C++/Cplex for Optimization Problems: a short tutorial and implementation

Rahman Khorramfar (rkhorra@ncsu.edu)

North Carolina State University

March 11, 2021

Part 1

Why C++/Cplex for optimization

C++

- It is generally (very) fast compared to interpreter-based languages like Python
- Many open-source solvers are developed in (C/C++)/Cplex. See, for example, COIN-OR, and COR@L
- Still many researchers use C++

Why C++/Cplex for optimization

C++

- It is generally (very) fast compared to interpreter-based languages like Python
- Many open-source solvers are developed in (C/C++)/Cplex. See, for example, COIN-OR, and COR@L
- Still many researchers use C++

Cplex

- More popular among OR practitioners
- Free for academic use
- Good documentations, but not the best
- More established than similar solvers such as Gurobi (another good one!)

Example 1: Transportation Problem

- $\mathcal{S}$: set of supplier — $\mathcal{D}$: set of customers (demand points)
- S: supply array — D: Demand array
- c_{sd}: cost of sending a unit from supplier s to demand point d
- X_{sd} (decision variable): amount of shipment from supplier s to demand point d

$$\min \sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{D}} c_{sd} X_{sd} \tag{1a}$$

$$s.t. \sum_{d \in \mathcal{D}} X_{sd} \leq S_s \qquad s \in \mathcal{S} \tag{1b}$$

$$\sum_{s \in \mathcal{S}} X_{sd} \geq D_d \qquad d \in \mathcal{D} \tag{1c}$$

$$X_{sd} \in \mathbb{Z}^{+} \qquad s \in \mathcal{S}, d \in \mathcal{D} \tag{1d}$$

Setup Cplex for C++ in Visual Studio (VS)

Make sure the Ilog Cplex is installed, and you have at least one ".cpp" file in your project:

1. Make sure the compiler is using the x64-bit platform

2. In the solution Explorer tab, click on the project name and select properties

3. Go to C/C++ general ->" additional include directories" -> paste (find) these two directories:
   - C: \Program Files\IBM\ILOG\CPLEX_Studio129\concert\include
   - C: \Program Files\IBM\ILOG\CPLEX_Studio129\cplex\include

Setup Cplex for C++ in Visual Studio (VS)

4. Go to C/C++ general ->"Preprocessors"->"Preprocessor Definitions" and add these commands:
   - WIN32
     _CONSOLE
     IL_STD
     _CRT_SECURE_NO_WARNINGS
   - or
   - NDEBUG
     _CONSOLE
     IL_STD
5. In the Project1 property page, select: "C/C++" - "code generation" - "runtime library", set to "multithreaded DLL (/MD)".

Setup Cplex for C++ in Visual Studio (VS)

6. In the Project1 property page, select: "Linker" - "Input" - "Additional Dependencies", and add these paths:
   - C:\Program Files\IBM\ILOG\CPLEX_Studio129\cplex\lib\x64_windows_vs2017\stat_mda\cplex129.lib
   - C:\Program Files\IBM\ILOG\CPLEX_Studio129\cplex\lib\x64_windows_vs2017\stat_mda\ilocplex.lib
   - C:\Program Files\IBM\ILOG\CPLEX_Studio129\concert\lib\x64_windows_vs2017\stat_mda\concert.lib

7. Add #include"ilcplex/ilocplex.h" to the ".cpp" file when needed

if you're using visual studio 2017 with cplex 12.8, you may encounter an error for which you can find a fix at:
https://www-01.ibm.com/support/docview.wss?uid=ibm10718671

Essential Cplex Commands

- IloEnv: to create a modeling environment
- IloModel: to create a model object
- IloNumVarArray: to define a one-dimensional decision variable
- IloRangeArray: to get the duals
- IloExpr: to define a variable to store a collection of terms
- IloMinimize: to add a minimization objective
- IloCplex: to create a cplex object and solve the model

Essential Cplex Object Methods

IloEnv env; IloModel Model(env); IloCplex cplex(Model);

- Model.add: add objective function and constraints
- cplex.solve(): solve the model
- cplex.cplex.getObjValue(): get objective function value
- cplex.getMIPRelativeGap(): get the gap
- cplex.getValue(IloNumVar): get the value of a decision variable
- cplex.getDual(IloRange): get dual value of a constraint
- cplex.getRay(IloNumArray,IloNumVarArray): get extreme rays

Essential Cplex Parameters

IloEnv env; IloModel Model(env); IloCplex cplex(Model);

- cplex.setParam(IloCplex::TiLim, 3600): set a time limit of 3600 seconds
- cplex.setParam(IloCplex::EpGap, 0.60): set the minimum required gap
- cplex.setOut(env.getNullStream()): turn off logging output on the console window
- cplex.exportModel("Name.lp"): print the model in a ".lp" format.
- cplex.getStatus(): status of the solution (optimal, unbounded, infeasible)

Example 1: Transportation Problem

Codes in the "TP0" to "TP4" folders, with varying automation level

Part 2:
Column Generation and Bender's Decomposition
March 11th, 2021

Example 2: Cutting-stock Problem

Cutting-stock Problem is the "Hello World!" of column generation algorithms

- L: the total length of each log
- $\mathcal{I}$: set of order lengths $\mathcal{P}$: set of patterns
- a_{ip}: number of pieces of length i cut in pattern p
- b_i: demand for order length i
- X_p: Number of logs cut using pattern p

$$\min \sum_{p \in \mathcal{P}} X_p \tag{2a}$$

$$s.t. \sum_{p \in \mathcal{P}} a_{ip} X_p \geq b_i \qquad i \in \mathcal{I} \tag{2b}$$

$$X_p \in \mathbb{Z}^+ \qquad p \in \mathcal{P} \tag{2c}$$

Example

Parameter	Value(s)
Log Length (L)	40
Order Length ($\mathcal{I}$)	[4, 2, 6, 7, 8, 12]
Demand (b_i)	[20, 41, 23, 12, 9, 34]

Possible Patterns:

$\mathcal{P}_1$=[10 0 0 0 0 0] $\implies a_{11} = 10, a_{12} = 0$

$\mathcal{P}_2$=[0 20 0 0 0 0]

$\mathcal{P}_3$=[1 1 2 0 0 0]

Example 2: Cutting-stock Problem

Delayed column generation idea:

- the number of all patterns (columns) can be huge. Also, optimal solution only uses a small subset of them.
- no need to generate the complete set of patterns
- with a small number of patterns and generating additional patterns as needed
- generate additional patterns by solving a *knapsack problem* in each iteration

Example 2: Cutting-stock Problem

Delayed column generation algorithm:

- create initial patterns
- solve the relaxed master problem and get the duals ω_i
- solve the following knapsack problem where a_i is now a variable that determines the number of pieces of length i cut in new pattern

$$\max\ z = \sum_{i \in \mathcal{I}} \omega_i a_i \tag{3a}$$

$$s.t.\ \sum_{i \in \mathcal{I}} R_i a_i \leq L \tag{3b}$$

$$a_i \in \mathbb{Z}^+ \qquad\qquad i \in \mathcal{I} \tag{3c}$$

- add the new pattern and stop when $1 - z^* \geq 0$

Codes in the "Cutting_Stock" folder

Example 3: Fixed Cost Transportation Problem (FCTP)

- everything same as the transportation problem, plus
- f_{sd}: fixed cost of sending items from supplier s to demand point d
- Y_{sd} (binary decision variable): 1, if a transshipment occurs between supplier s to demand point d, 0 otherwise

$$\min \sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{D}} (f_{sd} Y_{sd} + c_{sd} X_{sd}) \tag{4a}$$

$$s.t. \sum_{d \in \mathcal{D}} X_{sd} \leq S_s \qquad s \in \mathcal{S} \tag{4b}$$

$$\sum_{s \in \mathcal{S}} X_{sd} \geq D_d \qquad d \in \mathcal{D} \tag{4c}$$

$$X_{sd} \leq M Y_{sd} \qquad s \in \mathcal{S}, d \in \mathcal{D} \tag{4d}$$

$$X_{sd} \geq 0, Y_{sd} \in \{0, 1\} \qquad s \in \mathcal{S}, d \in \mathcal{D} \tag{4e}$$

Bender's Decomposition

Consider the following problem:

$$[MIP] \qquad \min_{x,y} \; f^T y + c^T x \tag{5a}$$

$$s.t. \; Ax + By \geq b \tag{5b}$$

$$y \in \mathcal{Y} \subseteq \mathbb{R}, x \geq 0 \tag{5c}$$

Consider the following problem:

$$[MIP] \qquad \min_{x,y}\ f^T y + c^T x \tag{5a}$$
$$s.t.\ Ax + By \geq b \tag{5b}$$
$$y \in \mathcal{Y} \subseteq \mathbb{R}, x \geq 0 \tag{5c}$$

Equivalently:

$$\min_{y}\ f^T y + V(y) \tag{6a}$$
$$s.t.\ y \in \mathcal{Y} \subseteq \mathbb{R} \tag{6b}$$

where $V(y)$ is:

$$[LP^P] \qquad \min_{x}\ c^T x \tag{7a}$$
$$s.t.\ Ax \geq b - B\hat{y} \quad [u] \tag{7b}$$
$$x \geq 0 \tag{7c}$$

The resulting model is a LP where its optimal solution can be obtained by solving its dual.

$$[LP^D] \qquad \min_u \; (b - By)^T u \tag{8a}$$
$$s.t. \; A^T u \leq c \tag{8b}$$
$$u \geq 0 \tag{8c}$$

- The feasible region of LP^D is independent of y
- Assume that for any given y, the primal problem (7) is feasible
- Assuming that the feasible region is not empty, the optimal solution for the original problem can be found by implicitly enumerating all the extreme points and rays
- Let $\hat{u}_j, j \in J$ be the set of all extreme point, and $\hat{u}_r, r \in R$ be set of extreme rays

Bender's Decomposition

Then the original problem becomes:

$$[MIP] \qquad \min_{y} \; z \tag{9a}$$

$$s.t. \; z \geq f^T y + (b - By)^T \hat{u}_j \qquad j \in J \tag{9b}$$

$$(b - By)^T \hat{u}_r \leq 0 \qquad r \in R \tag{9c}$$

$$y \in \mathcal{Y} \subseteq \mathbb{R} \tag{9d}$$

- Constraint (9b) is called optimality cuts because they ensure optimality of the dual problem (8)
- Constraint (9c) is called feasibility cuts because they ensure that (8) is not unbounded, thus the primal problem (7) is feasible which is what we assumed
- But enumerating all extreme points and rays is not practical

Bender's Decomposition

Start with $\hat{K} \subseteq K$ and $\hat{R} \subseteq R$ and form the *restricted master problem (RMP)*

$$
\begin{aligned}
[MIP] \qquad & \min_{y} \; z && && (10a)\\
& s.t. \; z \geq f^T y + (b - By)^T \hat{u}_j && j \in \hat{J} && (10b)\\
& (b - By)^T \hat{u}_r \leq 0 && r \in \hat{R} && (10c)\\
& y \in \mathcal{Y} \subseteq \mathbb{R} && && (10d)
\end{aligned}
$$

Get a new y variable $\hat{y}$ in each iteration and solve the subproblem:

$$
\begin{aligned}
[LP^D] \qquad & \min_{u} \; (b - B\hat{y})^T u && (11a)\\
& s.t. \; A^T u \leq c && (11b)\\
& u \geq 0 && (11c)
\end{aligned}
$$

Bender's Decomposition, Algorithm

- Initialize y, $LB = -\infty$, $UB = \infty$, k=0;
- while $(UB - LB > \epsilon$ do:
- k = k+1;
- solve the subproblem
- if unbounded then
  - get the extreme ray $\hat{u}_r$ and add the feasibility cut to the RMP
- if optimal then:
  - get extreme point $\hat{u}_j$ and add the optimality cut to the RMP
  - $UB = \min\{UB, f^T\hat{y} + (b - B\hat{y})^T\hat{u}_j\}$
- Solve the RMP and get z^k
- $LB = z^k$

Example 3: Fixed Cost Transportation Problem (FCTP)

Now recall the FCTP in its canonical form:

$$
\begin{aligned}
\min \quad & \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{D}} (f_{sd} Y_{sd} + c_{sd} X_{sd}) & & & \text{(12a)}\\
s.t. \quad & -\sum_{d\in\mathcal{D}} X_{sd} \geq -S_s & s \in \mathcal{S} & \quad [\alpha] & \text{(12b)}\\
& \sum_{s\in\mathcal{S}} X_{sd} \geq D_d & d \in \mathcal{D} & \quad [\gamma] & \text{(12c)}\\
& -X_{sd} \geq -MY_{sd} & s \in \mathcal{S}, d \in \mathcal{D} & \quad [\omega] & \text{(12d)}\\
& X_{sd} \geq 0,\ Y_{sd} \in \{0,1\} & s \in \mathcal{S}, d \in \mathcal{D} & & \text{(12e)}
\end{aligned}
$$

Example 3: Fixed Cost Transportation Problem (FCTP)

Then the Bender's subproblem becomes:

$$
\begin{aligned}
\max \quad & \sum_{s \in \mathcal{S}} -S_s \alpha_s + \sum_{d \in \mathcal{D}} D_d \gamma_d + \sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{D}} (-M \hat{y}_{sd}) \omega_{sd} & \\
s.t. \quad & -\alpha_s + \gamma_d - \omega_{sd} \leq c_{sd} & s \in \mathcal{S}, d \in \mathcal{D} \\
& \alpha_s, \gamma_d, \omega_{sd} \geq 0 & s \in \mathcal{S}, d \in \mathcal{D}
\end{aligned}
$$

Example 3: Fixed Cost Transportation Problem (FCTP)

And the Bender's RMP is:

$$\min_{y} \; z$$

$$s.t. \; z \geq \sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{D}} f_{sd}y_{sd} + \sum_{s\in\mathcal{S}}(-S_s)\hat{\alpha}_s^{(k)} + \sum_{d\in\mathcal{D}} D_d\gamma^{(k)} + \sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{D}}(-M\omega_{sd}^{(k)})y_{sd}$$

$$\sum_{s\in\mathcal{S}}(-S_s)\hat{\alpha}_s^{(k)} + \sum_{d\in\mathcal{D}} D_d\gamma^{(k)} + \sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{D}}(-M\omega_{sd}^{(k)})y_{sd} \leq 0$$

$$y_{sd} \in \{0,1\} \quad s \in \mathcal{S}, d \in \mathcal{D}$$

Codes in the "FCTP-Random Data" folder

What's more...

- More examples and techniques
- Methods based on branching
- Speed-up the code by compiling in release mode, or on Linux
- Cplex callable library
- Parallelism and multi-threading

Codes and the presentation will be available at my Github page here:
https://github.com/RahmanKhorramfar91/Cpp-Cplex-Tutorial

Thank You