mipuser

Table of Contents

IBM MIP Solutions 1
MIP Solutions User Guide 3

[User's Guide | Download | Bottom of page]

IBM MIP Solutions

The mixed integer programming solutions product, OSLMSLV, is a fully stand-alone program that incorporates library modules to solve mixed integer programming problems. More specifically, it uses a branch and bound algorithm to minimize a linear objective function, subject to linear and integrality constraints. OSLMSLV can be executed from the command line, or from a shell script. Brief descriptions of the branch and bound algorithm incorporated into this program is included in the opening sections of the library User's Guide.

OSLMSLV, and each of the OSL application programs, accepts many command line parameters (which may be accessed from a parameter file) to specify program options. The program options available in all the application programs include: the size of the program workspace (limited only by available memory), whether the objective is to be minimized or maximized, names of input and/or output files, and specification of new values for OSL control variables. All the settable control variables are accessible by this means. Control variables and their uses are explained in the library User's Guide.

All the application programs accept problem data (in MPS format) from *stdin*, and direct output to *stdout*. Input data may be piped in from another program, which generates the data or retrieves it from some other source, or it may be read from a file. Output may be redirected into a file in the usual way, or written directly to an output file whose name has been supplied as a program option.

For further information about the IBM MIP Solutions product, click on User Guide, to view the document, or on Download to obtain a trial copy of the program and a sample MPS data set.

If you would like to have softcopy of the html user guide on your PC, click here to download the zipped postscript HTML (24 KB)

If you prefer to handle hardcopy of documentation click here to download either a postscript (125KB) or zipped postscript (40KB) version of the documentation.

Notes:	1. Viewing these postcript files is problematical. An inability to view them satisfactorily does not necessarily mean that they will not print correctly. 2. There are no hypertext links in the postcript files, and so the cross references are undetectable.

[**Top of Page | Previous Page | User's Guide**]

IBM MIP Solutions User Guide

© IBM Corp. 1995, 1997. All rights reserved.

The mixed integer programming solutions product, OSLMSLV, uses a branch and bound algorithm to solve MIP problems with a linear objective functions. At a user's direction, it will use depth first, breath first, or its own proprietary search strategies. OSLMSLV can be executed from the command line. No compilers or run-time libraries are required. Command line options allow users great flexibility in controlling program execution.

Usage Examples:

Example 1

oslmslv "?"

This command results in usage information and the display of keywords, their meanings along with the default values.

Example 2

oslmslv -maxsols=3 < test1.mps > test1.out

The oslmslv program is started using test1.mps as input data. Output is directed to the file test1.out. The program will stop after 3 feasible integer solutions have been found.

Module Name: oslmslv

Purpose: To solve linear Mixed-Integer Programming problems
Usage: oslmslv "?" [-keyword=value] [...] [@keyword_file]
keyword_file: Name of file containing "-keyword=value" strings.
Each "-keyword=value" string in this file must be on a separate line.

Valid Keywords and Default Values:

maxmin = min|max - default: min
input = input_mps_file - default: standard input
output = listing_file - default: standard output
dspace = value - default: 3000000
isolmask = value - default: 0
iprtinfomask = value - default: 31
presolve = yes|no - default: yes
presolvetype = 0|1|2|3 - default: 3
crash = yes|no - default: yes
scale = yes|no - default: yes
inbasis = in_basis_file - default: null
outbasis = out_basis_file - default: null
outmps = out_mps_file - default: null
imaxiter = value - default: 9999999
alg = osl|primal|dual - default: osl
strategy = value - default: 1
search = depth|breadth|osl-default;osl
mpretype = 0|1|2|3 - default: 1
maxsols = value - default: 9999999
maxnodes = value - default: 9999999
heurpass = value - default: 1
sos13flag = yes|no - default: no
genprsl = 0|1|2 - default: 0
branch = up|down|osl - default: osl
maxtime = value - default: 9999999
msglevel = verbose|terse|minimal - default: verbose
matunit = value - default: 96
basunit = value - default: 97
osli_n_ = value - default: a function of n
oslr_n_ = value - default: a function of n
oslc_n_ = value - default: a function of n

Explanation of Keywords:

maxmin : Solve as a maximization or minimization problem.
input : Name of file containing input data - MPS format.
output : Name of file to receive messages.
dspace : Size in double words of the work area.
isolmask : Matrix bit mask for EKKPRTS print function.

= 1 : selects the rows
= 2 : selects the columns
= 4 : selects the nonzero elements of the model

= 8 : selects the infeasible elements of the model
= 15 or 0 or 16 : selects the entire matrix

iprtinfomask : Solution information bit mask for EKKPRTS print function.

= 1 : selects the statistics
= 2 : selects iteration count, value of the objective function and the problem status
= 4 : selects the names of the columns and rows
= 8 : selects the status of each variable
= 16 : selects the row and column solution values
= 32 : selects the dual values and reduced costs
= 64 : selects the lower bounds
= 128 : selects the upper bounds
= 256 : selects the input costs
= 512 : selects the matrix elements
= 0 : selects all of first six options (same as 63).

presolve : Use EKKPRSL function to pre-solve problem.
presolvetype : Type of reduction to be performed by EKKPRSL.

= 0 : the redundant rows are eliminated, the variables summing to zero are fixed.
= 1 : type 0 reductions are done and the doubleton rows are eliminated.
= 2 : type 0 reductions are done and variable substitutions performed.
= 3 : all of type 0, 1, and 2 reductions are performed.

crash : Use EKKCRSH function to create a starting basis.
scale : Use EKKSCAL function to scale the coefficient matrix.
inbasis : Name of file containing an input basis to be used by EKKBASI.
outbasis : Name of file containing an output basis created by EKKBASO.
outmps : Name of output file (MPS format) created by EKKBCDO.
imaxiter : The maximum number of iteration that will be done.
alg : Simplex algorithm to be used.

= primal : primal algorithm
= dual : dual algorithm
= osl : algorithm to be chosen by the solver

strategy : The bit mask for preprocessing and supernode processing.

= 0 : probes on all variables
= 1 : probes on satisfied 0-1 variables
= 2 : uses solution strategies that assume a valid integer solution has been found

= 4 : choses the branch opposite the maximum pseudo-cost
= 8 : computes new pseudo-costs as variables are branched on
=16 : computes pseudo-costs for unsatisfied variables
=32 : computes pseudo-costs for both satisfied and unsatisfied variables

mpretype : Type of MIP preprocessing and the condition of matrix on exit.

= 0 : no preprocessing is done
= 1 : uses supernodes on the branch-and-bound keeps the new matrix with extra rows and tighter bounds
= 2 : uses regular branch-and-bound (no supernodes) keeps the new matrix with extra rows and tighter bounds
= 3 : uses regular branch-and-bound (no supernodes) restores the matrix to its original form

maxsols : The maximum number of feasible integer solutions to find.
maxnodes : The maximum number of nodes to be searched through in the Branch-and-Bound tree.
heurpass : The number of heuristic passes to be made by EKKMPRE.
sos13flag : Identify SOS of type 1 and 3
genprsl : Type parameter for EKKBMPR

= 0 : do not call EKKBMPR
= 1 : transform using powers of 2
= 2 : transform using natural numbers

branch : Favorable branching direction.

= up : branches up
= down : branches down
= osl : branch is chosen by the solver

maxtime : Maximum wall clock time (seconds) allowed for optimization.
msglevel : The amount of output messages to be displayed.

= verbose : displays all and sundry
= terse : displays intermediate solutions and nodes
= minimal : displays intermediate solutions only

matunit : Identifies extension added to the file named "fort" where the matrix information is to be written.

Integer 0 <= matunit < 90

basunit : Identifies extension added to the file named "fort" where the basis information is to be written.

Integer 0 <= basunit < 90

osli_n_ : Integer Control variable n; 0 < n < 62.
oslr_n_ : Real Control variable n; 0 < n < 46.
oslc_n_ : Character Control variable n; 0 < n < 11.

See the library User's Guide. for more information on Control variables.

Example 1

oslmslv "?"

This command results in the display of valid keywords, their meanings along with the default values.

Example 2

oslmslv -maxsols=3 < test1.mps > test1.out

The oslmslv program is started using test1.mps as input data. Output is directed to the file test1.out. The program will stop after having found 3 feasible integer solutions.

Example 3

If the file key_file.001 contains the following 4 lines:

- maxnodes=50
- mpretype=0
- osli_5_=20000
- maxtime=3600

then the command:

```
oslmslv -dspace=1000000 @key_file.001 < test1.mps > test1.out
```

will start oslmslv with a work area of 1 million double words. MIP pre-processing will not be done. The program will terminate after 50 nodes have been examined or 3600 seconds (wall clock time) have elapsed or if an optimal solution has been found. Input to the program is the file test1.mps. Output is directed to the file test1.out.

Example 4

The command:

```
oslmslv -dspace=1000000 @key_file.001 -input=test1.mps -output=test1.out
```

is equivalent to:

```
oslmslv -dspace=1000000 @key_file.001 < test1.mps > test1.out
```

---

oslmslv USAGE NOTES

1. If oslmslv is having difficulty with a problem, then you should increase the size of work area using the "dspace" keyword and try solving the problem again.
2. When the option "-msglevel=terse" is specified with oslmslv, the following scheme is used to control the printing of messages giving statistics for the nodes of the branch-and-bound tree. For nodes numbered from 0 to 9, all the statistics are printed. For nodes numbered from 10 to 100, the statistics for nodes with numbers 10, 20, ..., 100 (i.e, divisible by 10) are printed. Similarly, for nodes numbered between 101 and 1000, statistics are printed only for nodes numbered 200, 300, 400, and so on. For nodes numbered 1000 and higher, the statistics are printed only for nodes numbered 1000, 2000, 3000, and so on.
3. When the option "-mpretype=value" with "value" = 1, 2, or 3 is specified with oslmslv, the MIP preprocessor function ekkmpre() is invoked. The following notes apply to the interactions between ekkmpre() and the MIP solver function ekkmslv(). Complete solution statistics are not printed, if ekkmpre() finds a feasible solution and the branch-and-bound function ekkmslv() fails to improve on it. However, the feasible solutions found by ekkmpre() are printed in the output listing as they are generated.
4. When the option "-maxsols=value" is specified with oslmslv, the number of feasible solutions to be found by ekkmslv() is limited. However, this does NOT restrict the number of solutions that may be generated by ekkmpre().
5. The MIP preprocessing function ekkmpre() attempts to add rows (cuts). Therefore, if ekkmpre() is to be used (command line option "mpretype=value" with value = 1, 2, or 3), space must be provided for these extra rows. This can be done by specifying the option "-osli_9_=value", where here "value" is the negative of the number of additional rows to be provided.

For example, to provide space for 2000 additional rows, the user may specify "-osli_9_= -2000". In oslmslv, the default value of osli_9_ is -1000.

System Requirements

The installation instructions included in the download package include System Requirements.

IBM HOME ORDER PRIVACY LEGAL CONTACT IBM