=== Decision Management Community ===

Challenge Sep 2024 “Rental Boats”

Solution using CPLEX with API Concert in Java

by CHAGNEAU GUILLAUME guillaume.chagneau@m6.fr

The problem was defined at https://dmcommunity.org/challenge-sep-2024/:

Challenge Sep-2024

Rental Boats Solutions

BOAT RENTAL

Floataway tours has $420,000 that may be used to purchase new rental boats for hire during the summer. The boats can be purchased from two different manufacturers. Floataway tours would like to purchase at least 50 boats and would like to purchase the same number from Sleekboat as from Racer to maintain goodwill. Also, Floataway Tours wishes to have a total capacity of at least 200. Data about the boats is summarized below:

Boat	Manufacturer	Cost	Seating	Expected Daily Profit
Speedhawk	Sleekboat	$6,000	3	$70
Silverbird	Sleekboat	$7,000	5	$80
Catman	Racer	$5,000	2	$50
Classy	Racer	$9,000	6	$110

Your task is to create a decision model that will help Floataway to purchase boats while satisfying the above requirements and maximizing profit. Send your solutions to DecisionManagementCommunity@gmail.com.

Solution with CPlex with API Concert in Java :

```java
package challenge.dmcommunity.m202409;

import java.util.HashMap;
import java.util.Map;

import ilog.concert.IloException;
import ilog.concert.IloIntVar;
import ilog.cplex.IloCplex;
import lombok.extern.log4j.Log4j2;

@Log4j2
public class Solver {

    // Manufacturer object
    private record Manufacturer(
            String name) {}

    // Boat object
    private record Boat (
            String name,
            Manufacturer manufacturer,
            int cost,
            int seating,
            int profit) {}


    // Main method
    public static void main(String[] args) {
        new Solver().solve();
    }


    // Solve method
    private void solve() {

        // Create manufacturers
        var manufacturers = new Manufacturer[] {
                new Manufacturer("Sleekboat"),
                new Manufacturer("Racer")
        };

        // Create boats characteristics
        var boats = new Boat[] {
                new Boat("Speedhawk" , manufacturers[0], 6000, 3,  70),
                new Boat("Silverbird", manufacturers[0], 7000, 5,  80),
                new Boat("Catman"    , manufacturers[1], 5000, 2,  50),
                new Boat("Classy"    , manufacturers[1], 9000, 6, 110),
        };
```

```
        // Max budget
        final int BUDGET = 420_000;

        // Map of decision variables : One integer variable for every boat with the number of boats to
purchase
        Map<Boat, IloIntVar> decisionVars = new HashMap<>();


        try {
            // CPlex model
            var model = new IloCplex();

            // Decision vars
            for (var boat : boats) {
                decisionVars.put(boat, model.intVar(0, BUDGET / boat.cost));
            }

            // Constraints

            // 1) At least 50 boats
            var exprBoatsNumber = model.linearIntExpr();
            for (var boat : boats) {
                exprBoatsNumber.addTerm(decisionVars.get(boat), 1);
            }
            model.addGe(exprBoatsNumber, 50);

            // 2) Max budget
            var exprBudget = model.linearIntExpr();
            for (var boat : boats) {
                exprBudget.addTerm(decisionVars.get(boat), boat.cost);
            }
            model.addLe(exprBudget, BUDGET);

            // 3) At leat 200 seatings
            var exprSeatings = model.linearIntExpr();
            for (var boat : boats) {
                exprSeatings.addTerm(decisionVars.get(boat), boat.seating);
            }
            model.addGe(exprSeatings, 200);

            // 4) Same number of boats for each manufacturers
            var expr0 = model.linearIntExpr();
            for (var boat : boats) {
                if (boat.manufacturer == manufacturers[0]) {
                    expr0.addTerm(decisionVars.get(boat), 1);
                }
            }
```

```
            for (int i = 1; i<manufacturers.length; i++) {
                var expr = model.linearIntExpr();
                for (var boat : boats) {
                    if (boat.manufacturer == manufacturers[i]) {
                        expr.addTerm(decisionVars.get(boat), 1);
                    }
                }
                model.addEq(expr0, expr);
            }

            // Objective function
            var exprProfit = model.linearIntExpr();
            for (var boat : boats) {
                exprProfit.addTerm(decisionVars.get(boat), boat.profit);
            }
            model.addMaximize(exprProfit);

            // Solve model
            var solved = model.solve();


            // Write solution
            log.info("Solved : {}", solved);

            // Write numbers of boat per model
            for (var boat : boats) {
                log.info("Boat {} : {}", boat.name, model.getValue(decisionVars.get(boat)));
            }
            log.info("Budget : {}", model.getValue(exprBudget));
            log.info("Boats : {}", model.getValue(exprBoatsNumber));
            log.info("Seatings : {}", model.getValue(exprSeatings));
            log.info("Profit : {}", model.getValue(exprProfit));

            // End CPlex model
            model.close();

        } catch (IloException e) {
            log.error(e);
        }
    }

}
```

The result logs :

```
Default variable names x1, x2 ... being created.
Default row names c1, c2 ... being created.
Version identifier: 22.1.1.0 | 2022-11-27 | 9160aff4d
Tried aggregator 1 time.
```

MIP Presolve modified 14 coefficients.
Reduced MIP has 4 rows, 4 columns, and 12 nonzeros.
Reduced MIP has 0 binaries, 4 generals, 0 SOSs, and 0 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Found incumbent of value 4550.000000 after 0.00 sec. (0.02 ticks)
Tried aggregator 1 time.
Reduced MIP has 4 rows, 4 columns, and 12 nonzeros.
Reduced MIP has 0 binaries, 4 generals, 0 SOSs, and 0 indicators.
Presolve time = 0.00 sec. (0.00 ticks)
MIP emphasis: balance optimality and feasibility.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 12 threads.
Root relaxation solution time = 0.00 sec. (0.01 ticks)

Node	Nodes Left	Objective	IInf	Best Integer	Best Bound	Cuts/ ItCnt	Gap
*	0+ 0			4550.0000	14170.0000		211.43%
*	0 0	integral	0	5040.0000	5040.0000	3	0.00%

Elapsed time = 0.00 sec. (0.03 ticks, tree = 0.00 MB, solutions = 2)

Root node processing (before b&c):
  Real time   =   0.00 sec. (0.03 ticks)
Parallel b&c, 12 threads:
  Real time   =   0.00 sec. (0.00 ticks)
  Sync time (average)   =   0.00 sec.
  Wait time (average)   =   0.00 sec.

Total (root+branch&cut) =   0.00 sec. (0.03 ticks)
Solved : true
Boat Speedhawk : 28.0
Boat Silverbird : -0.0
Boat Catman : -0.0
Boat Classy : 28.0
Budget : 420000.0
Boats : 56.0
Seatings : 252.0
Profit : 5040.0

Best regards,

GUILLAUME CHAGNEAU

SENIOR IT PROJECT MANAGER

Métropole Télévision

56, AV. CHARLES DE GAULLE 92200 NEUILLY-SUR-SEINE

TEL. +33 1 41 92 79 56 MOBILE. +33 7 88 47 26 97 dsi-pub-tv@m6.fr