HITsにおけるWord文書の採点プログラム2019年度版の開発

長谷部　勝　也
松　井　吉　光
谷　口　正　明

要　　約：本学名古屋校舎で開講される一般教養科目「情報リテラシー・入門，応用」では本学で独自に開発したe-learningシステムであるHITsが使われている。本稿は2019年度からHITsのWord部門で使用する予定の文書解析・採点プログラムwt2019.pyについて述べる。

キーワード：e-Learning，教材，自動採点，Word®，Python

1　はじめに

大学教育において，また卒業後の就業時において，キーボード入力，文書作成および，表の計算は基本的なスキルとなってきている。その様な状況に対応するために本学名古屋校舎では2006年度から「情報リテラシー・入門，応用」を開講して来た。この講義は対面教育とe-learningを併用していて，e-learning部分で用いているのがHITs[1]である。HITsとはタイピング練習とMicrosoft社のWord®とExcel®の練習問題を対話的自動採点する，本学で独自に開発したe-learningシステムである[3]。

本稿では，HITsのWord®部門の解析・採点エンジンであるPython3で記述されたwt2019.pyの開発について述べる。今回の開発の出発点は，wt2016.py[6]における問題の解決であり，そのためのwt2019.pyの改良点とその仕様について述べる。

尚，現行のwt2018.py[7]もwt2016.pyの問題点に対処しているが採点対象と同一段落，且つ採点箇所の前方で採点課題と同じ修飾が誤ってなされると，正解を不正解と誤判定する副作用が生ずる。wt2019.pyではその問題も解決している。

2　wt2016.pyの問題点

Word®は文書をxml形式（樹構造）[1]で管理する。文字列（ここに青い小鳥がいます）をWord®を使って太字で書いたとしよう。この場合樹は一本の枝を持ちその枝には（ここに青い小鳥がいます）が格納され併せてこの文字列が太字であるという属性も格納される。

ここで（青い）に青色の活字を使うとどうなるか？この時枝は三本に増やされる。そうして最初の枝には文字列（ここに）と太字属性，中の枝には，（青い）と太字属性と青色属性，最後の枝には（小鳥がいます）と太字属性が格納される。この様にWord®は状況に応じて文字列を幾つかに分割し夫々に必要な

[1] Highly Interactive Training system

属性をつけた枝から成る樹を作る。

ここで改めて編集をやり直し（青い）についていた青色属性を解除したとする。即ち全文を太字で書いた最初のWord®文書に戻したとする。この時のWord®の動作は単に中の枝の青色属性を消すにとどまる。三本の枝が一本に纏まって最初の樹構造に戻れば単純だし，効率的だがWord®はその種の最適化動作をしない[2]。従って複雑な編集履歴を経たWord®文書は不必要な文字列分割を抱えている。Word®文書がどの様な文字列分割を持つかは予測不能なのである。この事をWord®の練習問題の作成と解析エンジンの動作の側から眺めてみよう。

1．受講生には問題文（ここに青い小鳥がいます）が与えられこの全文を太字にし，且つ部分文字列（青い）を青色にすることが指示される。

2．受講生は指示に従って問題文を編集し，解答文として返す。

3．解析エンジンは解答文を検査し正解か不正解かを判定する。

wt2016.pyの検査手順は以下の通りである。

1．文字列が（ここに, 青い, 小鳥がいます）に三分割されていることを想定する。

2．その各々の分割が太字属性を持てば太字問題を正解とする。

3．分割（青い）が青色属性を持てば青色問題を正解とする。

この手順の問題点は解答文に特定の分割を想定していることにある。恐らく模範的答案が来ればその想定が実現し正解と判定するだろう。しかし不完全な解答文に出会うと困ったことが起きる。例えば（青い）ではなく（青い小鳥）を青色にした解答文では想定した分割が得られないことが引き金となって太字の判定さえできず二問とも不正解とする。

3　wt2019.pyの改良点

不具合は特定の分割を想定することから発生する。そこで逆に解答文がどの様な分割を持つか調べることを出発点としよう。受講生は与えられた問題文の文字列の変更（文字列の追加或いは削除）をしてはいけないことになっている。それは採点箇所を正確に特定するためであってこれまでも度々言及して来た[6] [3]。しかし更に踏み込んで言えばこの時文字は文字番号で置き換えることができる。つまり文字列の分割を次の様に表現することが可能になる。

(1,13)(13,24)(24,36)...
　　太字問題のみ正解

最初の(1,13)は第1文字から始まる12文字（13文字ではない）を示す。即ち分割は二つの数字を並べそれを（,）で括って表現される。左の数字は分割の先頭の文字の番号，右の数字は左の数字にその分割の長さ（文字数）を加えたものである。だから次の分割（13,24）は第13文字から始まる11文字を示す。その11文字が（ここに青い小鳥がいます）であるとしよう。もしこの分割が太字属性を持っていればこれは太字について正解の解答文である。だがこの解答文は（青い）に対応する分

[2] 経験則

[3] footnote を付けよの様に極的に文字列の加筆を指示する練習問題も存在するが特殊例として今は無視する

割を持たないから（青い）を青色にしていない。そこで完全な正解文を作れば分割は次の様に変更されるだろう。

(1,13)(13,16)(16,18)(18,24)
(24,36)...
　完全正解

　対応関係は(13,16)=(ここに),(16,18)=(青い),(18,24)=(小鳥がいます)である。この時太字の採点は

1．分割の表現に（13,と,24）が現われる。

2．（13,と,24）で挟まれる全ての分割が太字属性を持つ。

を満たせば正解，そうでなければ不正解である。なお，上で言う「全ての分割」とは不完全正解については（13,24），完全正解については（13,16）（16,18）（18,24）である。

　青色の採点は（13,と,24）を（16,と,18）に置き換え，太字属性を青色属性に置き換えたものである。

　解答文の編集履歴によっては更に細かい分割も発生するだろうがそれでもこの手順は正しい判定を与える。

　尚，この分割の表現は文字列を持たない分割をも扱うことができる。例えば，(13,16)(16,16)(16,18)は（ここに）と（青い）の間に何かが，例えばtabが入り込んだ場合である。

4　制御ファイル

　解析エンジンへの入力は解答，模範解答，制御ファイルの三つである。解答は勿論受講生が返して来たWord®文書，模範解答はエンジン側が予め準備した完全正解のWord®文書，制御ファイルはどの様に採点するかをエンジンに指示するファイルである。

　HITsサーバには長年の間に受講生が返して来た多数の解答文が保存されている。それらの中から今回の改訂の動機となった問題番号W-0-03-0の最初の7個の解答文を選んで採点して見よう。W-0-03-0は22個の単一問題の集まりだがその前から3個だけを採点する。

Excersise03_0001.docx
4oo

Excersise03_0002.docx
4oo

Excersise03_0003.docx
4oo

Excersise03_0004.docx
4oo

Excersise03_0005.docx
ooo

Excersise03_0006.docx
ooo

Excersise03_0007.docx
ooo

docxで終わる第１行がファイル名で，第２行の4oo等が採点結果である。採点結果はoが正解，それ以外はエラーコードになっていて４は正解文には設定されている筈の属性が解答文に設定されていないことを示す。

　この問題への制御ファイル(の最初の三問)を以下に示す。

##制御ファイルw-0-03-0.ctl
##但し第四問以降を省略

```
¥XMLFILE=word/document.xml; ;

##第１問
##この問題は文字飾りの...
##活字を「ＭＳ明朝」に設定する
    10 rFonts
    10 1 97
    フォントの種類
    eastAsia=ＭＳ明朝
; ;

##第２問
##この問題は文字飾りの...
##活字の大きさを10ポイントに設定する
    10 sz
    10 1 97
    フォントサイズ
    val=20

; ;

##第３問
##これは太字
##「これは太字」の部分を太字に設定する
    10 b
    10 38 43
    太字
; ;
```

まず制御ファイルの動作を概括する。第１問は第10段落の第１文字に始まる96文字の活字（rFonts）をＭＳ明朝に設定することを指示し，第２問はその大きさをval=20即ち10ポイントに指定し，第３問は同じ段落の第38文字から始まる５文字（これは太字）を太字bにせよと言っている。前の章の太字中の色付け問題がここでは活字中の太字問題に対応する。制御ファイルの書式を以下に述べる。

1．#はコメント記号で自分を含めたその右は無視される。

2．; ; は終了記号である。その右は無視される。

3．メタ文は¥で始まる一行から成る。末尾に ; ; が必要である。

4．問題文は最低三行から成り，四行目以降はオプション但し採点対象である。

5．問題文の最後に ; ; が必要である。

6．制御ファイルはメタ文と問題文を並べたものである。

制御ファイルの最初に書かれたメタ文
¥XMLFILE=word/document.xml; ;
は採点対象を指定している。Word®文書は実は幾つかのxmlファイルをZIP形式で集約・圧縮したものであり，制御ファイルの最初でどのxmlファイルが対象となるかを指定しなければならない。尚，制御ファイルの途中で改めてメタ文¥XMLFILE=. . .xml; ;を書いて別のxmlファイルを指定することもできる。

第１問の最初の行

```
10 rFonts
```

は第10段落には（活字を指定する）rFonts要素が存在しなければならないと言っている。

第２行

```
10 1 97
```

は格納された文字列への言及であって第10段落の分割が（1,と,97）を持たねばならないと言っている。又この文字列は自動的に模範解答の対応する文字列と照合される。

第４行

eastAsia=ＭＳ明朝

はオプション（書式上必須ではない）であって第２行で確定した全ての分割が持つべき属性を指定する。纏めて言うと第１問は解答文の第10段落の第1文字から始まる96文字が「ＭＳ明朝」になっていれば正解，そうでなければ不正解と判断する。

オプションは複数個指定でき，全てが満たされる場合のみ正解とする。

第２問は第１問と同じ，但し活字の大きさをval=20と指定している。これは10ポイントを表す。

第３問はオプション無しで単に第10段落の第38文字から始まる5文字が要素bを持つことを調べる。要素bを持つ分割は太字で修飾される。第3行は単なるコメントで採点のロジックと無関係である。受講者に採点結果を報告する時に併せて表示して受講者の理解を助けるためのものである。

この節のこれ以降の部分はwt2016.py以前から実装された機能の再説明だが制御ファイルの書式としては新たなものである。

制御ファイルの第４行目の属性の指定には，正解の値に幅を持たせたり，他の問題の値との一致をみたりするための方法を用意してある。問題番号W-0-37-0の制御ファイルを例に取ろう。第２問目は

```
5 ind
5 0 0
左インデント
leftChars; 0 < leftChars
```

となっているが，４行目の意味は第５段落のleftCharsの属性の値を解答ファイルから読み取り，その値が０より大きい場合であれば正解とする，ということである。また，第４問目では

```
¥AND; ;
##〒４６４ - ８６０２
    5 ind
    5 0 0
    左インデント揃え
    leftChars; keep(leftChars,1)
; ;

##名古屋市東区筒井２丁目１０番
    6 ind
    6 0 0
    左インデント揃え
    leftChars; comp(leftChars,1)
; ;
¥END; ;
```

となっている。ここに登場するkeep( )とcomp( )は解析エンジンに準備された関数であってkeep(leftChars,1)は第５段落のleftCharsの値を１番目の内部変数の値として保持し，comp(leftChars,1)は第６段落のleftCharsの値が１番目の内部変数の値と一致するか否かを点検する。この二問は¥AND; ;と¥END; ; で括られているので第５段落のleftCharsと第６段落のleftCharsが共に存在し且つ同一の値である場合のみ正解，それ以外は不正解と判定される。

上の二例は共に属性値を問題にしているが表問題において表の数，列の数，セルの数を判定する関数count_table( )或いは採点対象範囲にある特定の要素の数を数える関数count_path( )が用意されている。例として問題番号W-0-30-0の制御ファイルを示す。

```
21 tab
21 0 0
タブ数（1）
count_path('tab',21)==1
; ;
```

この場合，第21段落に出現するtabの数を取得し，１個なら正解その他は不正解と判定する。

５　段落問題

別の制御ファイルの例を示す。

```
##制御ファイルw-0-22-0.ctl
##但し第１問のみ

¥XMLFILE=word/document.xml; ;

##第１問
    11 jc
    11 0 0
    中央揃え
    val=center
; ;
```

前の章のw-0-03-0.ctlとの違いは第２行の文字列指定が0 0となっていることでこの場合は分割を無視し段落全体の文字列に着目する。第１行の要素名jcは文字揃えでありオプションのval=centerは中央揃えを意味する。尚，今の場合もこの段落の全文が自動的に模範解答の対応する部分と照合され，不一致であればエラーコード１（書き換え禁止違反）の不正解と判定される。文字列照合をする必要が無い場合は第２行の11 0 0から段落指定を落として単に0 0とする。

この問題の様に分割を無視する問題を段落問題と呼ぶ。反対にこの章以前で論じてきた分割が意味を持つ問題をrun問題と呼ぶ。

先に述べた通りWord®文書は幾つかのxmlファイルの集合だがrun問題として扱うことを真に要求されるのはword/document.xmlだけである。そこでその他のxmlファイルについては段落問題に限定し解析エンジンのコードの簡明を図っている。

実例で言えばword/footnote.xmlは脚注を扱う。このxmlファイルに関しては指定された文字列例えば（これは芭蕉の俳句です）が正しく脚注として出現するか否かだけを問題とし脚注の一部を太字に修飾せよ等の問題を作らないということである。

６　複合問題

単一問題を幾つか並べて一つの複合問題と見做したいことがある。そのためにメタ文

```
¥AND; ;
¥OR; ;
¥END; ;
```

が用意されている。

```
¥AND; ;
  PROB_1; ;
  PROB_2; ;
  . . .
  PROB_n; ;
¥END; ;
```

はn個の単一問題が全て正解の場合に複合問題として正解，それ以外を不正解とする。又

```
¥OR; ;
  PROB_1; ;
  PROB_2; ;
  . . .
  PROB_n; ;
¥END; ;
```

は単一問題が全て不正解の場合に複合問題を不正解とし，それ以外を正解とする。

これらのメタ文は入れ子にしても良い。

```
¥OR; ;
¥AND; ;
  PROB_1; ;
```

```
    PROB_2; ;
  ¥END; ;
    PROB_3; ;
  ¥END; ;
```

は第１問，第２問が共に正解か第１問，第２問と無関係に第３問が正解なら正解，第１問，第２問の少なくとも一方が不正解，且つ第３問も不正解なら不正解を与える。

単一問題の正解/不正解を逆転して解釈したい場合がある。例えば文字列を太字で書いた問題文が与えられ，受講生は太字指定を解除するよう求められる場合等である。この時，解析エンジンは解答文の該当箇所に太字指定を発見しようとし発見すれば不正解，発見できなければ正解とする。この為のメタ文

```
  ¥NOT; ;
```

は例えば

```
  ¥AND; ;
    PROB_1; ;
    ¥NOT; ;
          PROB_2; ;
  ¥END; ;
```

の様に使う。この時の採点は第１問が正解，第２問が不正解の場合にのみ複合問題として正解，それ以外は不正解とする。尚，¥NOT; ; は一回限り有効なメタ文で¥END; ; 不要である。

コードを簡明にするために¥NOT; ; に続いて他のメタ文を書くことを禁止している。従って以下の三つの例は書式エラーである。

```
  ¥NOT; ;
  ¥NOT; ; ##NOT NOTの禁止

  ¥NOT; ;
    ¥AND; ; ##NOT ANDの禁止
          PROB_1; ;
          PROB_2; ;
    ¥END; ;

  ¥NOT; ;
    ¥OR; ; ##NOT ORの禁止
          PROB_1; ;
          PROB_2; ;
  ¥END; ;
```

即ちこれらは別の書式で表現しなければならない。ド・モルガンの定理というものがあって書き直し可能性は保証されているが，具体的に言えば第一の例NOT NOTは何もしないことだから無視して良く，第二の例NOT ANDは

```
  ¥OR; ;
    ¥NOT; ;
          PROB_1; ;
    ¥NOT; ;
          PROB_2; ;
  ¥END; ;
```

と書き換え，最後のNOT ORは

```
  ¥AND; ;
    ¥NOT; ;
          PROB_1; ;
    ¥NOT; ;
          PROB_2; ;
  ¥END; ;
```

とする。

7　その他のメタ文

その他のメタ文としては

```
¥STEM=. . . ; ;
¥STEX=. . . ; ;
¥STRIP; ;
```

がある。

例えば，w-0-03-0.ctlの第一問

```
##第１問
##この問題は文字飾りの. . .
##活字を「ＭＳ明朝」に設定する
    10 rFonts
    10 1 97
    フォントの種類
    eastAsia=ＭＳ明朝
; ;
```

では，10段落を採点対象としているが詳細情報としてコントロールファイルの最初に

```
¥STEM=/w:document/w:body/w:p[10]; ;
¥STEX=/w:document/w:body/w:p[10]; ;
```

と書いておく。解析エンジンは¥STEM以下にrFontsパスがあるかをチェックするとともに¥STEX以下に存在する（1,97）の文字列を抽出し，模範解答と照合する。この例ではパス照合の起点¥STEMとテキスト照合の起点¥STEXは同一だが一般的には異なる可能性が排除できずその対応として二つのメタ文が存在する。

メタ文¥STRIPは，段落の文字列の変更をチェックする際，解答文のテキストの前後にある空白文字を除いてから模範解答のテキストと照合することを指示する場合に用いる。これは脚注や文末脚注を扱う場合，解答者の意図とは関係なく，文字列の前後に空白が入ったり，入らなかったりすることへの例外的措置である。

８　wt2019.pyへの移行スケジュール

2018年８月の時点において，それ以前に受講生が返して来た解答文すべて（それらはHITsサーバに蓄積されている）をwt2019.pyで改めて採点し，問題なく動作することを確認した。今後は，HITsの予備システムのWord®採点エンジンをwt2019.pyに変更して，2019年２月までの間に動作確認をし，大きな不具合などが見つからなければ，2019年度から実際のHITsシステムでwt2019.pyを運用する予定である。

９　まとめ

本稿では，HITsシステムの一部であるWord®ファイルの自動採点プログラムの2019年度版wt2019.pyについて述べた。ここに見る様に自動採点プログラムは毎年改良を加え，採点の精度の向上に努めている。

しかし課題は実は別の方向にも存在する。それは誰にでも問題を作成できるための支援システムの実現である。

Word®文（問題文）を一つ案出し、それをしかるべく修飾せよと指示し（問題の記述）指示通りのWord®文（正解文）を作ることに困難はないだろう。

困難は対応する制御ファイルの作成である。そのためにはWord®のXMLに精通し，同時にWord®の操作によるXMLの変化にも精通していることが要求されるし、事実これまでの制御ファイルはXMLの詳細な知識を元に作成されて来た。

しかしwt2019.pyの開発では並行的に次の可能性、即ち問題文と正解文夫々のXMLファイルの差を抽出して自動的に制御ファイルを作成することを検討して来た。今後は，採点精度の向上は当然として，制御ファイルの自動作成にも取り組んでいく予定である。

参考文献

[1] Standard ECMA-376 Office Open XML File Formats, http://www.ecma-international.org/at/publications/standards/Ecma-376.htm
[2] XML Path Languageバージョン1.0 W3C 勧告，http://www.w3.org/TR/1999/REC-xpath-19991116
[3]“Excel, Word自動採点システムHITsの構築と運用”岩田員典，他，
愛知大学情報メディアセンター（20）No.1（2010）
[4]“HITsにおけるWord文書の採点プログラムの開発”
長谷部勝也，松井吉光，谷口正明
愛知大学一般教育論集（40）25-40（2011）
[5]“HITsにおけるWord文書の採点プログラム2013年度版の開発”
長谷部勝也，松井吉光，谷口正明
愛知大学一般教育論集，（45）41-53（2013）
[6]“HITsにおけるWord文書の採点プログラム2016年度版の開発”
松井吉光，長谷部勝也，谷口正明
愛知大学一般教育論集，（52）27-36（2017）
[7]“HITsにおけるWord文書の採点プログラム2018年度版の開発”
松井吉光，谷口正明
愛知大学一般教育論集，（54）43-49（2018）