HOW TO USE CONTEXT FREE GRAMMARS AND Q-GRAMMARS IN MAPLE

M. DELEST, J.P. DUBERNARD

ABSTRACT. The purpose of this note is to present a MAPLE library computing on algebraic grammars and q-grammars in a first approach of a language or a q-equation. We list here the main procedures.

1. INTRODUCTION

The methodology of M.P. Schützenberger [14] has lead to solve many enumeration problems. It consists in coding by words of an algebraic language the combinatorial objects to be enumerated ; if this language is generated by a context free grammar G, the generating function obtained from G is the solution of the problem. In particular, many applications are in the field of planar maps [4] or polyominoes [7, 8]. An overlook to this method can be found in [8] or [15].
Many problems can not be solved by this methodology and need the introduction of q-calculus, for example if the generating function of the studied objects according some parameters is not algebraic. The q-grammars were introduced by Delest and Fédou [5] in order to get some non algebraic equations by means of attribute grammars. In some sense, they extend the previous methodology. The q-equations which in general are not algebraic, can be obtained by this method. New results concerning polyominoes according to the area have been found using the q-grammars [1, 6].
On the other hand, many computer algebras libraries have been developped to help combinatoricists in their work. Let us quote the software of Wilf and Zeilberger which tries to prove automatically identities relative to (ordinary and "q") hypergeometric multisums/integrals [17, 16] or of Plouffe which, from the first terms of a series, researchs the corresponding generating function [13]. In a similar approach, we have implemented in a Maple package [2, 3], the notion of q-grammars. It contains tools which can not prove results but, using its, we can surround some properties on the studied combinatorial objects.

2. THE MAPLE PACKAGE COM_QGRAM

The package can be found by ftp anonymous at ftp.u-bordeaux.fr in the library
pub/ComputerScienceDpt/Software
in two Unix versions. The first, called COM_QGRAM, must be installed in the share library of Maple and can be used as any package of the share library using the command "with". The second version, called QGRAM, is like any of your

personnal Maple file. Once the package is read in your Maple session, you get description of syntax and functions using the help commands.
These packages have functions for either the Schützenberger methodology or the q-grammars. It is based on the classical functions of Maple dealing with commutative and non-commutative algebra. We only describe here the main function. The reader can find a manual by ftp in [10] (pub/ComputerScienceDpt/Publications/Rapports-internes/RR-9267.ps) or [11].

3. Usage outline

The user declares a grammar using the function *consgram* ; the result is a table according to the classicals fields of an algebraic grammar.
The function *itere* gives the first terms of the language generated by the grammar using a user criterion, called weigth. In general case, iterating a grammar leads to a construction of an infinite labelled tree. The weight which is introduced in the computation, stops the commutations according to the length of the branches. The implementation is made in a first depth left order on an infinite tree.
Two functions, *commute* and *comprime* allow to compute on partial commutative generating series. The function *commute* uses the Perrin algorithm [12].
The theorems on q-grammars [9] are implemented in the function *calcule_q_equation*, which the parameters is a q-grammar and returns the corresponding q-equations.
The function *q_itere* gives the first q-terms generated by a system of q-equations using a user weight to stop the iteration.
Below are the details of a Maple session which handles Dyck words.

```
> with(share):
> with(COM_QGRAM):
# gives the grammar G of Dyck words
> G:=consgram({x,y,t},{C},C,C=xty+xtyC+xCy+xCyC);
                G := table([
                        produc = table([
                                    C = xty + xtyC + xCy + xCyC
                                ])
                        axiome = C
                        alphterm = {t, y, x}
                        alph_non_term = {C}
                    ])

# defines the weigth
> poids:=<taille(mot,x)|mot>:
#words having at most 4 letters x
> serie:=itere(G,[poids,'+',4]);
 serie := xty + xtyxty + xtyxtyxty + xtyxtyxtyxty + xtyxtyxxtyy
      + xtyxxtyy + xtyxxtyxtyy + xtyxxxtyyy + xtyxxtyyxty + xxtyy
      + xxtyxtyy + xxtyxtyxtyy + xxtyxxtyyy + xxxtyyy + xxxtyxtyyy
      + xxxxtyyyy + xxxtyyxtyy + xxtyyxty + xxtyyxtyxty
      + xxtyyxxtyy + xxtyxtyyxty + xxxtyyyxty

# compresses the series output
> comprime(serie);
    xty + xtyxty + xtyxtyxty + xtyxtyxtyxty + xtyxtyx^2ty^2
         + xtyx^2ty^2 + xtyx^2tyxty^2 + xtyx^3ty^3 + xtyx^2ty^2xty
         + x^2ty^2 + x^2tyxty^2 + x^2tyxtyxty^2 + x^2tyx^2ty^3
```

```
         + x^3ty^3 + x^3tyxty^3 + x^4ty^4 + x^3ty^2xty^2
         + x^2ty^2xty + x^2ty^2xtyxty + x^2ty^2x^2ty^2
         + x^2tyxty^2xty + x^3ty^3xty

# non-commutative substitution
> subst(x=aa,y=epsilon,serie);
     aat + aataat + aataataat + aataataataat + aataataaaat
          + aataaaat + 2 aataaaataat + aataaaaaat + aaaat
          + 2 aaaataat + 3 aaaataataat + 2 aaaataaaat + aaaaaat
          + 3 aaaaaataat + aaaaaaaat

# 2nd argument = letters in commutative
# 3rd argument = commutation rules
> commute(serie,{},{{x,t}});
 txy + txytxy + txytxytxy + txytxytxytxy + txytxytxxyy + txytxxyy
    + txytxxytxyy + txytxxxyyy + txytxxyytxy + txxyy + txxytxyy
    + txxytxytxyy + txxytxxyyy + txxxyyy + txxxytxyyy + txxxxyyyy
    + txxxyytxyy + txxyytxy + txxyytxytxy + txxyytxxyy
    + txxytxyytxy + txxxyyytxy

# t is commutating with x and y
> commute(serie,{t});
        2         3           4             3              2
t xy + t  xyxy + t  xyxyxy + t  xyxyxyxy + t  xyxyxxyy + t  xyxxyy
   3            2            3                       2
+ t  xyxxyxyy + t  xyxxxyyy + t  xyxxyyxy + t xxyy + t  xxyxyy
     3            2                        2
+ t  xxyxyxyy + t  xxyxxyyy + t xxxyyy + t  xxxyxyyy + t xxxxyyyy
   2            2           3            2           3
+ t  xxxyyxyy + t  xxyyxy + t  xxyyxyxy + t xxyyxxyy + t xxyxyyxy
   2
+ t  xxxyyyxy

# all the letters are commutating
> commute(serie);
         2  2  2    3  3  3    4  4  4      4  3  4      3  2  3
x t y + x  t  y  + x  t  y  + x  t  y  + 6 x  t  y  + 3 x  t  y
           4  2  4   2    2    3    3    4    4
      + 6 x  t  y + x  t y  + x  t y  + x  t y

# definition of the q-grammar
> qsyst:=calcule_q_equation(G,[q],[q,0],[epsilon,lg_to(C,t)],
>  [epsilon,lg_to(C,t),0]);
 qsyst := table([
         q_axiome = G[axiome](t, y, x)
         q_eq = table([
                     C(t, y, x) = qxty + qxty.(C(t, y, x))
 + .x.(C(qt, y, x)).y
                          + ..x.(C(qt, y, x)).y.(C(t, y, x))
                 ])
         ens_qvar = {C(t, y, x)}
     ])

# definition of a new weight
> qpoids:=proc(mot)
> local i,poids;
> poids:=0;
> for i from 1 to nops(mot) do
>    if type(op(i,mot),`function`)
>    then poids:=poids+1
>    fi
> od;
> poids
> end:
```

```
# words of the q-series obtained with at most 3 iterations
> q_itere(qsyst,[qpoids,`+`,3]);
   qxty + qxtyqxty + qxtyqxtyqxty + qxtyqxtyqxtyqxty
       + qxtyqxtyxqxqtyy + qxtyxqxqtyy + qxtyxqxqtyqxqtyy
       + qxtyxxqxqqtyyy + qxtyxqxqtyyqxty + xqxqtyy
       + xqxqtyqxqtyy + xqxqtyqxqtyqxqtyy + xqxqtyxqxqqtyyy
       + xxqxqqtyyy + xxqxqqtyqxqqtyyy + xxxqxqqqtyyyy
       + xxqxqqtyyqxqtyy + xqxqtyyqxty + xqxqtyyqxtyqxty
       + xqxqtyyxqxqtyy + xqxqtyqxqtyyqxty + xxqxqqtyyyqxty
```

References

1. **M. Bousquet-Mélou**,"q-énumération de polyonominos convexes", Thèse de l'Université de Bordeaux 1 (1991).
2. **B.W. Char,K.O. Geddes, G.H. Gonnet, B.L.Leong, M.B. Monagan, S.M. Watt**, "MAPLE Language Reference Manual", Springer-Verlag (1992).
3. **B.W. Char,K.O. Geddes, G.H. Gonnet, B.L.Leong, M.B. Monagan, S.M. Watt**, "MAPLE Library Reference Manual", Springer-Verlag (1992).
4. **R. Cori** ,"Un code pour les graphes planaires et ses applications", Astérisque, Soc. Math. France 27 (1975).
5. **M.P. Delest, J.M. Fédou**,"Attribute grammars are useful for combinatorics",Theor. Comp. Sci. (1992).
6. **M.P. Delest, J.M. Fédou**,"Enumeration of Skew Ferrers Diagrams", Discrete Math. (to appear).
7. **M.P. Delest, D. Gouyou-Beauchamps, B. Vauquelin** ,"Enumeration of parallelogram polyominoes with given bond and site perimeter", Graphs and Combinatorics 3 (1987), 325-339.
8. **M.P. Delest, X.G. Viennot**,"Algebraic languages and polyominoes enumeration", Theor. Comp. Sci. 34 (1984), 169-206.
9. **M.P. Delest, J.P. Dubernard**,"q-grammars : results, implementation",STACS 94, Lecture Notes in Computer Science 775 (1994), 377-388.
10. **J.P. Dubernard**,"Utilisation des grammaires algébriques et q-grammaires en MAPLE",Rapport LaBRI $n°$ 790.93, 1992.
11. **J.P. Dubernard**,"q-grammaires et polyominos parallélogrammes",Thèse de l'Université Bordeaux I, Juin 1993.
12. **D. Perrin**, "Words over partially commutative alphabet", A. Apostolico and Z. Galil, eds., Combinatorial Algorithms on Words, NATO ASI-series, Vol F12(1985), 329-340.
13. **S. Plouffe**
"Approximations des séries génératrices et quelques conjectures", Rapport LaBRI n° 9261 (1992).
14. **M.P. Schützenberger**,"Ceratin elementary families of automata", Proc. Symp. on Mathematical Theory Automata, Polytechnic Institute of Brooklyn (1962), 139-153.
15. **X.G. Viennot**,"Enumerative combinatorics and algebraic languages", Lecture Notes in Comp. Sci. 199 (1985), 450-464.
16. **H.S. Wilf, D. Zeilberger** ,"An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities", submited, 1991.
17. **H.S. Wilf, D. Zeilberger** ,"Toward computorized proofs of indentities", Bull. Amer. Soc. 23 (1990), 147-158.

(Delest) LaBRI, Université de Bordeaux I, 351 Cours de la Libéraition, 33405 Talence Cedex, France.

E-mail address: maylis@labri.labri.u-bordeaux.fr

(Dubernard) Département d'Informatique, Université de Poitiers, 40 Avenue du recteur Pineau, 86022 Poitiers Cedex, France.

E-mail address: dubernar@matpts.univ-poitiers.fr.