COCONUT Deliverable D14

Set of combination algorithms for state of the art modules

EPFL – *Swiss Federal Institue of Technology – Lausanne*
IRIN – *University of Nantes*
University of Vienna

July 30, 2003

Chapter 2

Report Modules

Introduction

Report modules are auxiliary modules able to process some internal information. They are mainly provided for the user to have a view of the current internal processes (current solutions, state of the table, state of the store, ...).

2.1 C Write

2.1.1 Copyright

This information is copyright by Dr. Hermann Schichl from the Mathematics Department of the University of Vienna.

2.1.2 Module name

Module name is `C Write`

2.1.3 Type of module

`C Write` is a report module.

2.1.4 Description of the module

`C Write` is a module which produces `C` code for evaluating all constraints and the objective function of the model stored in the work node.

2.1.5 Limits

There are no hardcoded limits.

2.1.6 Service *print*

`C Write` produces two `C` functions, one for evaluating the objective and one for evaluating the constraints.

Kind of input

Any work node.

Control parameters

`print multipliers`: This parameter is of type `bool` specifies whether the variables associated with the Karush-John conditions (i.e. the multipliers) are printed as `y` and the multiplier of the gradient of the objective as `k`. If the value is `false` all variables are printed alike. The default is `true`.

`with comments`: With this parameter of type `bool` the user decides whether additional comments like bounds and node numbers are printed within the generated `C` code. The default is `true`.

In addition there is a number of string parameters which can be used to change function and variable names and insert additional code into between the generated `C` expressions.

header: The header part of the `C` file.

objective preamble: The part before the objective function.

objective postamble: The part between function definition and the generated expression.

constraints preamble: The part before the constraints function.

constraints postamble: The part between function definition and the generated expression.

switch preamble: The part before the `switch` which distinguishes between the different constraints.

switch postamble: The function part after the `switch`.

constraints function end: The remaining part of the constratints function.

footer: The end of the `C` file.

x: The name of the variables array.

y: The name of the multipliers array.

kappa: The name of the multiplier of the gradient of the objective function.

type: The type of the variables and multipliers.

kappatype: The type of the multiplier of the gradient of the objective function.

2.2 DAG Write

2.2.1 Copyright

This information is copyright by Dr. Hermann Schichl from the Mathematics Department of the University of Vienna.

2.2.2 Module name

Module name is `DAG Write`

2.2.3 Type of module

`DAG Write` is a report module.

2.2.4 Description of the module

`DAG Write` is a module which outputs the DAG of the work node in sequentialized format suitable for reading.

2.2.5 Limits

There are no hardcoded limits.

2.2.6 Service *print*

`DAG Write` produces an ASCII serialized representation of the model DAG in the work node.

Kind of input

Any work node.

Control parameters

None.

2.3 Ranges Write

2.3.1 Copyright

This information is copyright by Dr. Hermann Schichl from the Mathematics Department of the University of Vienna.

2.3.2 Module name

Module name is `Ranges Write`

2.3.3 Type of module

`Ranges Write` is a report module.

2.3.4 Description of the module

`Ranges Write` is a module which prints the ranges of all variable nodes or of all nodes of the work node.

2.3.5 Limits

There are no hardcoded limits.

2.3.6 Service *print*

`Ranges Write` prints the ranges of all variable nodes or of all nodes either in compact format as an array or in large format containing node numbers or variable names.

Kind of input

Any work node.

Control parameters

`preamble`: This parameter is of type `string` specifies what is printed before the ranges. The default is `"ranges:¨`.

`postamble`: Also of type `string`, this parameter is used to define what is printed after the ranges. The default is `"\n\n"`.

`compact`: With this parameter of type `bool` it is decided whether the ranges are print as a comma separated list of intervals, or whether it is printed with additional information like node numbers or variable names. The default is `true`.

`variables`: Using this `bool` the user can specify whether all node ranges are printed or only the ranges of the variable nodes. Default: `false`.

`entries per line`: This integer specifies how many entries are printed per line of output. Default: `4`.

2.4 Solution Report

2.4.1 Copyright

This information is copyright by Dr. Hermann Schichl from the Mathematics Department of the University of Vienna.

2.4.2 Module name

Module name is `Solution Report`

2.4.3 Type of module

`Solution Report` is a report module.

2.4.4 Description of the module

`Solution Report` is a module which prints the solution information.

2.4.5 Limits

There are no hardcoded limits.

2.4.6 Service *print*

`Solution Report` prints the best points from the `point` table and the entries of the `solution` table.

Kind of input

The search database.

Control parameters

None.

2.5 Table Report

2.5.1 Copyright

This information is copyright by Dr. Hermann Schichl from the Mathematics Department of the University of Vienna.

2.5.2 Module name

Module name is `Table Report`

2.5.3 Type of module

`Table Report` is a report module.

2.5.4 Description of the module

`Table Report` is a module which prints the entries of a specified table from the search database.

2.5.5 Limits

There are no hardcoded limits.

2.5.6 Service *print*

`Table Report` prints specified rows from one table of the search database. The user can decide whether all table entries should be printed or whether only entries from the work node view should be reported.

Kind of input

Any work node, the search database.

Control parameters

`table`: This parameter is of type `string` specifies entries of which table are printed.

`col`: This `int`-indexed parameter of type `string` specifies the column name of the ith output.

`enable`: This `int`-indexed parameter of type `bool` specifies whether ith output should be printed. The default is `true`.

`prefix`: This `int`-indexed parameter of type `bool` specifies what is printed before the value of the ith output row.

In addition there are a number of `string` parameters influencing the format of the output.

`header`: This string is printed before row entries are printed. Default: `""`

`footer`: This is printed after all rows. Default: `"\n"`

`separator`: This string is printed between two rows. Default: `"\n\n"`

`interdisp`: This is the separator of two columns of the same row. Default: `"\n"`