自然言語処理入門

岸山 健 (31-187002)

Dec. 3, 2018

課題

以下の文法を用いて，「博士が病院でもらった薬を飲んだ」を CYK 法で解析せよ．

まず必要となるのは「病院で」が属する曖昧性を反映できるかが気になるポイント．外部パッケージには rbind を配列にも拡張した abind のみを利用する．

```
install.packages("abind")
library(abind)
```

とりあえず文法を定義する．また，->で左辺と右辺を分割できる関数 ‘も定義しておく． R の c 関数は"concatenate"を意味し， 自身が含むオブジェクトが全て同じ型に属すことを保証できる． したがって，lapply‘関数を使うときも，“S -> PP VP” などの char 型を list に変換する関数が cfg.phrase がもつ char 型オブジェクトそれぞれに 適用できる．

```
cfg.phrase <- c("S -> PP VP",
                "PP -> NP P",
                "NP -> VP NP",
                "VP -> PP VP")
cfg.lexicon<- c("NP -> 薬",
                "NP -> 病院",
                "NP -> ヒロシ",
                "VP -> 飲んだ",
                "VP -> もらった",
                "P -> が",
                "P -> で",
                "P -> を")
cfg <- function(s) unlist(strsplit(s, " -> "))
lapply(cfg.phrase,cfg)
## [[1]]
## [1] "S"     "PP VP"
##
## [[2]]
## [1] "PP"   "NP P"
```

```
##
## [[3]]
## [1] "NP"    "VP NP"
##
## [[4]]
## [1] "VP"    "PP VP"
```

ボトムアップの解析であるため，‘太郎’ に対して ‘NP’ を返したり，’PP’ と’VP’ を見たら’S’ を返してくれる関数が欲しい．残念ながら R にはハッシュテーブルが用意されていない．したがって，以下の手順を踏み “PP VP” のような右辺をキーに “S” を取れるように工夫する．まずは先程の `fvg.phrase` を `data.frame` 型に変えて `t` 関数で転置する．転置すると型が `matrix` になってしまうので再度 `as.data.frame` で型を戻し，最後に列に `LHS` と `RHS` の名前をつける．

```
 <- lapply(cfg.phrase, cfg)
vector2table <- function(l) {
    table <- as.data.frame(t(as.data.frame(l, stringsAsFactors=FALSE)),
                     stringsAsFactors=FALSE)
    colnames(table) <-  c("LHS","RHS")
    return(table) }
p.table <- vector2table(p.vector)
p.table
##                   LHS   RHS
## c..S....PP.VP..    S PP VP
## c..PP....NP.P..   PP  NP P
## c..NP....VP.NP..  NP VP NP
## c..VP....PP.VP..  VP PP VP
```

上のテーブルにおいて `RHS` 列が “PP VP” の行を抽出すると char 型の atomic vector が得られる．

```
rhs2lhs <- function(table) function(rhs){
    unlist(subset(table, table$RHS==rhs)$LHS)}
rhs2lhs(p.table)("PP VP")
# [1] "S"  "VP"
```

以上の操作を lexicon でも行なう

```
l.vector <- lapply(cfg.lexicon, cfg)
l.table <- vector2table(l.vector)
rhs2lhs(l.table)("ヒロシ")
```

CKY 法は三角行列の a_{ij} をベースに構文解析を進めていくため，最後の準備として三角行列 (対角線で区切られている行列) を用意する．行列の i も j も長さは入力文字列がもつ前終端器号数となる．スペース区切りで形態素解析が終わっている文字列が与えられるとする．それを char 型のベクトルとして今後扱う．

```
s2v <- function(s) unlist(strsplit(s, "[ ]"))
input <- "ヒロシ が 病院 で もらった 薬 を 飲んだ"
v.i <- s2v(input)
## [1] "ヒロシ" "が" "病院"   "で"  "もらった" "薬"    "を"
```

```
## [8] "飲んだ"
length(v.i)
# [1] 8
```

ただ，行列内に更にベクトルが生まれることを考えると，行列を多次元に拡張した配列の使用が好ましい．まずは rhs2lhs を終端器号 (単語) に apply して前終端器号の list，v2l とする．つまり，“ヒロシ” を rhs2lhs に apply すると NP となる．これを行列に組み込む操作を一般化すると，文字列の i 番目の要素を行列の i 行目，i 列目に挿入，という操作になる．

```
v2l <- lapply(v.i, rhs2lhs(l.table))
# 行列の中がベクトルとなりうるので配列を使う.
triangle <- array(dim = c(len, len, 1))
mapply((function(x,i) triangle[i,i,1] <<- x),
       v2l, 1:(length(v2l)))
triangle
```

```
## , , 1
##
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,] "NP" NA   NA   NA   NA   NA   NA   NA
## [2,] NA   "P"  NA   NA   NA   NA   NA   NA
## [3,] NA   NA   "NP" NA   NA   NA   NA   NA
## [4,] NA   NA   NA   "P"  NA   NA   NA   NA
## [5,] NA   NA   NA   NA   "VP" NA   NA   NA
## [6,] NA   NA   NA   NA   NA   "NP" NA   NA
## [7,] NA   NA   NA   NA   NA   NA   "P"  NA
## [8,] NA   NA   NA   NA   NA   NA   NA   "VP"
```

```
triangle[1,1,]
## [1] "NP"
```

ミソは単純な i と j による 参照ができない，という点．とりあえず，土台を d と置いて話を進める．n-d では上の階層を見ている．

```
triangle <- array(dim = c(len, len, 1))
mapply((function(x,i) triangle[i,i,1] <<- x),
       v2l, 1:(length(v2l)))
triangle
```

```
c(outer(rhs1, rhs2, FUN=paste))
```

```
triangle
```

```
n <- 8
for(d in 1:(n-1)){
    for (i in 1:(n-d)){
```

```
        j <- i + d
        for (k in i:(j-1)){
            ik   <- triangle[ i , k, ]
            k1.j <- triangle[k+1, j, ]
            product <- c(outer(ik, k1.j, FUN=paste))
            m <- unlist(sapply(product, rhs2lhs(p.table)))
            if(length(m)==1){
                triangle[i,j,1] <- m
            }else if(length(m)>1){
                len.m <- length(m)
                tmp <- array(dim = c(n, n, len.m))
                tmp[i,j,1:len.m] <- m
                triangle <- abind(triangle,tmp)
            }
        }
    }
}

cat.table <- array(dim = c(len, len))
for(i in 1:8){
    for(j in 1:8){
    cat.table[i,j]<- toString(na.omit(triangle[i,j,]))
    }
}
cat.table
```

一番左上のレベルでは “S” が 3 つ作成されているため，3 つの曖昧性があると分かる． まず 「ヒロシが病院でもらった薬を (PP)」「飲んだ (VP)」 がある． 「病院で」という PP と 「もらったという」VP がマージして「病院でもらった」という VP となり，さらに「ヒロシが」という PP と 「病院でもらった」という VP がマージして「ヒロシが病院でもらった」という VP が作られる．この VP と「薬」がマージし NP，そしてさらに 「を」とマージして PP となる．これは「ヒロシが病院でもらってきた薬」を誰かは知らないけど誰かが飲んだ，という意味になる．

こちらは曖昧性が生じていないようである． しかし 「ヒロシが (PP)」「病院でもらった薬を飲んだ (VP)」のほうは VP が 2 つできてる．VP に統合できる右辺は PP と VP であることを念頭に 探すと，「病院でもらった薬を (3–7)」「飲んだ (8)」 と「病院で (3–4)」「もらった薬を飲んだ (5–8)」 のパターンがある．