

コーパス検索と UNIX コマンド

稲 木 昭 子

1. はじめに

コーパスを検索する際に検索語をどのように設定するかは、続く分析の要になるもので、非常に重要な仕事である。ところがこの設定はそれほど単純なことではなく、当該の語に付けられているタグを利用したり、先行する語、あるいは後続する語に制限をつけたりと、個別の検討が必要である。ここでは COBUILDDirect (以下 Cobuild と略記する) というインターネット・サービスを通じて、The Bank of English という大規模コーパスを検索するときに考慮すべき点を具体例で考えることにし、さらに、単純に検索できないときのデータ収集には UNIX コマンドが有効であることを示すことにする。

2. 検索語設定

「義務」および「論理的必然性」を表す法助動詞をコーパス検索による方法で分析することになると、まず検索語をどのように設定するかが問題となる。これらの意味を表す法助動詞の代表には、法助動詞 (modal) に加えて、半法助動詞 (semi-modal) が含まれる。前者には① must、後者には② have to ③ have got to ④ got to ⑤ gotta の4種類の型がある。これらはそれぞれ単純に文字通りに検索できるわけではない。例えば must を文字通りに検索すると、名詞用法の must も抽出したり、must に否定語がついた must not の「禁止」を表す場合等も抽出し、検索結果の全てが検索者の意図した「義務」および「論理的必然性」の must の用法とはならない。そこで、「義務」および「論理的必然性」を表す語 (句) の検索を具体的にどう設定すればよいか、①～⑤の順に考えていくことにする。

2.1. must の場合

must という単語には前述したように名詞用法もあるので、検索語を単純に “must” とすると、この名詞の場合も含まれてしまうことになる。そこで名詞の場合を排除するために、品詞タグを利用して、検索語を “must/MD” とする (MD: modal verb)。 (ただしこのコーパスのタグ付けは必ずしも正確に行われていないこともあるので注意を要する。) 次はタグが行頭に表出されているコンコーダンスの一部である。

MD row donors are found. Body tissue must closely match that of each victim.
 MD t day — and for that little kid it must have been like he'd walked a hundred
 MD humbling for us. Some good must come out of it — we will save lives.
 NN said otherwise, the city became a must for all serious pilgrims. <p>

法助動詞には動詞の原形が後続するということで、“must+VB|BE|DO|HV” (VB: verb base form, BE: verb ‘to be’ base form, DO: verb ‘to do’ base form, HV: verb ‘to have’ base form) という設定も考えられるが、この設定では must に副詞類、あるいは接続詞、あるいは会話体にみられる休止等が後続する場合は排除されてしまうことになるので、より正確な検索ということでは的確でない。次のコンコーダンスは動詞以外のものが後続している場合の例である。

MD But I simply must have some, I must!” <p> Some of you made the
 MD But i not telling them that You must <tc text = knocks table> do this <M01>
 MD ory about Nye Bevan? <M01> Yes you must. <tc text = laughs> <F01>
 MD ulations stipulated in its charter must <tc text = pause> insure to its best
 MD rried her to manage and manage she must”. <tc text = pause> That is a reality
 MD All right then MX well listen we must <tc text = pause> Oh is that what he's

そこで法助動詞の must の検索語は “must/MD” と設定される。次の①は subcorpus 別の頻度数の結果である。

- ① Query is “must/MD”
 26482 matching lines

Corpus	Total Number of Occurrences	Average Number per Million Words
usephem	828	676.1/million
ukephem	2055	657.7/million
ukbooks	3290	614.5/million
sunnow	3336	572.8/million
usbooks	3112	553.1/million
today	2751	524.2/million
times	2521	437.4/million
ukmags	1909	389.4/million
bbc	980	375.5/million
ukspok	3057	329.7/million
npr	993	317.3/million
oznews	1650	309.1/million

ところが、この結果には must not の「禁止」の用法も含まれてしまうので、「義務」等の意味で使用される must のみを検索するためには、must not の連語ででてくる must を排除しなければならない。この must not を検索するための検索語は “must+not” となり、その検索結果は次の②となる。

- ② Query is “must+not”
 700 matching lines

Corpus	Total Number of Occurrences	Average Number per Million Words
--------	-----------------------------	----------------------------------

ukephem	84	26.9/million
ukbooks	94	17.6/million
today	92	17.5/million
usbooks	88	15.6/million
sunnow	89	15.3/million
bbc	39	14.9/million
times	68	11.8/million
ukmags	48	9.8/million
oznews	46	8.6/million
usephem	8	6.5/million
npr	17	5.4/million
ukspok	27	2.9/million

①から②を引いたものが、検索したい「義務」あるいは「論理的必然性」を意味する `must` の結果となる。このコンコーダンスを得るには、①の設定語で検索した後、ソート（並べ替え）して、そのコンコーダンスから `must not` を削除していく方法もあるが、これは当該の行数が大きい場合には煩雑になる。そこでこの点で UNIX コマンドが役に立つ。Unix を使用する場合にはまず①の“`must/MD`”のデータを `ftp` でダウンロードし、このファイル名を `amust.txt` とする。次に②の“`must+not`”のデータを `ftp` でダウンロードし、これを `mustnot.txt` とする。次に UNIX を操作し、次のようにコマンドを打つ。

- (1) `shogun01% sort amust.txt > samust.txt`

`amust.txt` をソートしたものを、`samust.txt` とする。(sort : 指定されたファイルの全ての行をソートし、標準出力に書き出す)

- (2) `shogun01% sort mustnot.txt > smustnot.txt`

`mustnot.txt` をソートしたものを、`smustnot.txt` とする。

- (3) `shogun01% comm -3 samust.txt smustnot.txt | wc -l`

25782

`samust.txt` と `smustnot.txt` を比較して、共通でない行を出力し、その行数をカウントすると、25782となる。(comm : ソート済みの2つのファイル、`file 1` と `file 2` を比較して、片方だけある行と両方にある行に分類する。-3 : 3列目を出力しない、つまり違いの行を出力する。wc -l : 行数を表示する。) つまり、ここでは①の頻度数から②を引いた残りの頻度数がでてきていることになる。

- (4) `shogun01% comm -3 samust.txt smustnot.txt > must.txt`

`samust.txt` と `smustnot.txt` に共通する行を削除した残りのものを、`must.txt` とする。

この `must.txt` のコンコーダンスの一部を表出するものが次のコマンドである。

(5) shogun01% comm -3 samust.txt smustnot.txt | head -10

```

                                It must be less. It must be less than. Yes.
    <M01> Oh really. I must try that one. <FOX> Oh yeah I'll
                                It must be less. It must be less than. Yes. I mean <ZF1> that'
    <FOX> I think Tuesday must be the worst day is it FX? <FOX> It'
    <M02> I imagine there must be some still extant. <M01> Well I
    Right. I see <M02> He must have known <ZGY> come back again
    That reminds me there must be a new Viz out surely. <FOX> Mm.
    Yeah. Exactly yeah. I must remember to bring the cutters round
    <FOX> but I think they must have been ups up in the loft
    <M01> Yes. Mm. Now you must have known who were the big

```

Unix のコマンドに “uniq” (ファイル中の重複行の出力) があるが、これをたとえば次のように使用することもできる。

(6) Shogun01%cat amust.txt. mustnot.txt | sort | uniq -u > tmust.txt

cat amust.txt mustnot.txt で2つのファイルを連結し、その結果をパイプによって sort コマンドに渡す。sort で並べ替えを行う。同じ行が同じ場所に生じることになる。その結果を uniq -u に渡す。uniq -u で重複していない行を抽出する。結果として得られる tmust.txt は、amust.txt から mustnot.txt を差し引いた結果となる。

但しこのコマンドでは、仮にファイル中に3回重複していた行があったとすると、全て削除してしまうことになる。ところが Cobuild ではたとえば3回重複して検索された場合も頻度数はあくまで3回とカウントされるので、uniq コマンドでは Cobuild のいう頻度数が得られないことになる。ちなみに、この場合の行数をカウントすると、次のようになる。

(7) shogun01%wc -l tmust.txt

shogun01%25039

従って、Cobuild で得られた2つのファイルのうち file 1 から file 2 を引き算するときには、共通しているものを削除する comm が有効なコマンドとなる。

①および②から must の表を作成すると次のようになる。

25782 matching lines		
Corpus	Total Number of Occurrences	Average Number per Million Words
usephem	820	669.5/million
ukephem	1971	630.9/million
ukbooks	3196	596.9/million
sunnow	3247	557.5/million
usbooks	3024	537.5/million
today	2659	506.6/million
times	2453	425.6/million
ukmags	1861	379.6/million

bbc	941	360.6/million
ukspok	3030	326.8/million
npr	976	311.9/million
oznews	1604	300.5/million

2.2. have to の場合

to に後続する語は動詞の原形であるので、“to+VB|BE|DO|HV” (VB: verb base form, BE: verb ‘to be’ base form, DO: verb ‘to do’ base form, HV: verb ‘to have’ base form) と指定することも可能であるが、この指定をせずに検索してみると、動詞以外には、次のコンコードダンスに表出される副詞や、挿入節がみられる。ところがこれらの後には動詞が後続しているので、結局は構文的に同じものを検索していることになる。従って、検索語の幅を狭めて指定する必要もなく、to に後続する語の指定は不要となる。

a role in this as well. <p> And we have to, I think, at least put more what — I mean, I carve some because I have to. That’s my job. That’s my work. meet the new standards. Others will have to. <p> Skinner: Beginning in 1993, I heard. <p> Siegel: That there would have to then be some kind of political and does that mean the Iraqis will have to simply walk out of Kuwait? If so any Iraqi withdrawal is likely to have to actually occur through the allied s no freedom of the press, they will have to — you know — if it doesn’t follow or traditional methods and we’ll have to actually — literally turn to

次に to に先行する語としては、have, has, had, having の 4 通りに加えてこれらの省略形の、“’ve, ’s, ’d” が考えられる。Cobuild では、“’ve, ’s, ’d” 等のアポストロフィ付の検索はできないので、“ve, s, d” で検索することになる。ところが“s” で検索すると、以下の用例に見られる通り、be to 型の三人称の場合、to 不定詞の副詞的用法、it...to... 形式なども表出されることになるので、s が has の省略形の場合のみを検索するために、“s/HVZ” (HVZ: verb ‘to have’ 3rd person pres sing: *has*) と設定制限することが必要になる。

To be in this Soviet proposal. There’s to be no guarantee of Saddam Hussein’ in most cases officials don’t know who’s to blame. <p> At one of the many similar to his friend, Henry Miller’s, to return literature to life. s responsibility, not state government’s, to see that there is a safe and <p> Renee Hall (Tenant Yeah, I feel it’s to be a ploy, you know, to get the

ve の場合はタグ付けは不要であるが、d の場合はまた問題が生じる。検索語を、“d+to” とすると、90例抽出されるが、このコンコードダンスの中で、検索したい例文の d のタグは MD と付けられていることが分かる。したがって d の場合は“d/MD” と設定することが必要となる。

NP rds, unlike mammals, require Vitamin D to be in their diet as Vitamin D3.
NP books, but who ruled from 1000 A.D. to 1800 as governors, sultanas, and
NP ave never looked back.” <hl> No knee -d to panic Football; Match report </hl>
MD e me over as soon as possible, but I’d to point out we have one or two big
MD My eye filled up with blood and I’d to go to a specialist every day for a

以上をまとめると、have to の場合の検索語は、

`"have|has|had|having|ve|s/HVZ|d/MD+to"` と設定されることになる。

Query is "have|has|had|having|ve|s/HVZ|d/MD+to"

51717 matching lines

Corpus	Total Number of Occurrences	Average Number per Million Words
ukspok	13126	1415.6/million
sunnow	6283	1078.7/million
ukbooks	5103	953.1/million
npr	2959	945.6/million
today	4789	912.5/million
usbooks	4308	765.7/million
oznews	3911	732.7/million
times	4195	727.8/million
ukmags	3522	718.5/million
bbc	1663	637.2/million
ukephem	1466	469.2/million
usephem	392	320.1/million

2.3. have got to の場合

have got to の場合、have to と同様に `"have|has|had|ve|s/HVZ|d/MD+got+to"` と設定して検索すると、to に続いて、a, the 等の冠詞、this 等の指示形容詞、固有名詞等が後続する場合が含まれていることが分かる。

reporting: <p> The allied ships got to the port, chiefly on their radar convulsions and everything when they got to us. A journalist brought them a long waiting list. They have to got to a welfare office rather than a to Saddam Hussein. By the time he got to the northern Iraqi city of

そこで to に後続するものを制限する必要がでてくることになる。また having got to の形も予測されるが、検索してみると次のコンコーダンスに見られるように別の構文であるため、検索語の中からは削除している。

not to have pressed on, having got to 11 under. But nobody has that was sort of in the middle of having got to grips with the core subjects

そこで、have got to は `"have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB"`

(RB: adverb) と設定される。

Query is "have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB"

4857 matching lines

Corpus	Total Number of Occurrences	Average Number per Million Words
ukspok	2939	317.0/million
npr	211	67.4/million
sunnow	349	59.9/million

today	314	59.8/million
ukbooks	233	43.5/million
usbooks	212	37.7/million
ukmags	182	37.1/million
oznews	183	34.3/million
times	147	25.5/million
bbc	62	23.8/million
ukephem	22	7.0/million
usephem	3	2.4/million

2.4. got to の場合

got to を直接に検索する方法はない。というのも、“got+to”で検索すると、have 等が先行する場合も含まれることになる。そこで 2.1. で見た must の場合と同じ方法をとることになる。

“got+to+VB|BE|DO|HV|RB”を検索する。

- ① Query is “got+to+VB|BE|DO|HV|RB”
7526 matching lines

Corpus	Total Number of Occurrences	Average Number per Million Words
ukspok	4264	459.9/million
npr	347	110.9/million
usbooks	489	86.9/million
sunnow	503	86.4/million
today	421	80.2/million
ukbooks	361	67.4/million
ukmags	279	56.9/million
oznews	260	48.7/million
times	195	33.8/million
bbc	87	33.3/million
ukephem	42	13.4/million
usephem	8	6.5/million

“have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB”を検索する。

- ② Query is “have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB”
4857 matching lines

Corpus	Total Number of Occurrences	Average Number per Million Words
ukspok	2939	317.0/million
npr	211	67.4/million
sunnow	349	59.9/million
today	314	59.8/million
ukbooks	233	43.5/million
usbooks	212	37.7/million
ukmags	182	37.1/million
oznews	183	34.3/million
times	147	25.5/million
bbc	62	23.8/million
ukephem	22	7.0/million
usephem	3	2.4/million

①, ②から got to の各レジスター別頻度を表にすると、次のようになる。

2399 matching lines		
Corpus	Total Number of Occurrences	Average Number per Million Words
ukspok	1325	142.9/million
usbooks	277	49.2/million
npr	136	43.5/million
sunnow	154	26.4/million
ukbooks	128	23.9/million
today	107	20.4/million
ukmags	97	19.8/million
oznews	77	14.4/million
bbc	25	9.6/million
times	48	8.3/million
ukephem	20	6.4/million
usephem	5	4.1/million

この got to のコンコーダンスを得るためには、前述の must の場合と同様 UNIX コマンドが有効である。①の “got+to+VB|BE|DO|HV|RB” のデータのファイル名を agotto.txt、②の “have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB” のデータのファイル名を ahgotto.txt とする。UNIX を操作し、次のようにコマンドを打つことによってコンコーダンスは得られる。

- (1) shogun01% sort agotto.txt > sagotto.txt
- (2) shogun01% sort ahgotto.txt > sahgotto.txt
- (3) shogun01% comm -3 sagotto.txt sahgotto.txt | wc -l
- (4) shogun01% comm -3 sagotto.txt sahgotto.txt > gotto.txt

2.5. gotta の場合

gotta の場合は、他の品詞との混同は起きないので、検索語は、“gotta” と設定される。

Query is “gotta” 480 matching lines		
Corpus	Total Number of Occurrences	Average Number per Million Words
usbooks	234	41.6/million
ukspok	124	13.4/million
ukmags	44	9.0/million
npr	13	4.2/million
sunnow	18	3.1/million
today	13	2.5/million
usephem	3	2.4/million
oznews	10	1.9/million
ukbooks	10	1.9/million
times	8	1.4/million
ukephem	3	1.0/million
bbc	0	0.0/million

3. おわりに

検索語の設定は一見簡単に思われるが、当該の語（句）がどのような形式で 사용되는か、またどのような種類の語が検索語に先行し、さらに後続するか等を考慮に入れて、一つずつ検討する必要がある。「義務」および「論理的必然性」を表す法助動詞と半法助動詞の検索の設定は次のようにまとめられる。

must の場合：“must/MD” から “must+not” を引き算することによりデータが得られる。

have to の検索語：“have|has|had|having|ve|s/HVZ|d/MD+to”

have got to の検索語：“have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB”

got to の場合：“got+to+VB|BE|DO|HV|RB” から “have|has|had|ve|s/HVZ|d/MD+got+to+VB|BE|DO|HV|RB” を引き算することにより、データが得られる。

gotta の検索語：“gotta”

上記のうち2項目は直接に検索が不可能であるので、このときに UNIX コマンドを使用することにより、データを容易に抽出することができる。

参考文献

坂本 文. 2000. 『たのしい UNIX — UNIX への招待』. アスキー出版局.

滝沢直宏. 2000.10.14. 「コーパス利用のための UNIX の最初歩」. 英語コーパス学会・第16回大会・ワークショップ.

舟本 奨. 1997. 『UNIX ハンドブック 改訂新版』. ナツメ社.