

## DCG\_01

The prize of the barrel of Brent oil drops down to 12.17\$

### 1) CFG

S --> NP VP  
NP --> det noun  
NP --> propernoun  
NP --> quantity unit  
NP --> NP PP  
NP --> NP PP noun  
PP-> Preposition NP  
VP --> verb NP

det -> the  
preposition -> of | to  
noun -> prize | barrel | oil  
propernoun -> Brent  
verb --> drops down  
quantity --> 12.17  
unit --> \$

### 2)DCG

S([head:[number:Y]]) --> NP([head:[number:Y]]) VP([head:[number:Y]])  
NP([head:[number:Y]]) --> det noun([number:Y])  
NP([head:[number:Y]]) --> propernoun([number:Y])  
NP([head:[number:Y]]) --> quantity unit  
NP([head:[number:Y]]) --> NP([head:[number:Y]]) PP  
NP([head:[number:Y]]) --> NP([head:[number:Y]]) PP noun([number:Y])  
PP-> Preposition NP([head:[number:Y]])  
VP([head:[number:Y]]) --> verb([number:Y]) NP

#### Terminal entries

terminal(det, the) .  
terminal(preposition, of).  
terminal(preposition, to).  
terminal(noun ([number:singular]),prize).  
terminal(noun([number:singular]), barrel).  
terminal(noum([number:singular], oil).  
terminal(propernoun([number:singular]),Brent).  
terminal(verb([number:singular]), drops\_down).  
terminal(unit, \$).  
terminal(quantity,N):- quantity(N).

---

## DCG\_02

Several hundreds  
Less than one thousand  
About 15,000

Around five hundreds

Two or three dozens

327

Between two and three hundred

1. Propose a representation formalism for quantities like the ones in these examples.

(type of value, value)

(exact\_quantity, quantity) // (exact\_quantity, 327)

(modified\_value,(modifier, quantity)) // (modified\_value,(several, hundreds)),

(modified\_value,(less-than, 1000)),(modified\_value,(about, 15,000))

(modified\_value,(around, 500))

(values\_related, (relation,value1,value2)) // (values\_related,(or,24 ,36)),

(values\_related,(between,200,300))

2. DCG

numerical(sem:(exact\_quantity,N)):- number(N).

numerical(sem:[Semantic,N]):- modifier(Semantic), number(N).

numerical(sem:[between,[Val1\*Val,Val2\*Val]]):-

between, digit(Val1),and,digit(Val2), textexpression(Val).

numerical(sem:[or,[Val1\*Val,Val2\*Val]]):-

digit(Val1),or,digit(Val2), textexpression(Val).

number(N):- integer(N).

number(Val1\*Val):- digit(Val1), textexpression(Val).

number(N):- textexpression(N).

terminal(digit(1),one).

terminal(digit(2),two).

terminal(digit(3),three).

terminal(modifier(around),around).

terminal(modified(about),about).

terminal(modified(several), about).

terminal(textexpression(12),dozen).

terminal(textexpression(12),dozens).

terminal(textexpression(100),hundred).

terminal(textexpression(100),hundreds).

terminal(textexpression(1000),thousand).

terminal(textexpression(1000),thousands).

-----  
DCG\_03

11th September 2006

The day before yesterday

Last Saturday

Christmas day

In Winter

In middle of July

1. Propose a representation formalism for dates and time intervals like the ones in these examples.

(type\_date, date)  
(longdate, (day,month,year))  
(spetial\_day, name-day)  
(weekday, weekday)  
(modified spetialday, (modifierdate, name-day)) //last Christmas  
(modified weekday, (modifierdate, weekday)) // last Friday  
(intervals, season)  
(intervalm, month)  
(modified, season)  
(modified, month) | (middle, July)

2) Using the DCG formalism write a simple grammar for detecting in a sentence temporal expressions like these and representing them according to the representation system proposed in 1).

date -> longdate  
date -> relativeday  
date -> textdate  
date -> modifierdate textdate  
date -> in interval  
longdate -> day month year  
textdate -> weekday  
textdate -> specialday

interval -> period  
interval -> modifierinterval period  
period -> month  
period -> season

relativeday ->today | tomorrow | yesterday | the day after tomorrow | the day before tomorrow  
specialday -> Christmas day | Thanksgiving day  
weekday -> Monday| Tuesday| Wednesday| Thursday| Friday | Saturday | Sunday  
modifierdate -> this | last | next  
month -> January| February  
season -> Autum| Winter  
modifierinterval -> middle of| the end of

Propose a way of normalizing these temporal expressions.

Several examples

(day, month, year)  
longdate (day,month, year) -- **normalized**(day, month, year))  
relativeday (sem(Val))– actualdate (Date), Add( Date,Val,normalized(FinalDate))  
|tomorrow(sem(1)), yesterday(sem(-1))  
specialday (day,month, year) -- **normalized**(day, month, year))

```
modifierdate(M) weekday() -- actualdate (Date),
if ((Mod ==this or Mod ==next) && daynumber(weekdate)> daynumber ( Date)){
    finaldaynumber = daynumber(weekdate) - daynumber ( actualdate);
    if(Mod == next) {finaldaynumber =finaldaynumber +7;
        actualizemonth(Date, finaldaynumber) }}
    }
Monday - 0| Tuesday-1| Wednesday-2| Thursday-3| Friday-4 | Saturday-5 | Sunday-6
```