

Table of Contents

1 INTRODUCTION.....	1
2 GETTING STARTED WITH CoCoA.....	3
3 SETTING UP THE RING.....	4
4 EXECUTING COMMANDS.....	5
5 ABOUT IDENTIFIERS AND SPECIAL CHARACTERS.....	7
6 GENERAL PROCEDURES.....	8
7 ALGEBRAIC PROCEDURES AND FUNCTIONS.....	9
8 PARTIAL COMPUTATIONS.....	13
9 SUBSTITUTION.....	13
10 EXPRESSIONS.....	14
APPENDIX A OPTIONS FOR GRÖBNER BASIS COMPUTATION.....	16
APPENDIX B NUMERIC POLYNOMIALS AND FUNCTIONS.....	19
APPENDIX C SYNTAX.....	21

CoCoA User's Manual¹

(v. 0.99b - May 1989)

Alessandro Giovini & Gianfranco Niesi

Department of Mathematics, University of Genova (Italy)

1 Introduction

CoCoA is a Macintosh system for doing **C**omputations in **C**ommutative **A**lgebra. It runs on any Macintosh with at least 512K of RAM, but it takes advantage of any additional available memory; it runs also under MultiFinder (even though it is not yet capable of performing computations in the background). It is written in Pascal (apart from a few “glue routines” in assembler); the release 0.99b of the system consists of about 19,000 lines of code.

CoCoA has been designed for offering the maximum ease of use and flexibility to the mathematician with little or no knowledge of computers.

CoCoA is capable of performing simple and sophisticated operations on multivariate polynomial rings and on various data connected with them (ideals, modules,

¹The CoCoA system (hereby *the system*) has been entirely designed and implemented by Alessandro Giovini and Gianfranco Niesi (hereby *the authors*) exception made for the code related to the computation of Poincaré series and of Hilbert functions which has been written by Massimo Caboara and Anna Bigatti. The development of the system has taken great benefit from long discussions with Lorenzo Robbiano and Teo Mora, which have also suggested several improvements to some key algorithms and which we gratefully acknowledge.

The system is *freely* distributed to anyone who requests it, under the only following condition: any research activity which benefits of the usage of the system *should explicitly quote the system, the authors and the address where the system can be requested*.

The system is distributed on a 800k Macintosh diskette containing also other files, notably the document “CoCoA x.y User's Manual”, the Microsoft Word 3.02 file containing this manual. The system can be freely redistributed to other users but in this case the whole contents of the original diskette should be copied. In case of redistribution, the new users should notify the authors so they are included in a user's list. All users in this list will be kept up to date about the system progress and will have the possibility of freely receiving updated copies of the system as soon as they are available by simply sending a blank diskette at the address below.

Alessandro Giovini or Gianfranco Niesi,
Department of Mathematics, University of Genova,
viale Leon Battista Alberti 4, 16132, Genova – ITALY.

Questions and suggestions can be also sent to the following electronic mail addresses:

astes@igecuniv.bitnet (A. Giovini)

cocoa@igecuniv.bitnet (G. Niesi)

The system is distributed “as is”. The authors make no warranty on the fitness of the system for any particular application. They shall not be liable for any direct, indirect, special, incidental or consequential damages in connection with the use of the system or of the manual.

The system has been developed using THINKTM Pascal from Think Technology, so portions of the system are copyright ©Think Technology.

matrices); polynomials may have coefficients either in the field \mathbf{Q} of rational numbers or in the residue ring \mathbf{Z}_p . A current limit of the system is that the numerator and denominator of the coefficients in \mathbf{Q} must not exceed $2^{31}-1=2147483648$, and the integer p for \mathbf{Z}_p must not exceed $2^{15}-1=32767$.

Every operation (sometimes called *computation*) is performed within the “current ring”, which the user can easily set up and change by just pulling down a menu and editing some values; the advanced user has also the possibility of changing the values of some special parameters; the meaning of these parameters can be nevertheless safely ignored by most users.

The user can open up to eight windows (standard text-editing Macintosh windows) in which data can be entered in a format which has been chosen to be as close as possible to the usual mathematical notation. Several kinds of computations can be performed on the entered data and the results can be stored for later use. If the user modifies the ring, then the already entered or computed data can be easily transferred to the new ring (when that makes sense).

The system is capable of performing the basic operations on polynomials (sums, products, powers, derivatives), on ideals (sums, products, powers), on modules (sums) and on matrices (sums, products, powers, determinants) as well as more advanced operations like intersection and division of ideals, syzygies of ideals or modules, resultant of two polynomials, elimination and substitution of variables, etc. We just give some examples of expressions the system can evaluate:

$$\begin{aligned} & xy^{\text{TM}} - 2/3xt(x-z)^{\text{TM}}(xp-t-3/5)\mathcal{E} + 4)^\infty \\ & (x/x\leftarrow x\mathcal{E}^{\text{TM}})[x\neq F, x\mathcal{E}=G, x\leftarrow H] \\ & (\text{Elim}(t, \text{Ideal}(t\mathcal{E}-x, t\mathcal{C}-y, t^\infty-z))\&J):(x-y) \end{aligned}$$

The system displays the exponents as superscripts and the indexes as subscripts, taking advantage of the graphics capabilities of the Macintosh (the system uses its own font whose special features will be described later).

Below you see a typical CoCoA screen after that the user has selected and evaluated the expression on the first line; the result of this evaluation is displayed immediately below.

The screenshot shows a Macintosh window titled "CoCoA 1" with a menu bar containing "File", "Edit", "Keyboard", and "Ring". The main text area displays the command:

```
Resultant(normalform(-x3y2+x3z+x2-yz, ideal(x3-y,x4-z)), (x2+y5)2, x)
```

Below the command is a 6x6 matrix of coefficients:

```

-----
1   - z2   0   0   0   0
0   1   - z2   0   0   0
0   0   1   - z2   0   0
0   0   0   1   - z2   0
1   0   2y5   0   y10   0
0   1   0   2y5   0   y10
-----

```

Below the matrix is the result:

```
y20 + 2y15z4 + y10z8
```

The core of the system is an implementation of Buchberger's algorithm for computing the Gröbner basis of an ideal; the algorithm has been optimized in several senses and it is used as a 'building block' for some of the operations that the system is capable of doing; for most uses the user can however completely ignore the theory of Gröbner bases and even their existence: the system will do all the necessary 'Gröbner stuff' in the background. However, for an optimal use of the system (and of some system parameters) some knowledge of the theory may be useful.

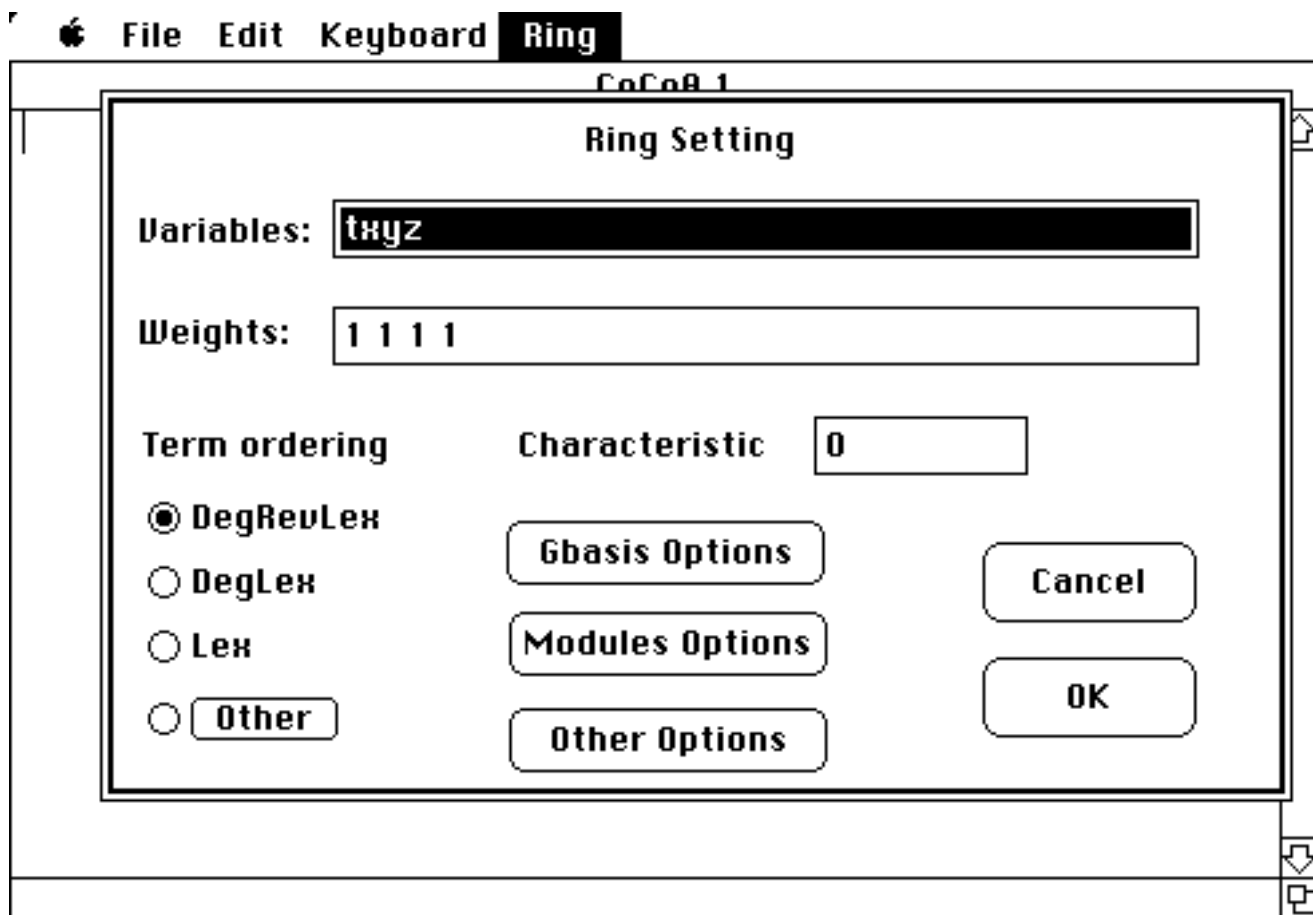
2 Getting started with CoCoA

Upon startup the system draws the menu bar, sets the default ring to be $\mathbf{Q}[t,x,y,z]$ and opens a text editing window.

The menu bar contains five menus. The first three menus (Apple Menu, File Menu, Edit Menu) are the standard menus which allow typical Macintosh operations (opening desk accessories, opening, saving and printing files and doing basic editing actions).

The fourth menu is the Keyboard menu. This menu allows the user to switch between the Italian keyboard and the UK keyboard. This makes easier to locate the keys corresponding to special characters (exponents, indexes, etc.).

The last menu is the Ring menu. At any time a "current ring" is active within the system; this ring is the polynomial ring within which the evaluations are done. The current ring setting can be examined and changed at any time by simply pulling down the menu 'Ring' and selecting the item 'Set Ring'. Then a dialog box appears (called 'Ring Setting'); this dialog shows how the current ring is set and enables the user to change the setting. The dialog box here below shows the default setting (the one set at startup); its content will be discussed in more detail in the next section.



3 Setting Up The Ring

The current ring is completely determined by:

- the name (and hence also the number) of the indeterminates;
- their weights;
- the field of the coefficients;
- the term ordering.

Usually the user simply specifies the name of the variables and the field of the coefficients, leaving the weights to 1's (the default) and the term ordering to DegRevLex (the default); in most cases this suffices for the system to work optimally.

The maximum number of indeterminates that the system can manage is 23. An indeterminate is represented by a name consisting of either a single letter or a letter followed by an index having up to eleven digits. Here 'letter' means a character in the set $\{a, \dots, z, A, \dots, Z\}$. Corresponding upper and lower case letter yield *different indeterminates*. For example, x , X and X_{123} are legal (and different) indeterminates; they are set by entering the string "xXx123" in the "Set Ring" dialog (*no* spaces!).

Each indeterminate has a weight (a positive integer). The system defaults to the value of 1 for all the indeterminates. If the weights specified in the dialog box are less than the indeterminates, then the missing ones – the last ones – are assumed to be equal to 1 (the system notifies the user that it is making this assumption).

The coefficients can be chosen to be either rational numbers or integers modulo p . This is done by setting the characteristic either to 0 or to p ($p < 32767$). When the dialog box is closed pushing the OK button, the system checks whether p is a prime number and, if it is not, then a message is given. However all non negative integers are accepted, so it is possible to do some work with polynomials whose coefficients are not in a field but *it is up to the user to ensure that no illegal operation will be attempted*. The system simply reports an error message every time the inversion of a zero-divisor is attempted, but does not interrupt a computation; beware hence of computing Gröbner bases in presence of zero-divisors, since it is very likely that you end up with a very long sequence of error messages.

Given a current ring, the system can handle objects of the following kind:

- polynomials;
- lists of polynomials;
- matrices of polynomials;
- ideals;
- modules (submodules of a free module).

Each object may have a 'name' (see sect. 4). Names of objects have the same structure of names of variables, i.e. they can be a single letter or a letter followed by an index (but, of course, an object cannot have the same name of a variable of the current ring).

The polynomials are always kept sorted with respect to the given term ordering. All the operations involving polynomials preserve and benefit of this ordering. In the current release the user can choose among three predefined term orderings or define custom orderings. The predefined term orderings are the following: the 'degree reverse lexicographic' (which is the default one), the 'degree lexicographic', and the 'pure lexicographic'. Selection of one of these orderings is achieved by just pushing the corresponding button in the dialog box. The first two term orderings use the weights appearing in the dialog box to calculate the degree. For computations requiring temporarily a different term ordering (for example, in the case of the elimination of variables from an ideal), the system changes automatically the term ordering to a more

suitable one, performs the computation, and then restores the initial term ordering and gives its output with respect to this one; in this way the user never has to deal with temporary changes.

For special purposes, the user can also enter directly a matrix corresponding to a custom ordering by pushing the button 'Other'. A new dialog is opened where the current ordering is displayed. The user can freely modify this ordering; the system does *not* check whether this is a term ordering, so be careful. **Note:** The ordering becomes effective only upon *exit* from the 'Ring setting' dialog, and so it applies to the number of indeterminates *at that moment*. So if the user enters a $4 \cdot 4$ matrix, changes the number of indeterminates to 5 and then exits, an error message will be displayed.

Each term ordering on the current ring induces several term orderings on free modules. The system allows to choose between two of them by pushing the button 'Modules options' in the ring dialog box. If M is a free module of rank r over the ring R and $\{e_i\}$ is the canonical basis, then a term of M is an element of the form Te_i where T is a term of R . The possible choices for comparing two terms of a module are:

- the ordering called 'TO-Pos' (which is the default one)

$$T_1 e_i > T_2 e_j \text{ iff } T_1 > T_2 \text{ in } R \text{ or, if } T_1 = T_2, i < j$$

- the ordering called 'Pos-TO'

$$T_1 e_i > T_2 e_j \text{ iff } i < j \text{ or, if } i = j, T_1 > T_2 \text{ in } R.$$

By pushing the button 'Gbasis options' in the ring dialog box, another dialog box appears; this dialog allows the advanced user to change the values of some special parameters which affect the way in which the computation of a Gröbner basis is carried out. These parameters will be discussed in some detail in Appendix A.

Finally the button 'Other Options' and the corresponding dialog box allow to set some parameters which affect the way in which numerical polynomials are written and to choose the variable for Poincare series. More details will be given in Appendix B.

4 Executing commands

As a general rule, to notify the system that some action has to be performed on some data, the user has to press the key 'Enter'; at this point a part of the text of the window is taken as being the 'currentcommand':

- if there is no selection point (the cursor is blinking somewhere) then the part of text between the beginning of the line where the cursor is placed and the cursor itself is taken as current command;
- if there is a nonnull selection range, then the whole selection is taken as current command (in this way the system can process multiline commands).

A command may consist of a single instruction or of a sequence of instructions separated by semicolons. In this latter case the instructions are executed in sequence; if some error occurs, then the instructions subsequent to the error are not executed.

Each instruction can be:

- **an expression** (including the case of a simple identifier): in this case the expression is evaluated and the result is displayed starting at the line after the cursor;

examples:

$(x+y)^2$; j ; $\det(M) + G \bmod F$; $(I \cap J)^2$

- **an assignment** (a command of the form 'identifier = expression'): in this case the expression is first evaluated and then the resulting value is assigned to the identifier; the computed value is not displayed; this is the way of giving a name to an object. The new value overrides any value previously associated to that name, which is hence lost.

examples:

$F = (xy+zt)^5$; $G = F [x=3y^2-t^5]$; $J = \text{Ideal}(F, x+y, G^2+xz) \cap I^2$

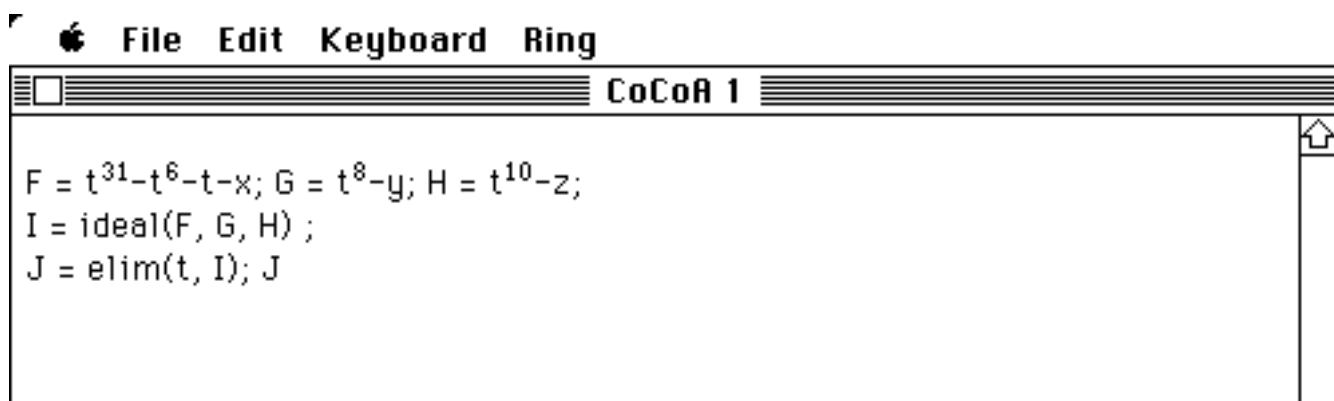
- **a procedure call** (possibly with a parameters list): the behaviour of the system in this case strictly depends on the procedure called;

examples:

`List;` `Help;` `Cancel(F,G,J);` `Gbasis(I)`

Named objects (i.e., objects which have been associated to a name by the execution of assignments) are stored in an *environment* the environment is a dynamic data structure which the system keeps transparent to the user. The name of an object may be used in expressions at every place where an object of that kind can occur. The list of the identifiers present in the environment together with their type can be obtained by executing the command 'List'.

The expressions follow a simple syntax which is very close to the usual mathematical one (its definition by means of syntax graphs can be found in Appendix C, but for most uses the 'intuition' will suffice). If the user selects for example the three lines below and hits enter,



then the system executes the following sequence of commands: first evaluates and assigns the three polynomials to the corresponding names, then builds the ideal consisting of these three polynomials and assigns it to I, then computes the ideal J obtained from I by eliminating the indeterminate t and finally displays the ideal J.

```

File Edit Keyboard Ring
CoCoA 1
F = t31-t6-t-x; G = t8-y; H = t10-z;
I = ideal(F, G, H);
J = elim(t, I); J

Ideal J
      y5 - z4 ,
      xy4z5 + 1/2yz7 - 2xy4z2 - xy4 - 2x2y2z - x3z2 - yz4 - 1/2x2y2 -
1/2yz2 + 1/2yz ,
      x3y4z4 - 1/2x2yz7 + xy3z5 - 2x3y4z + x2yz4 + 1/2z7 - 3/2x4y2 - x5z -
2xy3z2 + 1/2x2yz2 - xy3 - 5/2x2yz - z4 - 1/2x2y - 1/2z2 + 1/2z ,
      z8 - 2z5 - 2xy3 - x2yz - z3 + z2 ,
      y2z6 - 1/2xz7 - 2y2z3 + xz4 + 1/2x3y - y2z - 3/2xz2 + y2 - 1/2xz ,
      x6y4z3 + 3x4y3z5 - 2x5y2z6 + x4y3z4 - 2x6y4 + x3z7 - 6x4y3z2 + 4x5yz3
+ 2xy4z4 + x2y2z5 - x8 - 2x4y3z - yz7 - 3x4y3 - 4x5yz - 2x3z4 + yz6 - 5x5y -
4xy4z - 2x2y2z2 - x3z2 + 2yz4 - 4x2y2 - 3x3z - 2yz3 + yz2 - 2yz + y

```

Note: CoCoA 0.99b does not handle properly a transfer of values between rings in the following cases: a change in the order (or name) of the indeterminates, or a change to the characteristic of the ring. The term-ordering can instead be changed safely. In case of doubt it is safer to reenter the relevant data in the environment from the window. This bug will be eventually fixed.

5 About identifiers and special characters

There are two kinds of identifier:

- the identifier of an indeterminate or of an object;
- the identifier of a procedure or of a function built in the system.

The identifiers of the first kind must be made of either a single letter or of a letter followed by an index having at most 11 digits. Different identifiers do always correspond to different objects, and identifiers are case-sensitive (corresponding upper- and lower-case letters yield different identifiers). If to a name which is already associated with an object is assigned another object, then the new association overrides the old one.

The identifier of a procedure or a function consists of a contiguous sequence of letters. The corresponding upper- and lower-case letters in the word are equivalent, so, for example, LIST, list, List are all accepted and they all call the same procedure. If the procedure or function has some parameters, then these are enclosed within round brackets and usually separated by commas.

The system uses its own font containing special characters and makes provision for using both the italian keyboard and the UK keyboard. The italian keyboard is assumed by default. If the UK keyboard is used, then the user has to select 'UK keyboard' from the Keyboard menu. We give here a table showing how to enter special characters:

symbol to enter	UK keyboard	Italian keyboard
superscript digit	option-digit	option-shift-digit
subscript digit		option-digit option-shift-digit
intersection:	\cap	& &
square brackets	[,]	cmd-[, cmd-]
curly brackets		cmd-{ , cmd-} { , }

6 General procedures

The following procedures are not related to the part of the system devoted to algebraic computations, but only to the system management itself.

Cancel(i_1, \dots, i_n)

This procedure deletes the objects associated to the identifiers i_1, \dots, i_n from the environment; **Note:** large values stored in the environment tend to slow down the system; so values should be removed using this command as soon as they are not needed any more.

Help

This procedure invokes the online help. A window, called 'Help Window' containing a non editable text is opened; however the text can be copied and/or printed.

Info

This procedure displays information related to memory usage and garbage collection; it is of interest only to the system developers.

List

This procedure displays the list of the identifiers of the objects present in the environment together with their type (not their value);

Timeron

This procedure enables the displaying of the time that individual instructions take to execute; after its execution, the execution of subsequent commands will be followed by their execution time;

Timeroff

This procedure disables the displaying of the execution time.

Write(i) or **Write**(i . ext)

This procedure displays the object stored in the environment with identifier i or, in the second form, the part of the ideal or module specified by 'ext'; more precisely the system will display only

- the generators of i if ext = Gens;
- the gbasis of i if ext = Gbasis;

- the standard basis of i if $\text{ext} = \text{Stbasis}$; (only for ideals)
- the Poincaré series of i if $\text{ext} = \text{Poincare}$; (only for ideals)

7 Algebraic procedures and functions

We give in this section the list of the functions available in the system, together with the pattern with which they can be called; some functions can also be called like procedures, in which case after the call the resulting value will usually be displayed on the screen. For each of the patterns the type of the parameters and of the object returned or displayed are described.

The functions of the first five groups (divided according to the type of the value returned) leave unchanged their arguments. The sixth group contains those functions that can modify their argument. Finally a last group contains some special functions related to the theory of the Hilbert function of an ideal.

- *Functions returning a polynomial.*

Der(F, x)

F : polynomial; x : indeterminate;

It returns the derivative of the polynomial F w.r.t. the indeterminate x .

Det(M)

M : matrix;

It returns the determinant of the square matrix M .

Gcd(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

It returns the greatest common divisor of the polynomials F_1, \dots, F_n .

Homog(x, F)

x : indeterminate; F : polynomial;

It returns the polynomial which is the homogeneization of the polynomial F w.r.t. the indeterminate x and current weights of the indeterminates.

Lcm(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

It returns the least common multiple of the polynomials F_1, \dots, F_n .

NormalForm(F, I)

F : polynomial; I : ideal;

It returns the polynomial which is the normal form of the polynomial F w.r.t. the ideal I . If I is an ideal stored in the environment and a Gbasis of I has already been computed, then that Gbasis is used otherwise the Gbasis of I w.r.t. the current term ordering is computed.

Resultant(F, G, x)

F, G : polynomials; x : indeterminate;

It returns the resultant of the polynomials F and G w.r.t. the indeterminate x and also displays the Sylvester matrix.

- *Functions returning a list of polynomials.*

Note that the value of the expression $\{F_1, \dots, F_n\}$ where F_1, \dots, F_n are polynomials is the list of the polynomials F_1, \dots, F_n and it may be assigned to an identifier.

GBasis(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

GBasis(L)

L : list of polynomials;

Both of these two functions return a list of polynomials which is a Gröbner basis of the ideal generated by F_1, \dots, F_n or by the

list L . The result depends on the current term ordering and on the setting of the parameters in the window 'GBasis options'. The default setting produces the fully interreduced Gröbner basis with monic polynomials, which is unique.

Gens(I)

I : ideal;

It returns the list of polynomials that generate the ideal I .

Homog(x, F_1, \dots, F_n)

x : indeterminate;

F_1, \dots, F_n : polynomials;

Homog(x, L)

x : indeterminate;

L : list of polynomials;

These two functions return the list of polynomials whose members are the homogenized of the polynomials F_1, \dots, F_n or of polynomials in L w.r.t. the indeterminate x .

InterReduce(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

InterReduce(L)

L : list of polynomials;

These two functions return the list of polynomials obtained from the polynomials F_1, \dots, F_n or from the members of L by interreducing them. Note that, in general, the result depends on the current term ordering.

LeadTerm(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

LeadTerm(L)

L : list of polynomials;

They return the list of the leading terms of the polynomials F_1, \dots, F_n or of the members of L .

StBasis(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

StBasis(L)

L : list of polynomials;

They return a list of polynomials which is a standard basis of the ideal generated by F_1, \dots, F_n or by L .

- *Functions returning an ideal.*

Elim(x, I)

x : indeterminate;

I : ideal;

This function returns the ideal obtained by eliminating the indeterminate X from the generators of the ideal I . A suitable term ordering is set during the computation and then the initial ordering is restored.

Elim($x..y, I$)

x, y : indeterminates;

I : ideal;

It returns the ideal obtained by eliminating the indeterminates included between X and Y from the generators of the ideal I . (Note the two dots '..' as part of the syntax). During the computation a suitable term ordering is set and at the end the initial ordering is restored.

Homog(X, I)

X : indeterminate;

I : ideal;

It returns the ideal which is the homogeneization of the ideal I w.r.t. the indeterminate X .

Ideal(F_1, \dots, F_n)

F_1, \dots, F_n : polynomials;

It returns the ideal generated by the polynomials F_1, \dots, F_n .

Ideal(L)

L : list of polynomials;

It returns the ideal generated by the list of polynomials L .

LeadTerm(I)

I : ideal;

This function returns the ideal of the leading terms of the ideal I .

MaxMinors(M)

M : $p \cdot (p+1)$ or $p \cdot (p-1)$ -matrix;

It returns the ideal generated by the maximal minors of the matrix M .

- *Functions returning a module.*

InterReduce(M)

M : module;

It returns the module obtained from M by interreducing its generators.

LeadTerm(M)

M : module;

It returns the module generated by the leading term of the elements of the Gbasis of M w.r.t. the current term ordering.

Module(r, F_1, \dots, F_{rs})

r : positive integer;

F_1, \dots, F_{rs} : polynomials;

This function returns the submodule (of a free module of rank r) generated by the r -uples $(F_1, \dots, F_r), \dots, (F_1, \dots, F_{rs})$.

If necessary an appropriate number of zero polynomials is added at the end of the list.

Syz(I) I : ideal or module;It returns the module of syzygies of the generators of I .

- *Functions returning a matrix.*

Matrix(r, s, F_1, \dots, F_{rs}) r, s : positive integers; F_1, \dots, F_{rs} : polynomials;

It returns the $r \cdot s$ matrix whose entries are the polynomials F_1, \dots, F_{rs} . If the polynomials are less than rs , then an appropriate number of null polynomials is added at the bottom of the list.

Transp(M) M : matrix;It returns the transposed of the matrix M ;

- *Functions that can modify objects in the environment.*

When the parameter in the following patterns is an object stored in the environment (more precisely it is the identifier of an object stored in the environment), then the function modify the object (usually it adds some data to the object) and, if it is called like a procedure, nothing will be displayed.

GBasis(I) I : ideal or module;

It computes the reduced Gröbner basis of the ideal I with respect to the current term ordering and returns the result as list of polynomials; if I is the identifier of an ideal or module, then the Gröbner basis is also stored in the object I and not displayed, otherwise it is simply sent to the screen.

InterReduce(I) I : ideal;It returns the ideal obtained from I by interreducing the generators.**StBasis(I)** I : ideal or module;It computes a standard basis of the ideal I and it behaves like **GBasis**.**Poincare(I)** I : ideal;

It computes the Poincaré series of the ideal of the leading terms of the ideal I with respect to the current term ordering and displays the result; if I is the identifier of an ideal, then the Poincaré series is also stored in the object I .

- *Special procedures and functions of an ideal.*

If I is an homogeneous ideal, then I and the ideal $T(I)$ of the leading terms of the ideal I have the same Hilbert function. Hence for homogeneous ideals the functions below give some numerical invariants of the ideal I . Numeric value are returned as constant polynomials.

Dim(I) I : ideal;

It returns and displays the dimension of the ideal of the leading terms of the ideal I with respect to the current term ordering; if I is the identifier of an ideal, then the result is also stored in the object I .

HilbCoeff(I)

I : ideal;

It displays the coefficients of the Hilbert polynomial, in the binomial basis of the second type and alternating the signs, of the ideal of the leading terms of the ideal I with respect to the current term ordering.

Hilbert(I)

I : ideal;

It returns and displays the Hilbert polynomial of the ideal of the leading terms of the ideal I .

HilbertFn(I, n)

I : ideal;

It displays the n -th value of the Hilbert function of the ideal of the leading terms of the ideal I .

HilbertFn(I)

I : ideal;

It displays the values of the Hilbert function as far as the index of regularity is reached and then the Hilbert polynomial.

Mult(I)

I : ideal;

It returns and displays the multiplicity of the ideal of the leading terms of the ideal I with respect to the current term ordering.

Reg(I)

I : ideal;

It returns and displays the index of regularity (i.e. of the ideal of the leading terms of the ideal I with respect to the current term ordering).

8 Partial computations

It is possible to compute only a part of a Gbasis or of a syzygy module: when the commands `Gbasis(...)` and `Syz(...)` are followed by an index N , then the computation is stopped as soon as an element of degree greater than N is found. If the ideal is homogeneous, then this gives the whole Gbasis (or syzygy module) up to degree N .

9 Substitution of indeterminates

It is possible to substitute indeterminates with polynomials within expressions. If E is an expression denoting an object O , X_1, \dots, X_n are indeterminates and F_1, \dots, F_n are polynomials, then

$$E[X_1 = F_1, \dots, X_n = F_n]$$

is the object obtained from evaluating E to O and then by simultaneously substituting in O all the occurrences of the variables X_1, \dots, X_n with the polynomials F_1, \dots, F_n . Of course if O is an ideal or a module, then the substitution takes effect only on the generators and the resulting value is not associated with any Gröbner basis and/or standard basis with which is possibly associated O .

Example: if F is the polynomial $X^2 + Y^3 + Z^4 + T^5$ then $F [X=Y, Y=X, T=(Z-T)^2]$ is the polynomial $Y^2 + X^3 + Z^4 + (Z-T)^{10}$.

10 Expressions

Expressions have a rather natural syntax, quite similar to that of Pascal language. Expression are made of factors and terms. These objects are described in Appendix C by means of syntax graphs. Here we just give an informal description of this syntax.

- An **expression** consist of a list of **terms** separated by the (additive) operators: '+', '-'.
- A **term** consists of **factors** separated by (multiplicative) operators, which are: '*', ':', 'mod', 'div', '∩'.
- A **factor** consists of one of the followings
- a **coefficient**;
- an **identifier** (of an indeterminate or of an object);
- a **function call**
- an **expression** enclosed in round brackets

a factor can be possibly followed by an **exponent** and/or a **substitution** and/or the factorial operator.

- A **coefficient** is an **integer** (with or without sign) or a **fraction** i.e. a (signed) integer followed by the character '/' and a positive integer.

The multiplication symbol (*) can be omitted, and exponentiation does not require a symbol; it suffices to write the exponent with the appropriate characters (see sect. 5).

The priority of the operations – from the highest to the lowest – is:

- exponentiation, substitution, factorial
- *, :, ∩, div, mod
- +, -.

When in doubt, parentheses can be used to enforce a particular order of evaluation.

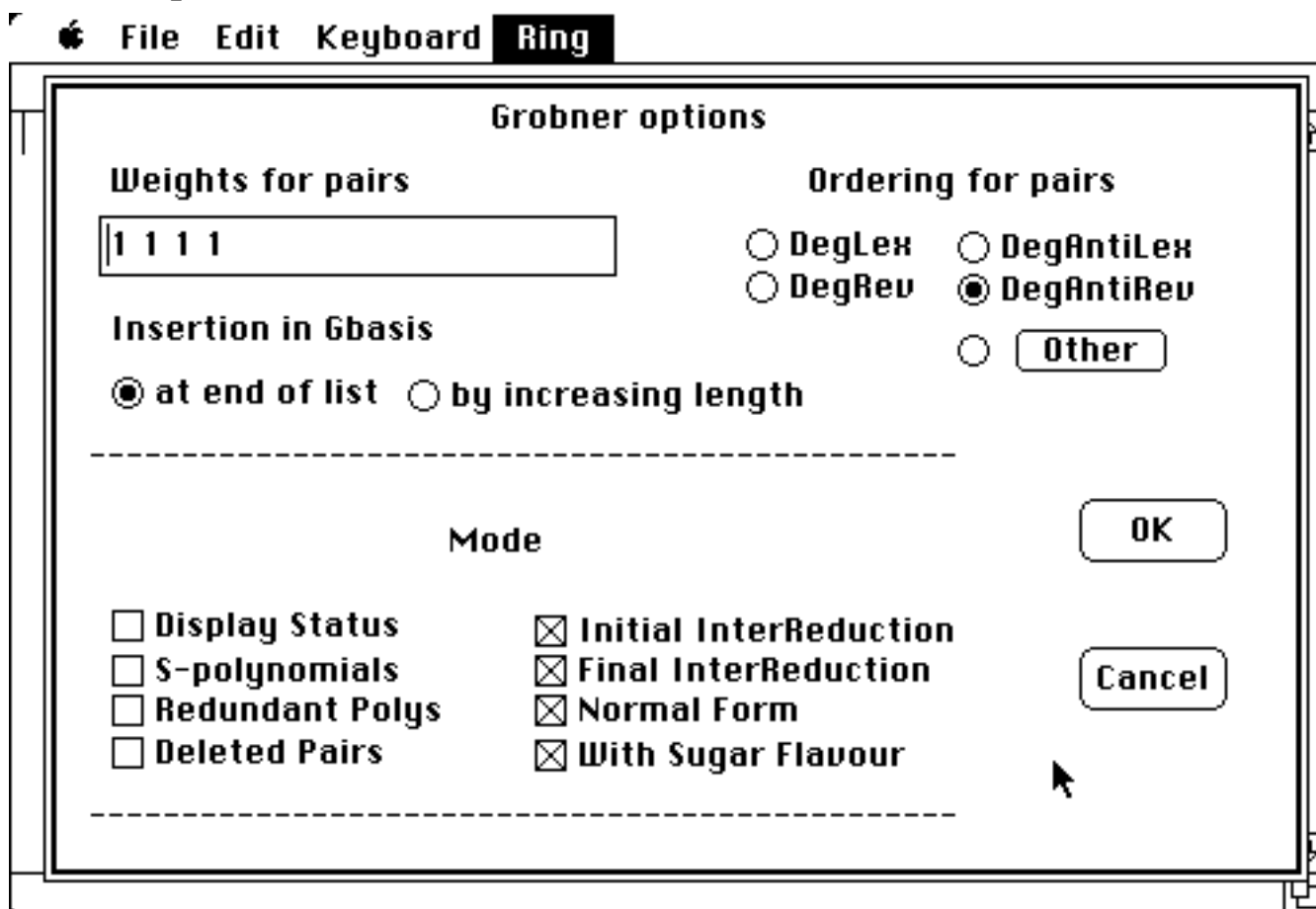
Operations with equal priority are performed from left to right.

The following table shows what operations the system can perform between two objects of the same or of different types; the first column contains is the type of the first operand and the first row the type of the second. So, for example, the symbol ':' in the box on the fourth row and second column indicates that it is possible to divide an ideal by a polynomial.

	Polynomial	List of poly	IdealModule	Matrix
Polynomial	+ - * mod div		*	*
List of poly		+		
Ideal	* :		+ * \cap :	
Module			+	
Matrix	*			+ - *

Appendix A Options for Gröbner basis computation

The user may set several parameters that affect the way in which the computations of a Gröbner basis is carried out. To have access to these parameters the user has simply to push the button 'Gbasis options' in the dialog box 'Ring Setting'; then a second dialog box, called 'Gröbner options' is displayed. Here below is shown the default setting of these parameters.



By means of this dialog the user may choose:

- The weights and the ordering for the critical pairs.

The critical pairs are ordered w.r.t the l.c.m. of the leading terms of the two polynomials of the pair and processed by increasing order; to this aim the indeterminates can be weighted and ordered in a different way from the ones used to manipulate polynomials as elements of the ring. However, if the option “with sugar flavour” is selected, then the critical pairs are processed in an order which is as close as possible to the order which would have been chosen if the polynomials had been homogeneous, hence the weights and ordering are ignored (in this way the algorithm runs generally faster – it is the well-known “CoCoA with sugar”).

- How to insert in the basis polynomials produced during the

computation.

The list of polynomials that at the end of the computation will be a Gröbner basis can be kept ordered by increasing length of its members or in the ‘natural way’ i.e.

by adding new polynomials at the end of the list. Choosing an ordering rather another may have considerable effect on the computation.

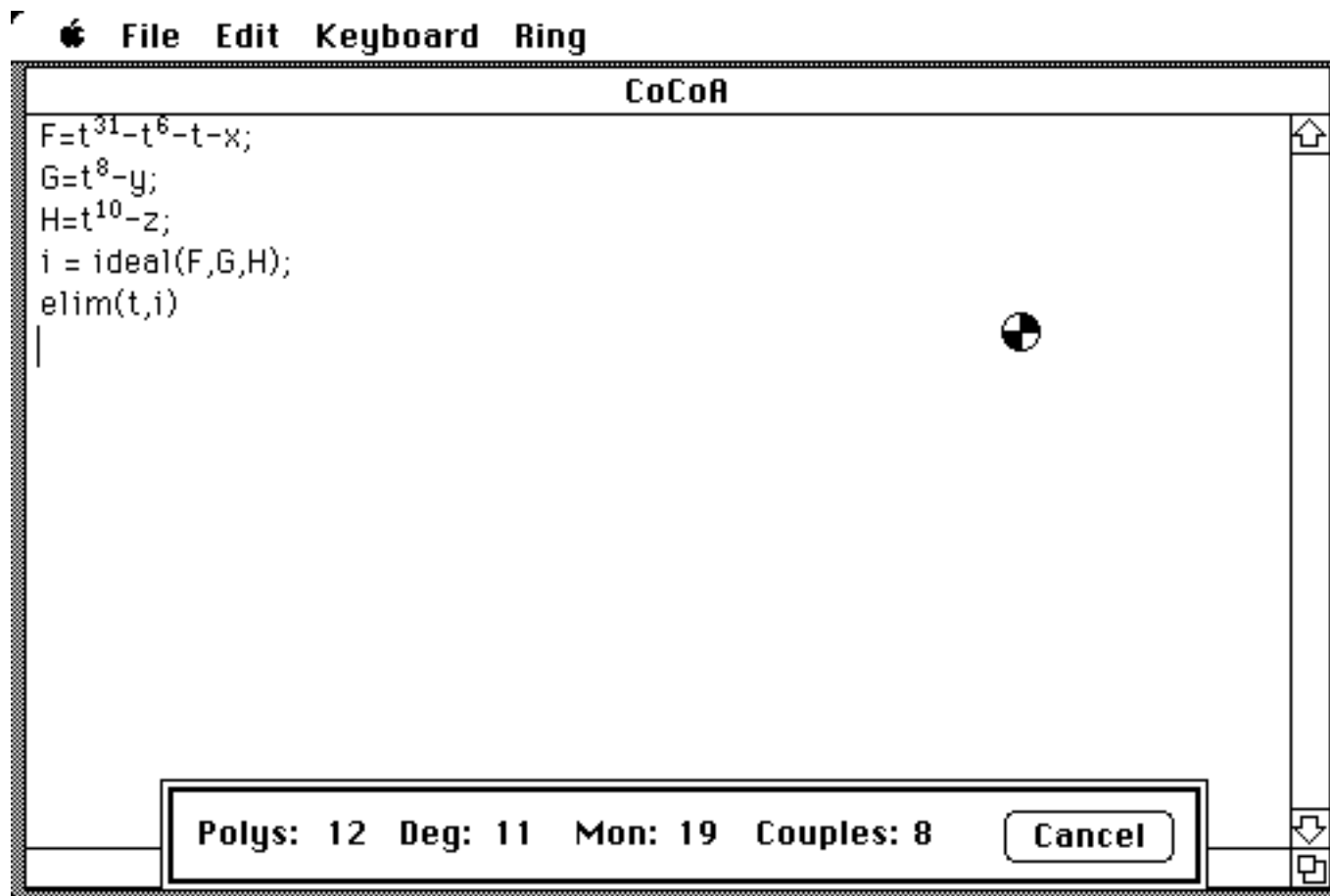
- Whether or not to interreduce the starting list of polynomials
- In this way if one of the starting polynomials is reduced to some other polynomial, then the original polynomial is not used in the reduction steps.
- Whether or not to interreduce the final list of polynomials
 - Whether or not to normalize the polynomial produced during the computation.

If the box 'Normal Form' is not checked, then only the leading term of a new polynomial is reduced as far as it is possible. When 'Normal Form' and 'Final InterReduction' are both checked the resulting Gröbner basis is the (unique) reduced Gröbner basis w.r.t. the current term ordering.

If the user wishes to follow on the screen the activity connected to the computation of a Gröbner basis, then, by checking the appropriate boxes, it may ask to the system to display

- the new nonnull S-polynomials as soon as they are computed and reduced;
- the leading term of redundant polynomials, i.e. of those polynomials whose leading term is multiple of some other polynomial in the basis;
- the useless critical pairs whose S-polynomial hence is not computed;
- the Display Status panel;

When this box is checked, an informative panel is kept open during the computations of Gröbner bases: the housekeeping of this panel can be a little time-wasting but the information it can give can be very important to the user about the progress of the computation; this panel displays (and continuously updates) the number of polynomials computed, their maximum degree, their maximum length and the number of critical pairs still to be considered; there is also a 'Cancel' button to interrupt the computations; obviously the results obtained from an interrupted computations are wrong. Note also that there may be a considerable delay between the point at which the cancel button is pressed and the point where the computation stops (CoCoA checks for such an interruption only after having considered a critical pair and not during interreduction). This is what the screen looks like during a computation of a Gröbner basis when DisplayStatus is selected:



Appendix B Numeric polynomials and functions

In this appendix we will describe two 'binomial' shapes available in the system for a univariate polynomial with integer coefficients.

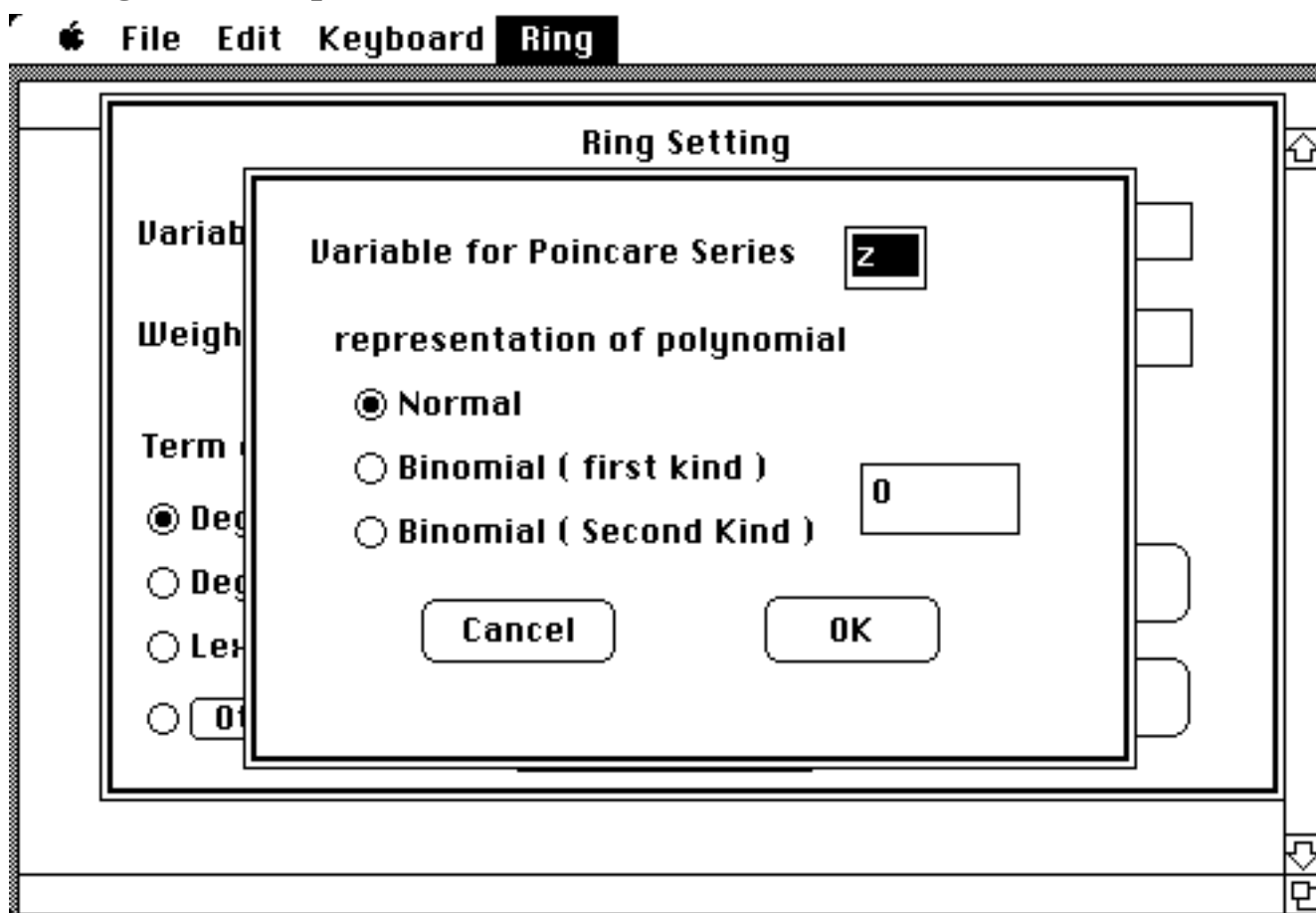
Given a polynomial F and an integer n , we put

$$\text{bin}(F,n) = F(F-1)(F-2)\dots(m-n+1) / n!$$

Let c be an integer and let E be the ring of the univariate polynomial in the variable x with integer coefficients. Then both the following two sets are basis of E (as free \mathbb{Z} -module):

- $\{ \text{bin}(x+c, i) / i \in \mathbb{N} \}$
- $\{ \text{bin}(x+c+i, i) / i \in \mathbb{N} \}$

The user can choose, besides the usual basis, one of these two by pushing the button 'Other Options' in the ring dialog box and handling the dialog box that will be open (in particular can be given the integer c). The same dialog box allows also to choose the variable by which the Poincaré series will be written. Here below is shown the default setting of these parameters.



The following function are useful in some computations involving 'binomials'.

Bin(F, n)

F : polynomial; n : integer;

It returns the polynomial $F(F-1)(F-2)\dots(F-n+1) / n!$ (if F is an integer, then this is the usual binomial).

BinExp(n, i)

n, i : positive integers;

It displays the binomial expansion of h relative to i , that is

$$h = \text{bin}(n_i, i) + \text{bin}(n_{i-1}, i-1) + \dots + \text{bin}(n_j, j)$$

where $n_i > n_{i-1} > \dots > n_j \geq j \geq 1$ (they are uniquely determined by h and i)

BinPower(n, i)

n, i : positive integers;

It displays the value of the expression

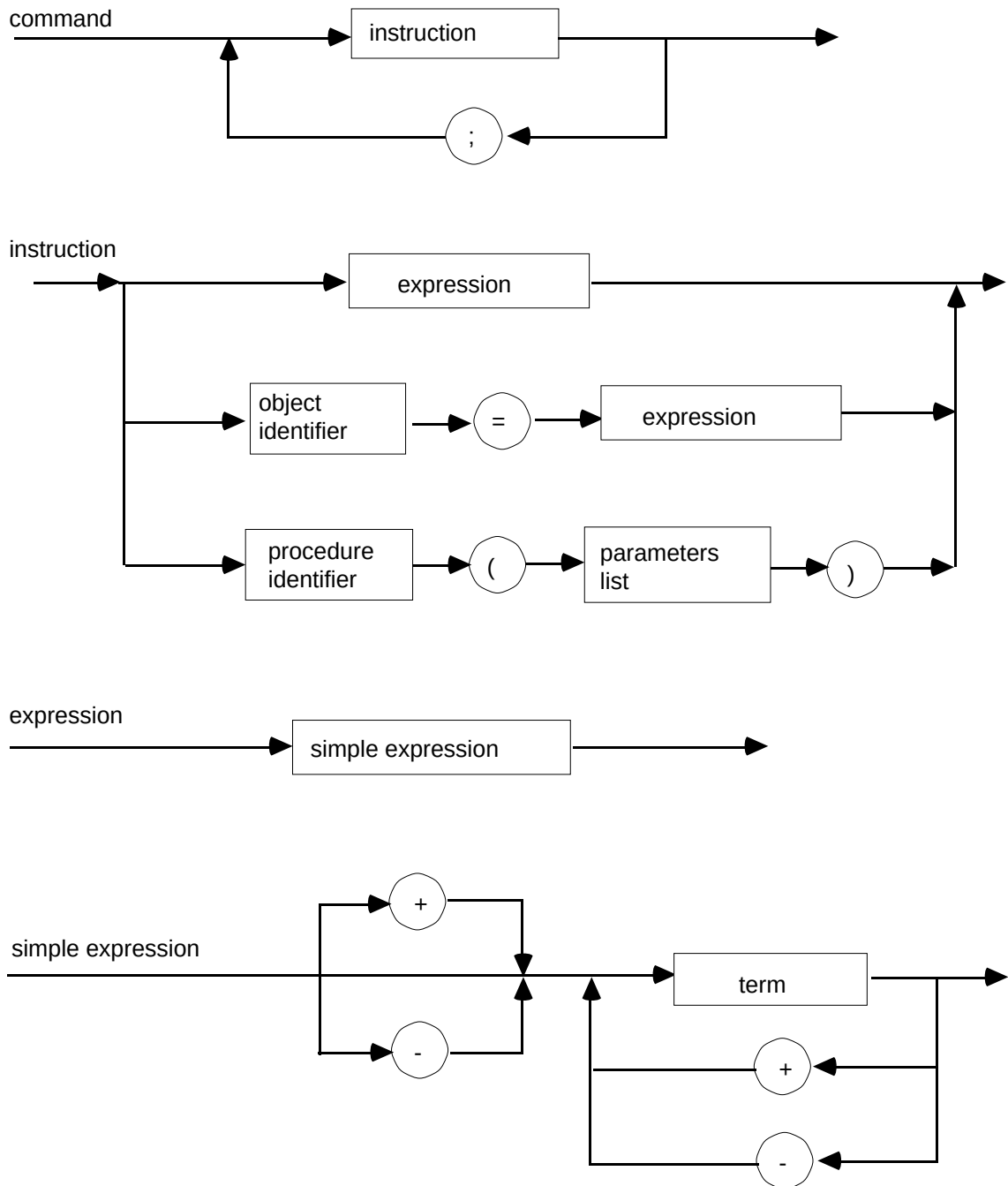
$$\text{bin}(n_i+1, i+1) + \text{bin}(n_{i-1}+1, i) + \dots + \text{bin}(n_j+1, j+1)$$

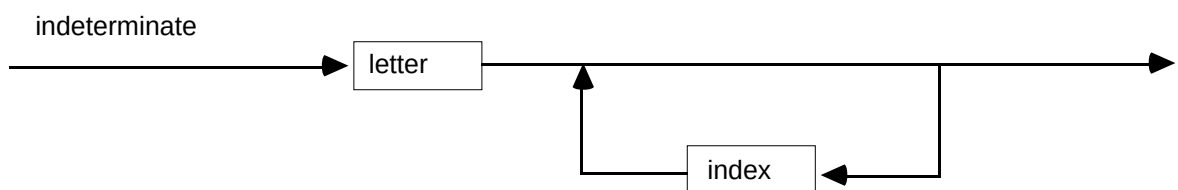
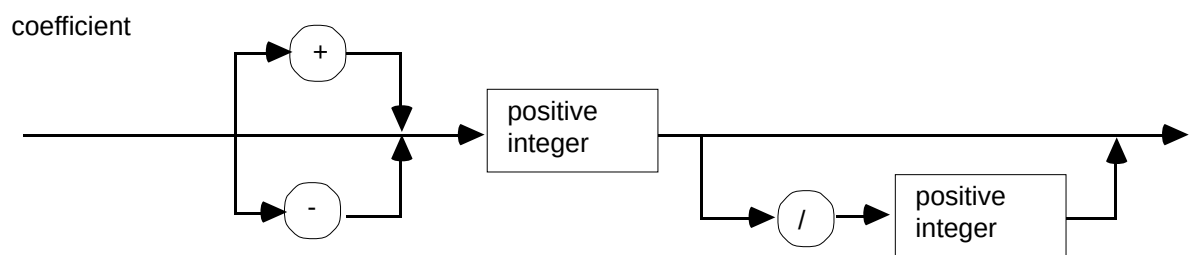
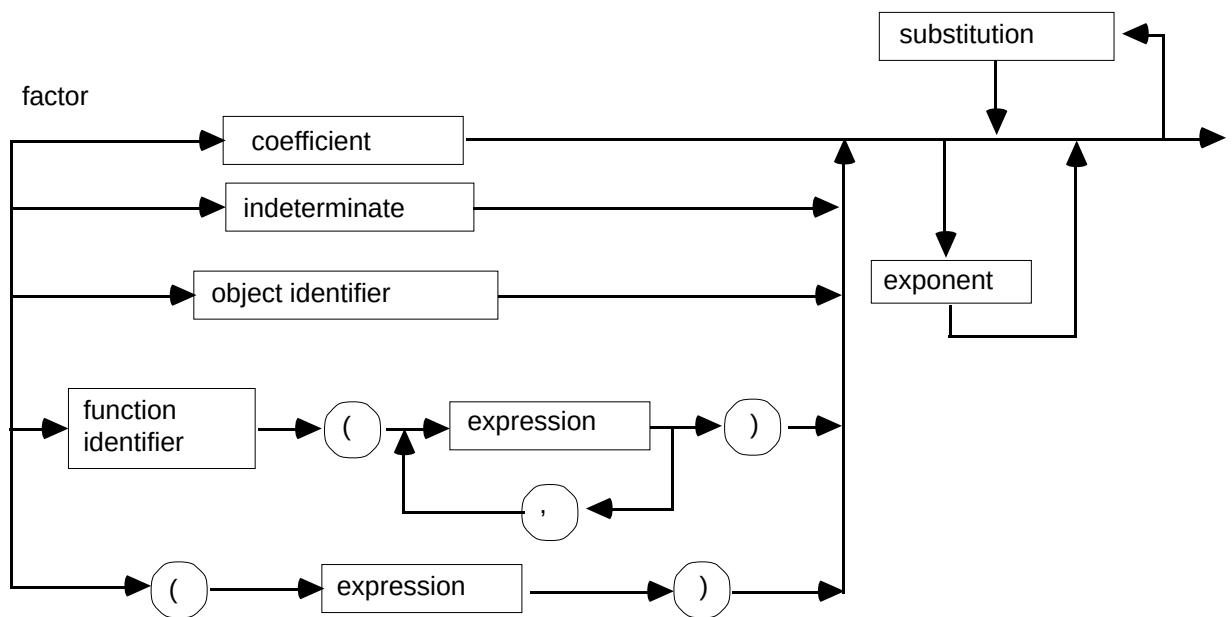
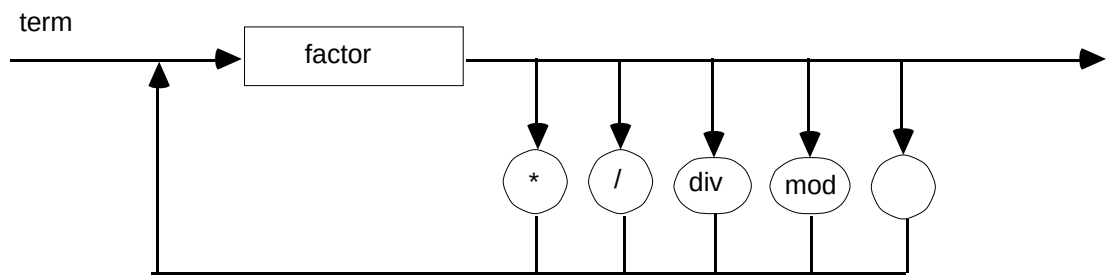
where

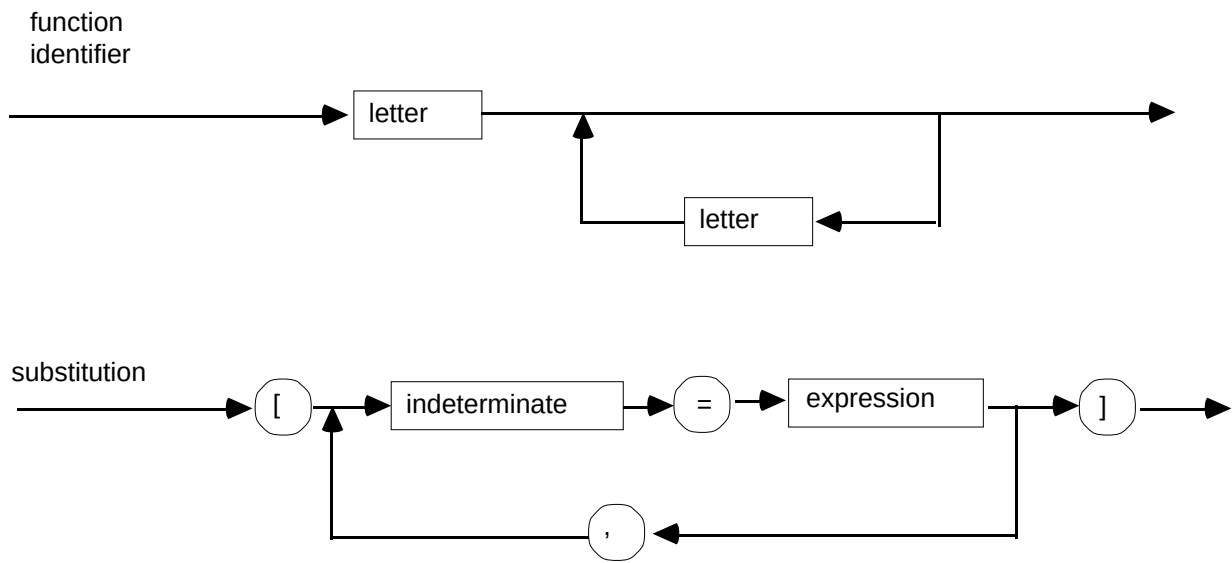
$$\text{bin}(n_i, i) + \text{bin}(n_{i-1}, i-1) + \dots + \text{bin}(n_j, j) = h.$$

Appendix C Syntax

We report here the syntax diagrams of commands, instructions, expressions, etc.







Index

- Apple Menu 3
- assignment 6
- $\text{Bin}(F, n)$ 19
- BinExp 20
- $\text{BinPower}(n, i)$ 20
- Cancel 8
- characteristic 4
- CoCoA with sugar 16
- coefficient 14
- coefficients 4
- critical pairs 16
- current command 5
- current ring 2, 3, 4
- custom ordering 5
- default ring 3
- DegRevLex 4
- $\text{Der}(F, x)$ 9
- desk accessories 3
- $\text{Det}(M)$ 9
- $\text{Dim}(I)$ 12
- Display Status panel 17
- Edit Menu 3
- $\text{Elim}(x \text{..} y, I)$ 11
- $\text{Elim}(x, I)$ 10
- environment 6
- expression 14
- factor 14
- factorial operator 14
- File Menu 3
- Gbasis options 5
- $\text{GBasis}(F_1, \dots, F_n)$ 10
- $\text{GBasis}(I)$ 12
- $\text{Gcd}(F_1, \dots, F_n)$ 9
- $\text{Gens}(I)$ 10
- Gröbner basis 3
- Gröbner options 16
- Help 8
- Help Window 8
- $\text{HilbCoeff}(I)$ 13
- $\text{Hilbert}(I)$ 13
- $\text{HilbertFn}(I, n)$ 13
- $\text{HilbertFn}(I)$ 13
- $\text{Homog}(x, F)$ 9
- $\text{Homog}(x, F_1, \dots, F_n)$ 10
- $\text{Homog}(X, I)$ 11
- $\text{Homog}(x, L)$ 10
- $\text{Ideal}(F_1, \dots, F_n)$ 11
- $\text{Ideal}(L)$ 11
- indeterminate 4
- Info 8
- $\text{InterReduce}(F_1, \dots, F_n)$ 10
- $\text{InterReduce}(I)$ 12
- $\text{InterReduce}(L)$ 10
- $\text{InterReduce}(M)$ 11
- Keyboard menu 3
- $\text{Lcm}(F_1, \dots, F_n)$ 9
- $\text{LeadTerm}(F_1, \dots, F_n)$ 10
- $\text{LeadTerm}(I)$ 11
- $\text{LeadTerm}(L)$ 10
- $\text{LeadTerm}(M)$ 11
- List 6, 8
- list of the polynomials 10
- $\text{Matrix}(r, s, F_1, \dots, F_{rs})$ 12
- maximum number of indeterminates 4
- $\text{MaxMinors}(M)$ 11
- menu bar 3
- $\text{Module}(r, F_1, \dots, F_{rs})$ 11
- Modules options 5
- $\text{Mult}(I)$ 13
- MultiFinder 1
- name 4
- $\text{NormalForm}(F, I)$ 9
- object 4
- Other Options 19
- Partial computations 13
- $\text{Poincare}(I)$ 12
- priority 14
- procedure call 6
- $\text{Reg}(I)$ 13
- $\text{Resultant}(F, G, x)$ 9
- Ring menu 3
- Ring Setting 3
- sequence of instructions 5
- Set Ring 3
- $\text{StBasis}(F_1, \dots, F_n)$ 10
- $\text{StBasis}(I)$ 12
- $\text{StBasis}(L)$ 10
- Substitution 13
- Sylvester matrix 9
- syntax diagrams 21
- $\text{Syz}(I)$ 12
- term orderings 4
- Timeroff 8
- Timeron 8
- $\text{Transp}(M)$ 12
- UK keyboard 3
- weight 4
- weights 16
- $\text{Write}(i \text{ . ext})$ 8
- $\text{Write}(i)$ 8
- zero-divisor 4
- $\{F_1, \dots, F_n\}$ 10