

Stigler's Diet Problem

Solution with OpenRules Rule Solver (without Java)

The problem is offered as a Decision Management Community [Challenge October-2026](#):

Stigler's Diet Problem is one of the classical problems in Operational Research. The problem finds its origin in the work of Nobel laureate George Stigler (1944), who posed this as a mathematical exercise, not as eating recommendations:

For a moderately active man weighing 154 pounds, how much of each of 77 foods should be eaten on a daily basis so that the man's intake of nine nutrients will be at least equal to the [recommended dietary allowances](#) in 1943, with the cost of the diet being minimal?

You can find more details, the nutrients and commodities lists, along with a possible solution, at the [OR-Tools website](#).

Glossary

Glossary glossary				
Variables	Business Concept	Attribute	Type	Used As
Nutrients	DietProblem	nutrients	Nutrient[]	in
Foods		foods	Food[]	in
TotalAnnualCost		totalAnnualCost	double	out
NutrientName	Nutrient	name	String	in
NutrientId		id	int	in
DailyIntake		dailyIntake	double	in
FoodId	Food	foodId	String	in
Commodity		commodity	String	in
Units		units	String	in
Price		price	double	in
NutrientQuantities		nutrientQuantities	double[]	in
AnnualCost		annualCost	double	out
NumberOfNutrients	:=	\${Nutrients}.length	int	
NutrientQuantity		\${NutrientQuantities}[\${NutrientId}-1]	double	

Test Data

We copied the OR-Tools data to Excel and added two lines to the headers.

DecisionData Nutrient nutrients		
NutrientId	NutrientName	DailyIntake
1	Calories (calories)	3
2	Protein (grams)	70
3	Calcium (grams)	0.8
4	Iron (milligrams)	12
5	Vitamin A (IU)	5
6	Thiamine (Vitamin B1) (milligrams)	1.8
7	Riboflavin (Vitamin B2) (milligrams)	2.7
8	Niacin (milligrams)	18
9	Ascorbic Acid (Vitamin C) (milligrams)	75

DecisionData Food foods												
FoodId	Commodity	Units	Price	NutrientQuantities								
F1	Wheat Flour (Enriched)	10 lb.	36	45	1411	2	365	0	55	33	441	0
F2	Macaroni	1 lb.	14.1	12	418	0.7	54	0	3.2	1.9	68	0
F3	Wheat Cereal (Enriched)	28 oz.	24.2	12	377	14	175	0	14	8.8	114	0
F4	Corn Flakes	8 oz.	7.1	11	252	0.1	56	0	14	2.3	68	0
F5	Corn Meal	1 lb.	4.6	36	897	1.7	99	30.9	17	7.9	106	0
F6	Hominy Grits	24 oz.	8.5	29	680	0.8	80	0	11	1.6	110	0
F7	Rice	1 lb.	7.5	21	460	0.6	41	0	2	4.8	60	0
F8	Rolled Oats	1 lb.	7.1	25	907	5.1	341	0	37	8.9	64	0
F9	White Bread (Enriched)	1 lb.	7.9	15	488	2.5	115	0	14	8.5	126	0
F10	Whole Wheat Bread	1 lb.	9.1	12	484	2.7	125	0	14	6.4	160	0
F11	Rye Bread	1 lb.	9.1	12	439	1.1	82	0	9.9	3	66	0
F12	Pound Cake	1 lb.	24.8	8	130	0.4	31	18.9	2.8	3	17	0
F13	Soda Crackers	1 lb.	15.1	13	288	0.5	50	0	0	0	0	0
F14	Milk	1 qt.	11	6.1	310	11	18	16.8	4	16	7	177
F15	Evaporated Milk (can)	14.5 oz.	6.7	8.4	422	15	9	26	3	24	11	60
F16	Butter	1 lb.	30.8	11	9	0.2	3	44.2	0	0.2	2	0
F17	Oleomargarine	1 lb.	16.1	21	17	0.6	6	55.8	0.2	0	0	0

...

DecisionTest testCases		
#	Define	Define
Test ID	Nutrients	Foods
1	nutrients	foods

Business Logic

We implemented using the newest version of [Rule Solver](#) without any [Java code](#) using only OpenRules decision tables.

Problem Definition

We define our problem in three steps described in the table “Define”:

Decision Define
Execute
Decision Tables
CreateVariables
PostConstraints
SetObjective

Step 1. Create Main Constraint Variables

Decision CreateVariables [for each Food in Foods]		
SolverCreateReal		
Variable Name	Domain	Array Name
{{FoodId}}	0..1000000	"AllFoodVariables"

For every food in the array “Foods” this table creates unknown decision variables defined from 0 to a large number (we selected 1,000,000) and adds it to the array “AllFoodVariables” that will be used to define the objective “TotalAnnualCost”.

Step 2. Post Daily Intake Constraints

Decision PostConstraints [for each Nutrient in Nutrients]
Execute
Tables
CreateConsumedNutrientArray
PostDailyIntakeConstraints

For each nutrient in the array “Nutrients” this table will create an array of constrained variables “Consumed{{NutrientId}}” and then post a constraint

$$Sum (Consumed\{\{NutrientId\}\}) \geq (\{\{DailyIntake\ of\ Nutrient\}\} * 365.0)$$

Decision CreateConsumedNutrientArray [for each Food in Foods]		
SolverDefineReal		
Variable Name	Variable Expression	Array Name
Consumed_{{NutrientId}}_{{FoodId}}	{{FoodId}} * {{NutrientQuantity}}	Consumed{{NutrientId}}

Decision PostDailyIntakeConstraints	
SolverPost	
Variable Name	
Sum (Consumed{{NutrientId}}) >= ({{DailyIntake of Nutrient}} * 365.0)	

Step 3. Set Optimization Objective

Decision SetObjective		
SolverDefineReal		SolverSetObjectiveReal
Variable Name	Expression	Objective
"TotalAnnualCost"	Sum("AllFoodVariables")	"TotalAnnualCost"

This table defines a constrained variable “*TotalAnnualCost*” as *Sum("AllFoodVariables")* and sets it as the optimization objective.

Problem Resolution

We use the predefined method “SolverMinimize” calling it from the standard table “Solve”:

Decision Solve	
Execute	
Actions	
SolverMinimize	
AssignSolution	

After an optimal solution is found, this table will invoke the following table to assign the found results to the output decision variables "AnnualCost" for each food in the array “Foods”:

Decision AssignSolution [for each Food in Foods]					
SolverIfSolutionReal			SolverAssignSolutionReal		Print
Variable Name	Oper	Value	Business Variable Name	Solver Variable Name	Text
{{FoodId}}	>	0	"AnnualCost"	{{FoodId}}	{{Commodity}} = \$ {{AnnualCost}}

Applied Linear Solvers

We tried different linear/MIP solvers by changing the property “SOLVER” inside the “test.bat”:

```
@echo off
cd %~dp0
set SOLVER=Scip
rem set SOLVER=CLP
rem set SOLVER=ojAlgo
call mvn clean test -e
```

They all produced good results:

```
TotalAnnualCost[39.66173222461947]
*** Execution Profile ***
Execution time: 1089 msec
SolverSaveSolution
Wheat Flour (Enriched) = $ 10.7744567645
Liver (Beef) = $ 0.6907836618
Cabbage = $ 4.0932688874
Spinach = $ 1.8277959785
Navy Beans, Dried = $ 22.2754269323
'testCases-1' succeeded. Elapsed time 1590.03 ms
```