CPLEX Callable Library (C API)

Fabio Furini

June 7, 2018

Outline

Reference Manual

- The Callable Library is the C Application Programming Interface (API) of CPLEX .
- There are separate API for the C++, Java, C#.NET, and Python.

Internet Site

http://www-01.ibm.com/support/knowledgecenter/SSSA5P_12.6.2/ilog.odms.cplex.help/refcallablelibrary/homepageCrefman.html?lang=fr

CPLEX Data

CPLEX Data

- `env`: a pointer to the CPLEX environment as returned by `CPXopenCPLEX`
- `lp`: a pointer to a CPLEX problem object as returned by `CPXcreateprob`
- `status`: a pointer to an integer, where an error code is placed by a routine
- `nzcnt`: an integer that specifies the number of nonzero constraint coefficients

CPLEX Data

- `ccnt`: an integer that specifies the number of columns
- `obj`: an array containing the objective function coefficients
- `lb`: an array containing the lower bound on the columns
- `ub`: an array containing the upper bound on the columns
- `xctype`: an array containing the type of the columns ('C','B','I')
- `colname`: an array containing pointers to character strings that specify the names of the columns

CPLEX Data

- `rcnt`: an integer that specifies the number of rows
- `rhs`: an array containing the righthand side term for each row
- `sense`: an array containing the sense of each row ('L','E','G')
- `rngval`: an array containing the range values for the rows
- `rowname`: an array containing pointers to character strings that specify the names of the rows

CPLEX Data

- `cmatbeg`: Array that specifies the beginning of the nonzero elements of the columns
- `cmatind`: Array that specifies the positions of nonzero elements of the columns
- `cmatval`: Array that specifies the values of nonzero elements of the columns
- `rmatbeg`: Array that specifies the beginning of the nonzero elements of the rows
- `rmatind`: Array that specifies the positions of nonzero elements of the rows
- `rmatval`: Array that specifies the values of nonzero elements of the rows

CPLEX Data

- `objval` : a pointer to a variable of type double where the objective value is stored.
- `bestobjval`: a pointer to a variable of type double where the best dual bound value is stored.
- `x`: An array to receive the values of the primal variables for the problem
- `pi`: An array to receive the values of the dual variables for each of the constraints
- `lpstat`: solution status of the most recent optimization performed on the CPLEX problem object
- `nodecount`: number of nodes used to solve a mixed integer problem.
- `cur_numrows`: number of rows
- `cur_numcols`: number of columns
- `coef_p`: a pointer to a double to contain the specified matrix coefficient.

Generic Functions

CPXopenCPLEX

CPXENVptr CPXopenCPLEX**(int * status)**

- The routine CPXopenCPLEX initializes a CPLEX environment.
- The routine CPXopenCPLEX must be the first CPLEX routine called.
- The routine returns a pointer to a CPLEX environment. This pointer is used as an argument to every other CPLEX routine .

Example:

- env = CPXopenCPLEX (&status);

CPXcreateprob

CPXLPptr CPXcreateprob(CPXCENVptr env, int * status, char const * probname)

- The routine CPXcreateprob creates a CPLEX problem object in the CPLEX environment

Example:

- lp = CPXcreateprob (env, &status, "myprob");

CPXchgobjsen

int CPXchgobjsen (CPXCENVptr env, CPXLPptr lp, int maxormin)

- The routine CPXchgobjsen changes the sense of the optimization for a problem, to maximization or minimization (CPX_MIN – CPX_MAX).

Example:

- status = CPXchgobjsen (env, lp, CPX_MAX);

CPXwriteprob

int CPXwriteprob (CPXCENVptr env, CPXCLPptr lp, char const * filename, char const * filetype)

- The routine CPXwriteprob writes a CPLEX problem object to a file in one of the formats (SAV – MPS – LP).

Example:

- status = CPXwriteprob (env, lp, "myprob.lp", NULL);

CPXreadcopyprob

int CPXreadcopyprob (CPXCENVptr env, CPXLPptr lp, const char * filename_str, const char * filetype_str)

- The routine CPXreadcopyprob reads an MPS, LP, or SAV file into an existing CPLEX problem object.

Example:

- status = CPXreadcopyprob (env, lp, "myprob.lp", NULL);

CPXfreeprob

int CPXfreeprob (CPXCENVptr env, CPXLPptr * lp_p)

- The routine CPXfreeprob removes the specified CPLEX problem object from the CPLEX environment and frees the associated memory used internally by CPLEX .

Example:

- status = CPXfreeprob (env, &lp);

CPXcloseCPLEX

int CPXcloseCPLEX **(CPXENVptr * env_p)**

- This routine frees all of the data structures associated with CPLEX.

Example:

- status = CPXcloseCPLEX (&env);

Model Construction Functions

CPXnewcols

int CPXnewcols (CPXCENVptr env, CPXLPptr lp, int ccnt, double const * obj, double const * lb, double const * ub, char const * xctype, char ** colname)

- The routine CPXnewcols adds empty columns to a specified CPLEX problem object.

Example:

- status = CPXnewcols (env, lp, ccnt, obj, lb, ub, xctype, NULL);

CPXnewrows

int CPXnewrows (CPXCENVptr env, CPXLPptr lp, int rcnt, double const * rhs, char const * sense, double const * rngval, char ** rowname)

- The routine CPXnewrows adds empty constraints to a specified CPLEX problem object.

Example:

- status = CPXnewrows (env, lp, rcnt, rhs, sense, NULL , NULL);

CPXaddcols

int CPXaddcols (CPXCENVptr env, CPXLPptr lp, int ccnt, int nzcnt, const double * obj, const int * cmatbeg, const int * cmatind, const double * cmatval, const double * lb, const double * ub, char ** colname)

- The routine CPXaddcols adds columns to a specified CPLEX problem object

Example:

- status = CPXaddcols (env, lp, ccnt, nzcnt, obj, cmatbeg,cmatind, cmatval, lb, ub, NULL);

CPXaddrows

int CPXaddrows (CPXCENVptr env, CPXLPptr lp, int ccnt, int rcnt, int nzcnt, double const * rhs, char const * sense, int const * rmatbeg, int const * rmatind, double const * rmatval, char ** colname, char ** rowname)

- The routine CPXaddrows adds constraints to a specified CPLEX problem object
- ccnt = 0 if only constraints are added and not columns at the same time

Example:

- status = CPXaddrows (env, lp, ccnt, rcnt, nzcnt, rhs, sense, rmatbeg, rmatind, rmatval, NULL , NULL);

Linear Programming Functions

CPXlpopt

int CPXlpopt (CPXCENVptr env, CPXLPptr lp)

- The routine CPXlpopt finds a solution to that problem using one of the CPLEX linear optimizers

Example:

- status = CPXlpopt (env, lp);

CPXgetpi

int CPXgetpi (CPXCENVptr env, CPXCLPptr lp, double * pi, int begin, int end)

- The routine CPXgetpi accesses the dual values for a range of the constraints of a linear or quadratic program

Example:

- status = CPXgetpi (env, lp, pi, 0, CPXgetnumrows(env,lp)-1);

CPXchgprobtype

int CPXchgprobtype **(CPXCENVptr env, CPXLPptr lp, int type)**

- The routine CPXchgprobtype changes the current problem to a related problem.

Example:

- status = CPXchgprobtype (env, lp, CPXPROB_LP);

Mixed Integer Linear Programming Functions

CPXmipopt

int CPXmipopt (CPXCENVptr env, CPXLPptr lp)

- the routine CPXmipopt is used to find a (optimal) solution to a problem.

Example:

- status = CPXmipopt (env, lp);

CPXgetbestobjval

int CPXgetbestobjval (CPXCENVptr env, CPXCLPptr lp, double * bestobjval_p)

- The routine CPXgetbestobjval accesses the currently best known bound of all the remaining open nodes in a branch-and-cut tree.

Example:

- status = CPXgetbestobjval (env, lp, &objval);

CPXgetnodecnt

int CPXgetnodecnt **(CPXCENVptr env, CPXCLPptr lp)**

- The routine CPXgetnodecnt accesses the number of nodes used to solve a mixed integer problem.

Example:

- nodecount = CPXgetnodecnt (env, lp);

Solution Access Functions

CPXgetobjval

int CPXgetobjval (CPXCENVptr env, CPXCLPptr lp, double * objval_p)

- The routine CPXgetobjval accesses the solution objective value.

Example:

- status = CPXgetobjval (env, lp, &objval);

CPXgetx

int CPXgetx (CPXCENVptr env, CPXCLPptr lp, double * x, int begin, int end)

- The routine CPXgetx accesses the solution values for a range of problem variables

Example:

- status = CPXgetx (env, lp, x, 0, CPXgetnumcols(env, lp)-1);

CPXgetstat

int CPXgetstat (CPXCENVptr env, CPXCLPptr lp)

- The routine CPXgetstat accesses the solution status of the problem after an LP, QP, QCP, or MIP optimization

Example:

- lpstat = CPXgetstat (env, lp);

Information Access Functions

CPXgetnumcols

int CPXgetnumcols **(CPXCENVptr env, CPXCLPptr lp)**

- The routine CPXgetnumcols returns the number of columns

Example:

- cur_numcols = CPXgetnumcols (env, lp);

CPXgetnumrows

int CPXgetnumrows **(CPXCENVptr env, CPXCLPptr lp)**

- The routine CPXgetnumrows returns the number of rows

Example:

- cur_numrows = CPXgetnumrows (env, lp);

CPXgetcoef

int CPXgetcoef (CPXCENVptr env, CPXCLPptr lp, int i, int j, double * coef_p)

- The routine CPXgetcoef accesses a single constraint matrix coefficient of a CPLEX problem object.
- *i* specifies the numeric index of the row.
- *j* specifies the numeric index of the column.

Example:

- status = CPXgetcoef (env, lp, 10, 20, &coef);

CPXgetub

int CPXgetub (CPXCENVptr env, CPXCLPptr lp, double * ub, int begin, int end)

- The routine CPXgetub accesses a range of upper bounds on the variables of a CPLEX problem object.

Example:

- status = CPXgetub (env, lp, ub, 0, cur_numcols-1);

CPXgetlb

int CPXgetlb (CPXCENVptr env, CPXCLPptr lp, double * lb, int begin, int end)

- The routine CPXgetlb accesses a range of lower bounds on the variables of a CPLEX problem object.

Example:

- status = CPXgetlb (env, lp, lb, 0, cur_numcols-1);

Parameters Functions

CPXsetintparam

int CPXsetintparam (CPXENVptr env, int whichparam, CPXINT newvalue)

- The routine CPXsetintparam sets the value of a CPLEX parameter of type CPXINT.
- The CPLEX Parameters Reference Manual provides a list of parameters with their types, options, and default values.

Example:

- status = CPXsetintparam (env, CPX_PARAM_SCRIND, CPX_ON);

CPXsetdblparam

int CPXsetdblparam (CPXENVptr env, int whichparam, double newvalue)

- The routine CPXsetdblparam sets the value of a CPLEX parameter of type double.
- The CPLEX Parameters Reference Manual provides a list of parameters with their types, options, and default values.

Example:

- status = CPXsetdblparam (env, CPX_PARAM_TILIM, 1000.0);

Quadratic Programming Functions

CPXchgqpcoef

int CPXchgqpcoef (CPXENVptr env, CPXLPptr lp, int i, int j, double newvalue)

- This routine changes the coefficient in the Q matrix of a quadratic problem corresponding to the variable pair (i,j) of the value newvalue.
- Note that if i is not equal to j, then both Q(i,j) and Q(j,i) are changed to newvalue.
- All the coefficient must be multiplied by 2 since the format is

$$\frac{1}{2}[\mathbf{x}^\top \mathbf{Q}\mathbf{x}]$$

Example:

- status = CPXchgqpcoef (env, lp, 10, 12, 82.5);

fabio.furini@dauphine.fr