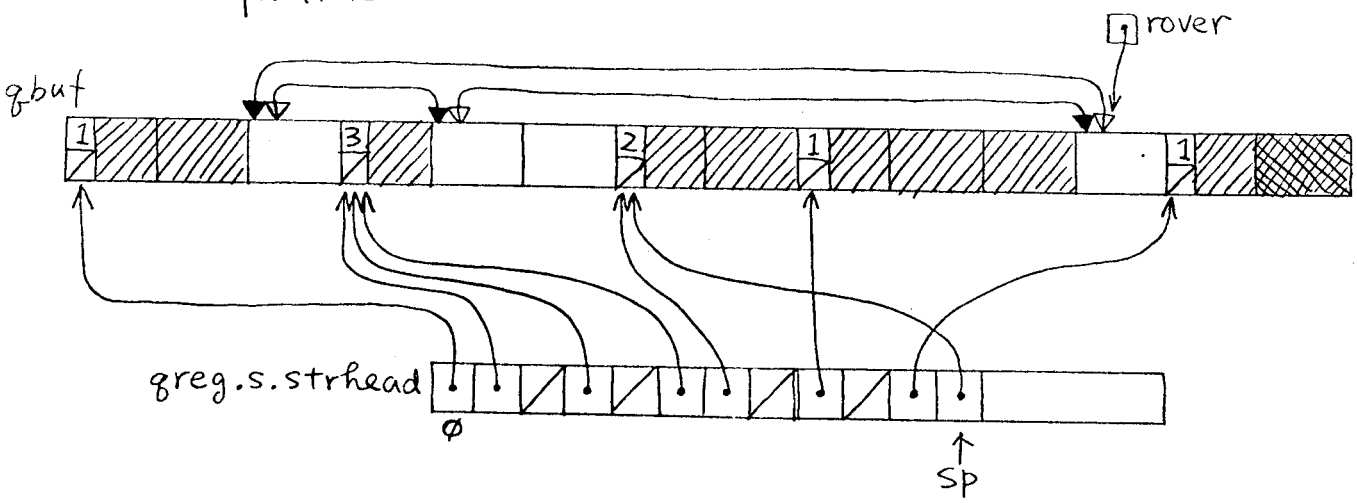
case 1  左のあき領域を拡大

case 2  右のあき領域を削除
全体をひとつのあき領域へ

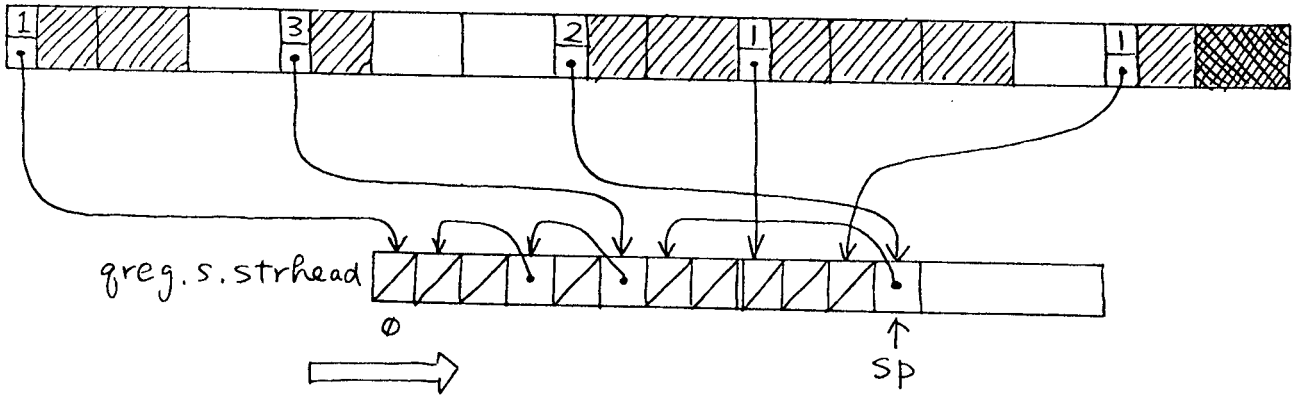
case 3  右のあき領域を拡大
ヘッダーの移動あり

case 4  あき領域の生成

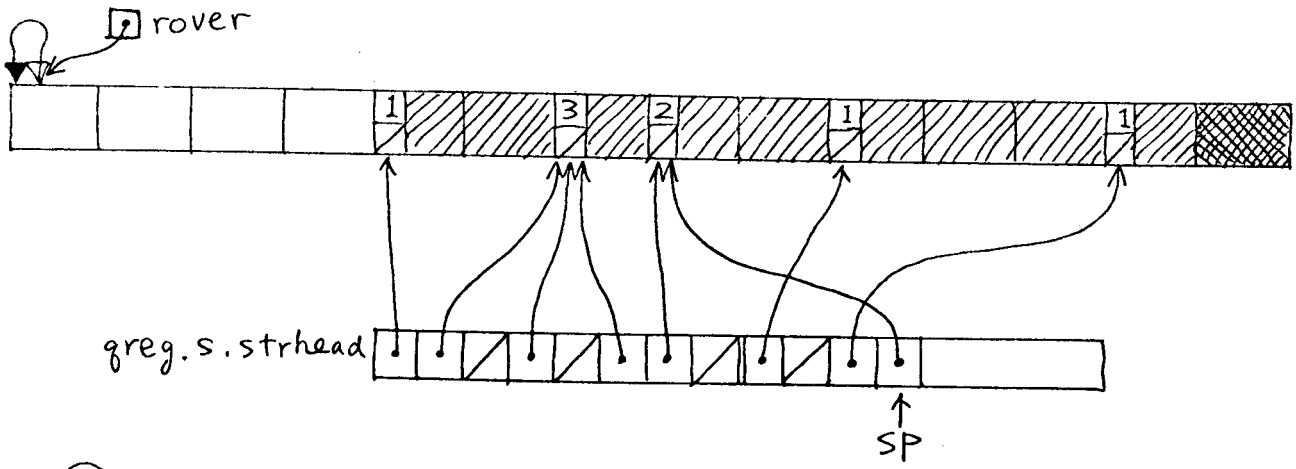
E. Compaction



PHASE 1 (ポインタ逆転)



PHASE 2 (pageの移動)



(注) cpos (program counter のようなもの) の更新は, 別に行なう。

① compaction を引起したコマンドの実行中

② 〃 以降に行なわれるマクロからの復帰の直後

・使いみち

edit のほかに，次の目的でしばしば使った。

① ファイルの中味を見る（見るだけ！）

② 一覧表（procedure 等の），統計（keyword 等の）をとる

・マクロの使われ方

① ほとんど使わない

② もっぱらミニエディタを使用

③ ミニエディタ＋ TECO

④ 個人用のマクロ群を常駐（例：表示を映像端末むけに変更）

筆者の場合，マクロは主に特殊操作用（一回限り）。汎用のものとしては，

a. 行単位入力（初期入力，行追加用）

b. 行番号づけ，行番号はがし　　c. indentation の変更

が有効だった。ただし b. は マクロにするほど複雑なものではない。

・便利だと思った点、

① 字エディタ方式，字間ポインタ方式である点（行中にポインタをおきっぱなしにできる）

② 複数のコマンドを 1 行に並べられる点

③ ふつうの文字と改行文字の扱いが統一的である点

④ 操作の対称性，たとえば 逆むきサーチ，行末への位置づけができる点

・不都合だと思った点、

① typewriter（あるいは glass typewriter）端末では，context addressing を基本とするエディタは 十分に威力を発揮できない。

例：　S/……/T　（サーチをしてその行を眺める）
　　2T　（その行だけでは identify できないので，前後も眺める）
　　4T　（それでもまだ identify できないので，更に広い範囲を眺める）

② 探索と置換 を分離して行なおうとすると不都合。

例：　Ss₁T で，変更すべき文字列を探し出し，確認した。さて次にどうするか？　Rs₁s₂ はダメ。　−Q/DIs₂ とするか，−Rs₁s₂ あるいは −RQ/s₂ としなければならない。

③ デリミタに関するエラーがやはり少なくない。最後のデリミタを忘れるエラーは減ったが，長い引用文字列の中で，デリミタの文字を使ってしまうエラーが増えた。