LAAI Module I - Sustainability KPIs Optimization

Facundo Nicolas Maidana
Riccardo Spolaor
Stefano Ciapponi

1 INTRODUCTION

1.1 Goal of the Project

The project consists in building a constraint programming model which aim is to maximize a series of environmental sustainability KPIs' scores of a company, while guaranteeing that specific constraints on production are satisfied.

1.2 What is a KPI?

KPI stands for *Key Performance Indicator* and it is a quantifiable measure of performance over time related to a specific business, marketing or sustainability objective.

To fulfill the aim of the project eight environmental sustainability related KPIs have been selected from a list provided by the study publication *Sustainability in Sourcing & Procurement* [13]. The chosen KPIs are listed below and divided according to their macro categories.

Raw materials KPIs

- CO2 (Green House Gases) emissions during production:
  - Relevance value: 4.31/5;
  - This KPI score is inversely proportional to the amount of CO2 emitted during the fiscal year;
- Environmental certificates and labels:
  - Relevance value: 4.20/5;
  - Environmental labels and certifications obtained by a company guarantees its business is sustainable and therefore an increase of the score of the KPI.
- Percentage of recyclable content:
  - Relevance value: 3.91/5;
  - Each item the industry can produce contains a certain amount of recyclable content, thus the score of this KPI increases as a higher ratio of recyclable items are produced.

Operating resources & equipment KPIs

- Water efficiency of equipment:
  - Relevance value: 4.06/5;
  - The score of the KPI increases when less liters of water are consumed during production.
- Energy efficiency of equipment:
  - Relevance value: 4.42/5;
  - The KPI score is proportional to the energy efficiency of the equipment used during production.
- Reject rate of equipment:
  - Relevance value: 3.68/5;
  - The KPI score worsens as the ratio of equipment that has been disposed by a company increases.

Logistics & transportation KPIs

- Transportation mode (Air, road, water, or rail):
  - Relevance value: 4.26/5;
  - Some transportation means are considered more sustainable and better in terms of efficiency than others. The choices on which and how many of them are selected to deliver the items influences the score of this KPI.
- Transport-unit capacity utilization:
  - Relevance value: 4.16/5;
  - The transportation contraptions used to deliver the items should be filled as much their capacity allows in order to increase this KPI score.

2 PROGRAM STRUCTURE

To model the given problem and satisfy its goal we implemented a `Minizinc` constraint satisfaction model which aims at maximizing each one of the selected KPIs score in a realistic scenario. The solver selected to compile and run the program is *Geocode 6.3.0* since it is general purpose and it could allow us to perform specific operations like float variable divisions.

The score optimization is subject to numerous constraints related to production. For instance the annual revenue obtained through the sold items should be above a certain minimum imposed threshold and a fixed budget is assigned to production.

The program has been implemented in such a way that it can be customized for different types of companies and production goals. In particular the user is asked to input a series of business related parameters through a `.dzn` data file, which are then considered for satisfying the constraints and in the concurring KPIs maximization task. Other static parameters defined as constants are present in the program.

The program contains a series of variables which values are assigned by the solver at each optimization iteration in such a way that guarantees constraint satisfaction and goal maximization.

Since modeling each aspect of the business in detail would have been too computationally expensive, we decided to add some simplifications to our model:

- Each piece of equipment is supposed to be able to produce any type of item the industry can manufacture and they are all able to work in parallel with each other;
- Some certification costs have been approximated since they could vary depending on the type of company;
- Items are all supposed to be delivered to the same fixed location when logistic procedures are involved;
- The items' volumes are considered equivalent when their delivery is concerned;
- Each category of mean of transport is considered to be able to hold the same number of items (e.g., all cargo planes are the same size);

- Most parameter and variable types have been set as integers for computational efficiency, although in some cases the use of floating points would have been more expressive.

2.1 Input parameters

The input parameters defined by the user are necessary to define the company information data and its current year production goals.

Some of them are mandatory like items that can be produced, whereas others are not. Depending on the chosen optional parameters some KPIs scores are not considered for the optimization. For instance if the business model doesn't make use of logistics, all logistics related constraints and variables are not computed.

They are divided into six main sections (Module 1).

Company goals and previous data:
Mandatory parameters that are used in different constraints and most of them represent resources that each KPI optimization is concurring for. They include:

- Available production budget of the company in thousands of dollars ($);
- Minimum production earnings to achieve in thousands of dollars ($);
- The company's previous fiscal year CO2 emissions in tons, water consumption in KLiters and the energy consumption in KWatts;
- Maximum available production time in hours;

Parameters regarding the items produced by the company:
Mandatory parameters expressing:

- Which items are produced by the company;
- Cost to produce each item, their selling prices, the time and tons of CO2 emissions.

Parameters giving information about the equipment:
These parameters are mandatory and define information about the equipment utilized by the company or that is available to be purchased. They include:

- Number of equipment pieces the company owns for each one of their types;
- Energy (KWatt) and water (KLiters) consumed by each one of them per hour along with the tons/h of CO2 they emit;
- Their purchase costs and the earnings obtained by disposing of them.
- Maximum pieces of equipment the company may own and the part of the budget assigned to purchase new ones.

Parameter regarding the environmental certifications that the company owns:
This parameter describes which environmental certifications the company already owns before production starts and it is needed to optimize the *Environmental certificates and labels* KPI score.

Parameter on the recyclable composition of each product:
This parameter is not mandatory and it is used to optimize the *Percentage of recyclable content* KPI score if provided.

Parameter giving information about the logistics:
Non mandatory parameter which optimize the logistics related KPI scores if inputed. It describes whether a logistic department is owned by the business or not.

2.2 Constants

Constants are parameters of the model which are intended to define static information that cannot be expressed by the user. Along with the input parameters some are used to satisfy constraints and in assigning variables values. Once again the constant parameters will be divided in different sections for clarity (Module 2).

KPIs related constants:
These parameters include information related to the KPIs, namely:

- Set of KPIs considered for the model as expressed in section 1.2;
- Sustainability relevance score of each KPI expressed on a decimal value between 0 and 5;
- KPIs considered for the optimization according to the user preference expressed in the input parameters.

Optimization related constants:
These parameters are used in the optimization process as illustrated in section 2.5 and express the target reduction rate of some environmental related issues according to the United Nations' *Sustainable Development Goals (SGDs)* for 2030. More precisely they include:

- Target reduction rate of emissions to obtain by 2030, that should be equivalent to 45% according to the 13^{th} SGD [7];
- Target reduction rate of energy consumption to reach by 2030, that is set to 50% as *The Sustainable Development Goals Report 2021 - Goal 7* expresses in section 7.3 [12];
- Target reduction rate of water consumption to obtain by 2030, that is set to 50% as *The Sustainable Development Goals Report 2021 - Extended Report - Goal 6* expresses in section 6.4 that the water consumption efficiency should be substantially improved by 2030 [11];

Certifications related constants:
These parameters express which environmental sustainability certifications are available for the company to own. We opted to include three certifications from the ones available in literature. Each one focuses on different environmental related aspects of the company:

- ***Brands for Good CarbonCare©***[1] is an international certification label that recognizes the efforts of a business in reducing greenhouse emissions. The certification cost varies based on the company revenues. Criteria to obtain it were based on the sister *CarbonCare©*[2] certification that express the company should reduce its greenhouse emissions by at least 5%.
- ***Global Recycled Standard (GRS)***[3] is an international certification label emitted by *Textile Exchange* that is assigned to companies that increase the use of recycled materials. The certification cost is estimated to be around $7000-$9,000. Criteria to obtain it were not expressed so we decided that the total percentage composition of recycled products should be above 50% to obtain it.
- ***WaterSense***[9] labeled materials are generally more efficient than other in terms of water consumption. As an achievement criteria to obtain it we decided that water consumption should be decreased by 20%. As a cost was not indicated we decided to assign arbitrarily one of $4000.

Logistic related constants:
These parameters concern the logistic information that the program

will use if the company has to physically deliver its products. In order to model them we followed the instructions given by the internet article *Types of Transportation in Logistics: Which Is Right for You?* [8]. They express:

- Transportation means available to the company, namely: planes; ships; trucks and trains.
- A decimal score between 0 and 1 of the quality of transportation of each vehicle alongside the weight related to it. Plane travels have the best quality, while truck ones the worst, due to the fact that not much control is assigned in handling the products.
- A decimal score from 0 to 1 related to the quality of fuel emissions of each vehicle with a weight connected to it. Trains have a higher score due to the fact that they are in general electric vehicles and emit less pollution, while planes have the lowest because of their scarce fuel efficiency.
- Transportation capacity measured in units of products of each mean.
- Their cost in thousands of dollars per travel.

2.3 Variables

Variables are unknown values that the program assigns dynamically in order to optimize the KPIs total score. They are subjected to a set of constraints as explained in section 2.4 and they are generally strictly dependent to the user input parameters. Some of them use functions to compute their values (available in Module 5) which have been defined to make the code more readable and cleaner (Module 3).

They are divided in the following sections:

KPIs related Variables:

- An array of decimal values which represents for each KPI its reached score in a range between 0 and its relevance.
  - Their value are used in order to compute a global KPI score.
- The KPI global score in terms of percentage.
  - The meaning of this score value is illustrated in Table 1 according to the publication *Normalization in sustainability assessment: Methods and implications* [14].
- CO2 emissions produced by both item manufacturing and the equipment usage.

Score	Sustainability level
0%	Extremely unsustainable
25%	Not sustainable but not as severely as in the previous case
50%	Discrete level of sustainability, but still far from target
75%	Satisfactory level of sustainability, yet not on target
100%	Fully sustainable

Table 1: Sustainability scores

Products related Variables:

- Array containing values of the number of singular items the program expects the business to manufacture.
- Total earnings obtained by selling the produced items.
- Total time requested to obtain the expected number of items.
- Total energy consumption in KWatts emitted during production by the equipment to make the items.
- Total water consumption during production in KLiters.

Equipment related Variables:

- Array expressing the equipment owned by the business for production after initial decisions.
  - The program may decide that the company should buy more efficient pieces of equipment to increase its environmental sustainability and to dispose of some obsolete ones in spite of a lower *Reject rate of equipment* KPI score.
- Actual usage in hours during production of each piece of equipment.
- Total energy consumption after production.
- Total water consumption after production.

Certifications related Variables:

- Array of boolean variables representing the certifications the business already had and that managed to obtain after production is completed.

Logistic Related Variables:

- An integer array illustrating the number of deliveries each mean of transport carried.
- An integer array representing the total number of items assigned to each mean of transport.

Due to their highly descriptive capabilities, we have decided to print most variables as the program's output alongside the KPIs score evaluation in order to express information about the production process.

2.4 Constraints

This section contains a brief explanation of the model constraints. Like what happens for variables the computation of some of them require the call of functions available in Module 5. They are defined as follows (Module 4):

Constraints on input parameters:
They check whether the logistics and recyclability related parameter inputs are correct if provided by the user.

Constraints on production:
These constraints aim at guaranteeing that some production standards are respected. Some of them use resources that different KPIs compete for.

- The total costs for production and buying certifications must not exceed the budget;
- The equipment costs must not exceed the equipment budget;
- The total earnings obtained by selling the products must be greater than a minimum provided threshold;
- The business can't have more equipment than a maximum imposed threshold;
- Each piece of equipment cannot be used more than its maximum usage limit;

- Total usage of the equipment must be strictly equal to the total time needed to manufacture all the items.
- Total production time must be less or equal than the available working hours.

Constraints on the certifications:
As specified in section 2.2, data related to each certification has been fetched from its certification authority website. They are subjected to the following constraints:

- Certifications already owned by the company are kept till the end of the fiscal year.
- To be eligible to own each certification each company must already hold one or met their assignment criteria.

Constraints on the logistic:
When a company decides to compute the KPIs' scores related to logistics the following constraints must be respected:

- Units of items that are delivered cannot exceed the total capacity of the used means of transport.
- Each product has to be delivered and assigned to a vehicle.

2.5 Optimization process

Multi-objective optimization is an area of multiple criteria decision making that is concerned with mathematical optimization problems involving more than one objective function to be optimized simultaneously.

Multi-objective optimization can be obtained through many techniques such as *Lexicographic Methods* that impose an order to objective functions prioritizing higher importance.

Maximizing a global KPI value for an industry consists in maximizing KPI score sub-objectives (Module 6), consequently multi-objective optimization was the best way to define our model solution.

Our team tried different approaches to solve the problem, but only one of them was able to work and compute a solution efficiently. More precisely the following techniques have been tried:

- ***OptiMathSAT [15]:*** is a tool used to solve Multi-objective Optimization problems developed by the *University of Trento*. We tried using the tool as a solver of our model, but we couldn't manage to make it work as it gave us errors when running the compiled `FlatZinc` code
- ***Yuck [10]:*** a Flatzinc interpreter that integrates with the Minizinc toolchain which approaches problem solving using local search. Given a series of tries we couldn't make it work because we noticed some problems with internal *Scala* library files.
- ***MiniSearch [5]:*** a language for specifying meta-search in a `MiniZinc` model. We tried using it to solve the problem through a *Lexicographic Branch and Bound Search*, but we noticed that the program would stop while flattening the code in `FlatZinc` format.
- ***Hybrid method using a weighted version of the Lexicographic method:*** each singular KPI score value has been scaled with respect to the relevance of the previous one (expressed in decreasing order) while also being weighted by its own relevance. This method revealed itself to be a working solution.

The Main project module (Module 6) contains the objective function we are trying to maximize. It is split into eight sub-objective functions, each one trying to optimize a single KPI score.

Every sub-objective function $s(k)$ consists in firstly normalizing a KPI (k) score between 0 and 1 and then weighting it with the respective KPI relevance value ($r(k)$). The normalizations have been applied following the rules of the publication [14].

CO2 emissions during production, *energy efficiency of equipment* and *water efficiency of equipment* KPI scores are normalized considering values taken from the user's input concerning the previous year sustainability data and a reduction rate which is supposed to be reached before 2030. They are computed through a `Min-Max` normalization technique. As an upper bound it uses the previous year sustainability data, while as a lower bound it computes a target sustainability goal. The result is then subtracted from 1 in order to give a higher score if the sustainability results are closer to the target and is force to be among 0 and 1 through a function $force_{[0,1]}$ in order to handle overflowing values.

The sub-objective functions have been implemented as follows:

Energy efficiency of equipment score
The score of this KPI has been normalized through the `Min-Max` normalization technique described above considering the current year energy emissions (em_{energy}). As an upper bound (ub) it uses the previous year energy emissions, while as a lower bound (lb) it computes the previous year energy emissions reduced by a percentage obtained by the target energy reduction rate by 2030 divided by the number of years left to reach this target.

$$s(k) = r(k) * force_{[0,1]}(1 - \frac{em_{energy} - lb_{energy}}{ub_{energy} - lb_{energy}})$$

CO2 emissions during production score
Similarly to the previous case this KPI score is computed normalizing the total CO2 emissions (em_{CO2}) using the previous year's emissions as an upper bound (ub) and the reduction rate expected up to 2030 to calculate a lower bound (lb).

$$s(k) = r(k) * force_{[0,1]}(1 - \frac{em_{CO2} - lb_{CO2}}{ub_{CO2} - lb_{CO2}})$$

Transportation mode score
This score value is computed solely when logistics are part of the business model and they are computed through the sum of the number of means of transport used for each vehicle t multiplied by their quality ($T_q(t)$) and energy ($T_e(t)$) efficiency scores divided by the total number of used vehicles (T_n) multiplied by the maximum efficiency and energy obtainable values (q_{max}, e_{max}).

$$s(k) = r(k) * \frac{\sum\limits_{t \in Transports} T_n(t) * T_q(t) * T_e(t)}{\sum\limits_{t \in Transports} T_n(t) * q_{max} * e_{max}}$$

Environmental certificates and labels score
Computes the certification score through a ratio between the number of certifications achieved ($C_{achieved}$) and the total number of achievable certifications.

$$s(k) = r(k) * \frac{C_{achieved}}{|Certifications|}$$

Transport-unit capacity utilization score
If Logistics are part of the Business model it is computed by normalizing the total number of items assigned to the mean of transport ($T_{assigned}$) divided by the total capacity in units of items of the used vehicles (T_{cap}):

$$s(k) = r(k) * \frac{T_{assigned}}{T_{cap}}$$

Water efficiency of equipment score
Computed in an analogous way to the *energy efficiency of equipment* and *CO2 emissions during production score*, although using values concerning water consumption.

$$s(k) = r(k) * force_{[0,1]}(1 - \frac{em_{water} - lb_{water}}{ub_{water} - lb_{water}})$$

Percentage of recyclable content score
Computed just when the recyclable percentage composition of the product i is defined by multiplying the number of produced items ($I_{produced}(i)$) for their recyclable composition ($I_{rec}(i)$) and normalized dividing by it the total number of items.

$$s(k) = r(k) * \sum_{i \in Items} \frac{I_{rec}(i) * I_{produced}(i)}{|Items|}$$

Reject rate of equipment score
Computed dividing the number of disposed equipment $E_{disposed}$ divided by the number of equipment the business had at the start of production E_{start} and subtracting it from 1 (maximum value).

$$s(k) = r(k) * (1 - \frac{|E_{disposed}|}{|E_{start}|})$$

Objective maximization function
The objective function we defined tries to maximize the KPI scores according to the total sum of the sub-objective function values defined above. Each score is scaled by the maximum relevance value of the following one in a lexicographic decreasing order. $kpi_i(k)$ is a function that returns the k^{th} KPI according to the imposed order.

$$\max \sum_{k=1}^{|KPIs|-1} (s(kpi_i(k)) * r(kpi_i(k+1))) + s(kpi_i(|KPIs|))$$

As an output the program expresses the total KPIs score in percentage (as interpreted according to Table 1) along with the single scores of each KPI and the comparison about the company previous and expected current year production information.

3 USE CASE

For testing the behaviour of the algorithm, a use case was implemented. The choice was to simulate the business model of a paper manufacturing company. The paper industry is the fourth most energy-intensive and polluting in Europe [4], so the optimization of its sustainability can play a big role on environment.

To improve the sustainability, a company can decide the type of items that will be produced. Our example company can produce different types of paper: virgin (made from pure cellulose), recycled (produced starting from other paper) and cardboard (similar to recycled one). Manufacturing one tonne of 100% recycled paper emits 60% less CO2 than paper produced from virgin fibres [6].

Producing recycled paper and cardboard naturally lowers the carbon footprint of the company, on the other hand it consists in a more complex production process compared to manufacturing brand new (virgin) paper. Those extra steps imply extra time needed for production and more expenses.

The company has just one unit of machinery at the start of the fiscal year and the possibility to buy new ones. There are three different types of machinery, each one with different properties concerning energy, water consumption, carbon footprint and costs.

Business and production constraints are added to the model, in particular regarding the amount of money that has to be achieved by selling the items and the maximum time available for production (one year). More detailed information about the company parameters are given in tables 2, 3 and 4.

The aim of our model is to produce a realistic output which represents a business scenario in which a higher sustainability score is reached.

An input data file has been prepared to handle the parametrization of the model and an output solution has been generated. The achieved results are interesting and the maximum reached **KPI score** percentage of optimization is **72.5%**, which can be considered a satisfactory level of sustainability according to table 1.

In order to reduce the amount of CO2 produced and the energy and water consumed, the solution proposed by the model was to manufacture a greater quantity of recycled paper rather than virgin, more specifically **45 units of paper against 415 units of recycled paper and no cardboard**. The production was not fully moved into recycled paper due to time and earning constraints (the recycled process requires more time to be achieved and has a higher cost, that is hence interpreted as a lower amount of earnings). All the **KPIs scores linked to emissions and consumption were fully optimized** by the model.

In regard to the machinery adopted, the model decided to dispose the sole machine it had before production started (**Machine 3**), in order to buy a new one (**Machine 1**) that has a better energy and water efficiency, and lower C02 emissions per hour. This choice helped optimizing the KPI scores mentioned above at the cost of **setting the *reject rate of equipment* score to zero**.

The **certifications** earned by the company remained **unvaried**. This result is probably connected to the fact that the production and business constraint have a huge impact on the conditions required to achieve them.

Regarding the logistic aspect of the company, it is evident how the model tried to optimize the sustainability even in this field. The items were mostly transported through ships (**260 items delivered with two ships**) and trains (**200 transported items with 4 vehicles**), which are the most sustainable means among the ones provided, although they are not the cheapest. The **total capacity** of the vehicles was **fully exploited**, thus completely optimizing the *transport-unit capacity utilization* score.

The resulting model output (Table 5) is an admissible solution, which can be interpreted by salesman of the company to adopt a more sustainable business plan for the next year, without any loss in production.

Table 2: Company goals and data user case input

Budget	$1,1 M
Minimum earnings	$1,8 M
Previous year CO2 emissions	2700 tons
Previous year energy consumption	2200 KWatts
Previous year water consumption	2200 KLiters
Available time	1 year (1760 h)
Years to reach the reduction goals	5
Starting certifications	Good CarbonCare
Uses logistics	Yes

Table 3: Items user case input

	Paper	**Recycled Paper**	**Cardboard**
Item cost	$1 k	$2 k	$1 k
Item selling price	$4 k	$4 k	$2 k
Item CO2 emissions	10 tons	1 ton	2 tons
Item extra energy consumption	5 KWatt	1 KWatt	1 KWatt
Item extra water consumption	5 KLiters	1 KLiters	1 KLiters
Item time to make	2 h	3 h	3 h
Item recyclable composition	100%	85%	40%

Table 4: Equipment user case input

	Machine 1	**Machine 2**	**Machine 3**
Yearly maximum usage	1400 h	1200 h	850 h
Co2 emission per hour	1 tons	3 tons	3 tons
Energy consumption per hour	1 KWatt	2 KWatts $	3 KWatts
Water consumption per hour	1 KLiter	2 KLiters	3 KLiters
Prices	$20 k	$25 k	$20 k
Disposal prices	$15 k	$12 k	$10 k
starting equipment	0	0	1
Maximum number of machines = 1			
Equipment budget = $10 k			

Table 5: User case output

Total KPI score	72.5%
Raw materials KPIs score	CO2 emissions during production: 4.31/4.31; Environmental certificates and labels: 1.4/4.2; Percentage of recyclable content: 3.38/3.91
Operating resources & equipment KPIs score	Water efficiency of equipment: 4.06/4.06; Energy efficiency of equipment: 4.42/4.42; Reject rate of equipment: 0.0/3.68
Logistics & transportation KPIs score	Transportation mode: 2.22/4.26; Transport-unit capacity utilization: 4.16/4.16
Company results	Earnings: $1,84 M; Production time: 1335 h
Company production	Paper: 45 units; Recycled paper: 415 units; Cardboard: 0 units.
Company final equipment and usage	One unit of Machine 1 used for 1335 h
Company earned certifications	Good CarbonCare
Company logistics information	Use of 2 ships to deliver 260 items and of 4 trains to deliver 200 items

MODULE 1: parameters.mzn

```
include "constants.mzn" ;
/* ================================ COMPANY GOALS AND PREVIOUS DATA ================================== */
/* Integer expressing the budget of company measured in thousands of dollars ($). */
par 0..1000000: budget;
/* Integer expressing the minimum earning to achieve from selling items measured in thousands of dollars
   ($). */
par 0..1000000: minimum_earnings;
/* Integer expressing the tons of C02 produced the previous year. */
par 0..100000000: previous_year_co2_emissions;
/* Integer expressing the KWatt of energy produced the previous year. */
par 0..100000000: previous_year_energy_consumption;
/* Integer expressing the KLiters of water consumed the previous year. */
par 0..100000000: previous_year_water_consumption;
/* Integer expressing the maximum available production time in hours. */
par 0..10000: available_time_in_hours;
/* Integer representing the years to reach the reduction goals */
par 1..10: years_to_reach_the_reduction_goals;
/* ================================== PRODUCTS DATA (MANDATORY) ==================================== */
/* Enum expressing the items that the company sells. */
enum Items;
/* Array of integers expressing the cost to make each item. */
array[Items] of par 0..10000: item_costs;
/* Array of integers expressing the selling price for each item. */
array[Items] of par 0..10000: item_selling_prices;
/* Array of integers expressing the CO2 emissions of the raw materials composing a unit of each item. */
array[Items] of par 0..10000: item_co2_emissions;
/* Array of integers expressing extra KWatts of energy consumption during production for each item */
array[Items] of par 0..100: item_extra_energy_consumption;
/* Array of integers expressing extra KLiters of water consumption during production for each item */
array[Items] of par 0..100: item_extra_water_consumption;
/* Array of integers expressing the time in hours to make a unit of each item. */
array[Items] of par 0..1000: item_times;
/* ================================= EQUIPMENT DATA (MANDATORY) ==================================== */
/* Enum expressing the possible equipment available. */
enum Equipment;
/* Array of integers expressing the maximum yearly usage in hours of each piece of equipment. */
array[Equipment] of par 0..10000: yearly_equipment_max_usage;
/* Array of integers expressing the KWatt/h consumed by each piece of equipment. */
array[Equipment] of par 0..10000: equipment_energy_consumption_per_hour;
/* Array of integers expressing the KLiters of water consumed by each piece of equipment per hour. */
array[Equipment] of par 0..10000: equipment_water_consumption_per_hour;
/* Array of integers expressing the price in thousands of dollars ($) to buy each piece of equipment. */
array[Equipment] of par 0..100000: equipment_prices;
```

```
/* Array of integers expressing the price in thousands of dollars ($) earned by disposing each piece of
   equipment. */
array[Equipment] of par 0..100000: equipment_disposal_prices;

/* Array of integers expressing the number of units of pieces of equipment that the company owns before
   starting production. */
array[Equipment] of par 0..max_equipment_number: starting_equipment;

/* Array of integers expressing the CO2 tons/h that each piece of equipment generates. */
array[Equipment] of par 0..10000: equipment_co2_emission_per_hour;

/* Integer expressing the maximum number of pieces of equipment that a company can own. */
par 0..100: max_equipment_number;

/* Integer expressing the available budget for purchasing new pieces of equipment. */
par 0..budget: equipment_budget;
/* ==================================== CERTIFICATIONS DATA ======================================= */
/* Array of bools representing the Certifications that the company owns before production starts. */
array[Certifications] of par bool: starting_certifications;

/* ============================= RECYCLABILITY DATA (NON MANDATORY) ================================ */
/* Array of floats representing the recyclable composition ratio (between 0 and 1) for each item (N/D = []).
   */
array[int] of par 0.0..1.0: item_reciclable_compositions;

/* ================================ LOGISTICS DATA (NON MANDATORY) ================================ */
/* Boolean expressing whether to include the logistics related scores in the environmental sustainability
   optimization. */
par bool: use_logistics_kpis;
```

MODULE 2: constants.mzn

```
include "parameters.mzn";

/* =================================== KPIS RELATED CONSTANTS ==================================== */
/* Enum expressing the various KPIs to optimize. */
enum KPIs = { Energy_Efficiency_KPI, CO2_KPI, Transportation_Mode_KPI, Certifications_KPI, Transportation_Capacity_KPI,
  Water_Consumption_KPI, Recycling_KPI, Equipment_Reject_KPI };

/* Array of floats expressing the sustainability relevances of the various KPIs. */
array[KPIs] of par 0.0..5.0: kpis_relevances = [4.42, 4.31, 4.26, 4.20, 4.16, 4.06, 3.91, 3.68];

/* Array of bools expressing which KPIs are considered for optimization of the environmental sustainability
   of the company. */
array[KPIs] of par bool: used_kpis = [ true, true, if use_logistics_kpis == true then true else false endif, true, if use_logistics_kpis ==
  true then true else false endif, true, if length(item_reciclable_compositions) == 0 then false else true endif, true ];

/* ================================= OPTIMIZATION RELATED CONSTANTS ================================ */
/* Float expressing the target reduction rate of total CO2 emissions by 2030 that is supposed be achieved by
   the company. */
par 0.45..0.45: emissions_reduction_rate_by_2030 = 0.45;

/* Float expressing the target reduction rate of total energy consumption by 2030 that is supposed be
   achieved by the company. */
par 0.5..0.5: energy_reduction_rate_by_2030 = 0.5;
```

```
/* Float expressing the target reduction rate of total water consumption by 2030 that is supposed be
   achieved by the company.                                                                            */
par 0.5..0.5: water_reduction_rate_by_2030 = 0.5;

/* =============================== CERTIFICATIONS RELATED CONSTANTS ================================ */
/* Enum expressing the available certifications that the company may own.                             */
enum Certifications = {Carbon_Care, Global_Recycle_Standard, Water_Sense};

/* Array of integers representing the cost of each certification.                                     */
array[Certifications] of par int: certifications_cost = [ if minimum_earnings < 1000 then 1 else 3 endif, 8, 2 ];

/* ================================== LOGISTICS RELATED CONSTANTS ================================== */
/* Enum expressing the possible means of transport to deliver the items.                               */
enum Transportation_Means = {Plane, Ship, Truck, Train};

/* Array of floats expressing the shipment quality score of each mean of transport (Regarding the integrity
   condition of the items and the availability of the service).                                        */
array[Transportation_Means] of 0.0..1.0: transportation_means_quality = [1.0, 0.7, 0.3, 0.5];
/* Array of floats expressing the fuel consumption efficiency score of each mean of transport.         */
array[Transportation_Means] of 0.0..1.0: transportation_means_consumption_efficiency = [0.1, 0.8, 0.8, 1.0];
/* Integer expressing the weight of the quality of transportation (used for computing the normalization of
   the KPI score).                                                                                     */
par 2..2: transportation_means_quality_weigth = 2;

/* Integer expressing the weight of the fuel consumption efficiency of transportation (used for computing
   the normalization of the KPI score).                                                                */
par 8..8: transportation_means_consumption_efficiency_weight = 8;

/* Array of integers expressing the capacity of each transportation mean (measured in units of items).   */
array[Transportation_Means] of int: transportation_means_capacity = [80, 130, 10, 50];

/* Array of integers expressing the cost in thousands of dollars ($) per travel of each mean of transport. */
array[Transportation_Means] of int: transportation_means_costs = [9, 1, 1, 2];
```

MODULE 3: variables.mzn

```
include "parameters.mzn";
include "constants.mzn";
include "functions.mzn";

/* ===================================== KPIs RELATED VARIABLES ===================================== */
/* Array of variable floats containing for each KPI the score achieved by the company.                 */
array[KPIs] of var 0.0..5.0: kpis_scores;

/* Variable float expressing the total CO2 emissions during production.                                */
var 0..10000000: total_production_emissions = get_items_total_emissions() + get_equipment_total_emissions();

/* Variable float expressing the total KPI score percentage.                                           */
var 0.0..100.0: total_kpi_percentage_score = 100 * sum(p in KPIs where used_kpis[p] = true)(kpis_scores[p]) / sum(p in KPIs where
  used_kpis[p] = true)(kpis_relevances[p]);

/* ================================== PRODUCTS RELATED VARIABLES =================================== */
/* Array of variable integers containing for each item the number of units actually produced.          */
array[Items] of var 0..500000: produced_items_number;

/* Variable integer expressing the total earnings.                                                     */
var minimum_earnings..10000000: total_earnings = get_item_total_earnings();
```

```
/* Variable integer expressing the total hours of production. */
var 0..available_time_in_hours: production_total_time = max([yearly_equipment_actual_usage[m] div max([1,
  current_equipment[m]]) | m in Equipment]);

/* ================================== EQUIPMENT RELATED VARIABLES ================================== */
/* Array of variable integers expressing the actual number of units of pieces of equipment that the company
   will own for production. */
array[Equipment] of var 0..max_equipment_number: current_equipment;

/* Array of variable integers expressing the number of units of piece of equipment that the company owns. */
array[Equipment] of var 0..10000: yearly_equipment_actual_usage;

/* Variable float expressing the total energy consumption of the equipment after production. */
var 0..100000000000: equipment_total_energy_consumption = sum(m in Equipment)(yearly_equipment_actual_usage[m] *
  equipment_energy_consumption_per_hour[m]) + sum(i in Items)(produced_items_number[i] * item_extra_energy_consumption[i]);

/* Variable float expressing the total water consumption of the equipment after production. */
var 0..100000000000:equipment_total_water_consumption = sum(m in Equipment)(yearly_equipment_actual_usage[m] *
  equipment_water_consumption_per_hour[m]) + sum(i in Items)(produced_items_number[i] * item_extra_water_consumption[i]);

/* ================================ CERTIFICATIONS RELATED VARIABLES ================================ */
/* Array of variable bools expressing the certifications currently owned by the company during production. */
array[Certifications] of var bool: current_certifications;

/* ================================== LOGISTICS RELATED VARIABLES ================================== */
/* Array of variable integers expressing the number of carried out deliveries for each transportation mean.
   */
array[Transportation_Means] of var 0..1000: transportation_means_used;

/* Array of variable integers expressing the number of total number of items assigned to each transportation
   mean. */
array[Transportation_Means] of var 0..10000: transportation_means_assigned_items;
```

MODULE 4: constraints.mzn

```
include "parameters.mzn";
include "variables.mzn";
include "functions.mzn";

/* ================================ PARAMETERS RELATED CONSTRAINTS ================================ */
/* Constraint to assert that the item_reciclable_compositions array is either empty or it has the same
   length as the available items. */
constraint assert( length(item_reciclable_compositions) == 0 \/ length(item_reciclable_compositions) == card(Items),
  "item_reciclable_compositions must be an empty array or an array that has the same length as Items" );

/* ================================== COSTS RELATED CONSTRAINTS ================================== */
/* The total costs cannot overcome the budget. */
constraint budget - get_total_costs() >= 0;

/* The equipment costs cannot overcome the equipment budget. */
constraint equipment_budget + get_equipment_total_earnings() - get_equipment_total_costs() >= 0;

/* The total earnings must be greater or equal than the minimum expected earnings. */
constraint total_earnings >= minimum_earnings;
```

```
/* ============================== EQUIPMENT AND PRODUCTS CONSTRAINTS ============================== */
/* The starting number of pieces of equipment must be less or equal than the maximum number of pieces of
   equipment. */
constraint sum(m in Equipment)(starting_equipment[m]) <= max_equipment_number;

/* The final number of pieces of equipment must be less or equal than the maximum number of pieces of
   equipment. */
constraint sum(m in Equipment)(current_equipment[m]) <= max_equipment_number;

/* Each piece of equipment usage must be less or equal than its maximum usage time limit. */
constraint forall(m in Equipment)(yearly_equipment_actual_usage[m] <= yearly_equipment_max_usage[m] *
  current_equipment[m]);

/* The total usage of the equipment must be strictly equal to the total production time of the items. */
constraint sum(yearly_equipment_actual_usage) == sum(p in Items)(item_times[p] * produced_items_number[p]);

/* The total time to produce the items must be less or equal than the total available time for production. */
constraint production_total_time <= available_time_in_hours;

/* =============================== CERTIFICATIONS RELATED CONSTRAINTS ============================== */
/* If the company owns a certification at the beginning it will own it even after production. */
constraint forall(c in Certifications)(starting_certifications[c] == true -> current_certifications[c] == true);

/* It is necessary to own a Carbon_Care certification that the company already owns it or that it reduces
   the carbon emissions by at least 5%. */
constraint current_certifications[Carbon_Care] == true -> ( starting_certifications[Carbon_Care] == true \/
  total_production_emissions <= previous_year_co2_emissions - previous_year_co2_emissions * 0.05 );
/* It is necessary to own a Global_Recycle_Standard certification that the company already owns it or that
   it doesn't produce physical items or that makes items which are on average 50% recyclable. */
constraint current_certifications[Global_Recycle_Standard] == true -> ( starting_certifications[Global_Recycle_Standard] == true \/
  used_kpis[Recycling_KPI] == true \/ sum(p in Items)(produced_items_number[p] * item_recyclable_compositions[p]) /
  sum(produced_items_number) >= 0.5 );

/* It is necessary to own a Water_Sense certification that the company already owns it or that it reduces
   the water consumption by at least 20%. */
constraint current_certifications[Water_Sense] == true -> ( starting_certifications[Water_Sense] == true \/
  equipment_total_water_consumption <= previous_year_water_consumption - previous_year_water_consumption * 0.2 );

/* ================================ LOGISTICS RELATED CONSTRAINTS ================================ */
/* Predicate that expresses that if the logistics scores are used, then the capacity of each mean of
   transportation must be respected. */
predicate respect_total_transportation_capacity() = if use_logistics_kpis == false then true else forall(t in
  Transportation_Means)(transportation_means_assigned_items[t] <= transportation_means_capacity[t] *
  transportation_means_used[t]) endif;

/* Predicate that expresses that if the logistics scores are used, then every item must be delivered. */
predicate deliver_all_items() = if use_logistics_kpis == false then true else sum(produced_items_number) ==
  sum(transportation_means_assigned_items) endif;

/* Constraint that checks the respect_total_transportation_capacity predicate. */
constraint respect_total_transportation_capacity();

/* Constraint that checks the deliver_all_items predicate. */
constraint deliver_all_items();
```

MODULE 5: functions.mzn

```
include "parameters.mzn";
include "variables.mzn";
include "constants.mzn";

/* ==================================== KPIs RELATED FUNCTIONS ===================================     */
/* Function that normalizes a KPI score.                                                              */
function var float: get_normalized_kpi_score(0.0..1.0: reduction_by_2030_rate, var float: value, float: previous_year_value) =
let {
var 0.0..1.0: target_reduction_rate = (reduction_by_2030_rate) / (years_to_reach_the_reduction_goals);
var 0.0..previous_year_value: target_value = previous_year_value - previous_year_value * target_reduction_rate;
var float: min_max = 1 - ((value - target_value) / (previous_year_value - target_value));
} in min([max([min_max, 0]), 1]);

/* ================================= PRODUCTION RELATED FUNCTIONS ================================     */
/* Function that computes the total earnings obtained by selling the items as a variable integer.     */
function var 0..100000: get_item_total_earnings() = sum (p in Items)(produced_items_number[p] * item_selling_prices[p]);

/* Function that computes the total cost of the raw materials to produce the items as a variable integer. */
function var 0..100000: get_items_total_costs() = sum (p in Items)(produced_items_number[p] * item_costs[p]);

/* Function that computes the total CO2 emissions in tons of the raw materials used to make the items as a
   variable integer.                                                                                  */
function var 0..100000: get_items_total_emissions() = sum(p in Items)(produced_items_number[p] * item_co2_emissions[p]);

/* Function that computes the total costs of production as a variable integer.                        */
function var 0..budget: get_total_costs() = get_items_total_costs() + get_certifications_total_costs() + equipment_budget +
  get_logistics_total_costs();

/* Function that computes the total CO2 emissions in tons of the equipment during production as a variable
   integer.                                                                                           */
function var 0..1000000: get_equipment_total_emissions() = sum(m in Equipment)(yearly_equipment_actual_usage[m] *
  equipment_co2_emission_per_hour[m]);

/* Function that computes the total earnings in thousands of dollars ($) that are achieved by disposing of
   part of the starting equipment as a variable integer.                                              */
function var 0..100000: get_equipment_total_earnings() = sum(m in Equipment where starting_equipment[m] >
  current_equipment[m])( equipment_disposal_prices[m] * abs(starting_equipment[m] - current_equipment[m]) );

/* Function that computes the total costs in thousands of dollars ($) to buy new equipment for production as
   a variable integer.                                                                                */
function var 0..100000: get_equipment_total_costs() = sum(m in Equipment where starting_equipment[m] <= current_equipment[m])(
  equipment_prices[m] * abs(starting_equipment[m] - current_equipment[m]) );

/* Function that computes the total costs in thousands of dollars ($) to buy new certifications as a
   variable integer,                                                                                  */
function var 0..100000: get_certifications_total_costs() = sum(c in Certifications where starting_certifications[c] == false
  /\current_certifications[c] == true)(certifications_cost[c]);

/* Function that computes the total costs in thousand of dollars ($) for the logistics as a variable
   integer.                                                                                           */
function var 0..100000: get_logistics_total_costs() = if use_logistics_kpis == true then sum (t in Transportation_Means)(
  transportation_means_costs[t] * transportation_means_used[t]) else 0 endif;
```

MODULE 6: main.mzn

```
include "parameters.mzn";
include "variables.mzn";
include "constants.mzn";
include "constraints.mzn";
include "functions.mzn";

/* ===================================== ENERGY EFFICIENCY SCORE ===================================== */
/* Constraint that normalizes the Energy Efficiency score and multiplies it by the Energy Efficiency KPI
   relevance. */
constraint kpis_scores[Energy_Efficiency_KPI] == kpis_relevances[Energy_Efficiency_KPI] *
 get_normalized_kpi_score(energy_reduction_rate_by_2030, equipment_total_energy_consumption,
 previous_year_energy_consumption);

/* ======================================= CO2 EMISSIONS SCORE ======================================= */
/* Constraint that normalizes the CO2 score and multiplies it by the CO2 KPI relevance */
constraint kpis_scores[CO2_KPI] = kpis_relevances[CO2_KPI] * get_normalized_kpi_score(emissions_reduction_rate_by_2030,
 total_production_emissions, previous_year_co2_emissions);

/* ==================================== TRANSPORTATION MODE SCORE ==================================== */
/* Predicate that normalizes the Transporation Mode score and multiplies it by the Transporation Mode KPI
   relevance. If logistic scores are not used for the maximization problem, the predicate sets the score at
   0. */
predicate transportation_mode_score() = if used_kpis[Transportation_Mode_KPI] == true then
 kpis_scores[Transportation_Mode_KPI] = (kpis_relevances[Transportation_Mode_KPI] * sum(t in
 Transportation_Means)(transportation_means_quality_weigth * transportation_means_quality[t] *
 transportation_means_consumption_efficiency_weight * transportation_means_consumption_efficiency[t] *
 transportation_means_used[t]) / (transportation_means_quality_weigth * transportation_means_consumption_efficiency_weight *
 sum(transportation_means_used))) else kpis_scores[Transportation_Mode_KPI] = 0 endif;
/* Constraint that calls the predicate transportation_mode_score. */
constraint transportation_mode_score();

/* ======================================= CERTIFICATIONS SCORE ======================================= */
/* Constraint that normalizes the Certifications score and multiplies it by the Certifications KPI
   relevance. */
constraint kpis_scores[Certifications_KPI] == kpis_relevances[Certifications_KPI] * sum(current_certifications) /
 length(current_certifications);

/* ================================== TRANSPORTATION CAPACITY SCORE ================================== */
/* Predicate that normalizes the Transporation Capacity score and multiplies it by the Transporation
   Capacity KPI relevance. If logistic scores are not used for the maximization problem, the predicate sets
   the score at 0. */
predicate transportation_capacity_score() = if used_kpis[Transportation_Capacity_KPI] == true then
 kpis_scores[Transportation_Capacity_KPI] = kpis_relevances[Transportation_Capacity_KPI] * sum(t in
 Transportation_Means)(transportation_means_assigned_items[t]) / sum(t in
 Transportation_Means)(transportation_means_capacity[t] * transportation_means_used[t]) else
 kpis_scores[Transportation_Capacity_KPI] = 0 endif;

/* Constraint that calls the predicate transportation_capacity_score. */
constraint transportation_capacity_score();
```

```
/* ====================================== WATER EFFICIENCY SCORE ====================================== */
/* Constraint that normalizes the Water Efficiency score and multiplies it by the Water Efficiency KPI
   relevance. */
constraint kpis_scores[Water_Consumption_KPI] == kpis_relevances[Water_Consumption_KPI] *
  get_normalized_kpi_score(water_reduction_rate_by_2030, equipment_total_water_consumption,
  previous_year_water_consumption);

/* ========================================== RECYCLING SCORE ========================================== */
/* Predicate that computes Recycling Efficiency score ratio and multiplies it by the Recycling Efficiency
   KPI relevance. If recycling scores are not used for the maximization problem, the predicate sets the
   score at 0. */
predicate recycling_score() = if used_kpis[Recycling_KPI] == true then kpis_scores[Recycling_KPI] = kpis_relevances[Recycling_KPI]
  * sum(p in Items)(produced_items_number[p] * item_reciclable_compositions[p]) / sum(produced_items_number) else
  kpis_scores[Recycling_KPI] = 0 endif;

/* Constraint that calls the predicate recycling_score. */
constraint recycling_score();

/* ==================================== EQUIPMENT REJECT RATE SCORE ==================================== */
/* Constraint that normalizes the Equipment Reject Rate score and multiplies it by the Equipment Reject Rate
   KPI relevance. */
constraint kpis_scores[Equipment_Reject_KPI] == kpis_relevances[Equipment_Reject_KPI] * (1 - sum(m in Equipment where
  starting_equipment[m] > current_equipment[m])(abs(starting_equipment[m] - current_equipment[m])) / sum(starting_equipment));

/* ============================================= SOLUTION ============================================= */
solve maximize sum(p in 1..card(KPIs) - 1)(kpis_scores[to_enum(KPIs, p)] * kpis_relevances[to_enum(KPIs, p + 1)]) +
  kpis_scores[to_enum(KPIs,card(KPIs))];

/* ============================================== OUTPUT ============================================== */
output [
"Initial company state: \n",
"\t* Budget: $\(budget) k ($\(equipment_budget) k allocated to purchase eventual new equipment);\n",
"\t* Available equipment: \(["\(Equipment[m]): \(starting_equipment[m])" | m in Equipment]).\n \n",

"Company achieved results: \n",
"\t* Total KPI percentage score: \(total_kpi_percentage_score)% \n",
"\t* Earnings: $\(total_earnings) k;\n",
"\t* Number of sold items: \(["\(p): \(produced_items_number[p])" | p in Items]);\n",
"\t* Production time: \(production_total_time);\n",
"\t* Final equipment: \(["\(Equipment[m]): \(current_equipment[m])" | m in Equipment]); \n",
"\t* Final equipment usage: \(["\(Equipment[m]): \(yearly_equipment_actual_usage[m])" | m in Equipment]).\n \n",

"Co2 KPI score: \(kpis_scores[CO2_KPI]) / \(kpis_relevances[CO2_KPI]) \n",
" \t* Previous CO2 emissions: \(previous_year_co2_emissions) tons; \n",
"\t* Reached CO2 emissions: \(total_production_emissions) tons. \n \n",

"Energy efficiency KPI score: \(kpis_scores[Energy_Efficiency_KPI]) / \(kpis_relevances[Energy_Efficiency_KPI])\n",
"\t* Previous energy consumption \(previous_year_energy_consumption) KWatt; \n",
"\t* Reached energy consumption \(equipment_total_energy_consumption) KWatt.\n \n",

"Water KPI score: \(kpis_scores[Water_Consumption_KPI]) / \(kpis_relevances[Water_Consumption_KPI]) \n",
"\t* Previous water consumption \(previous_year_water_consumption) KLiters;\n",
"\t* Reached water consumption \(equipment_total_water_consumption) KLiters.\n \n",

"Certifications KPI score: \(kpis_scores[Certifications_KPI]) / \(kpis_relevances[Certifications_KPI])\n",
"\t* Starting certifications: \(["\(Certifications[c]): \(starting_certifications[c])" | c in Certifications]); \n",
"\t* Final certifications: \(["\(Certifications[c]): \(current_certifications[c])" | c in Certifications]).\n \n",
```

```
if used_kpis[Recycling_KPI] == true
then
"Recycling KPI score: \(kpis_scores[Recycling_KPI]) / \(kpis_relevances[Recycling_KPI]).\n \n"
else "" endif,
"Equipment Reject Rate kpi score: \(kpis_scores[Equipment_Reject_KPI]) /ù \(kpis_relevances[Equipment_Reject_KPI]). \n \n",
if use_logistics_kpis == true
then
"Transportation mode KPI score: \(kpis_scores[Transportation_Mode_KPI]) / \(kpis_relevances[Transportation_Mode_KPI])\n" ++ "\t* Transportation means usage: \(["\(Transportation_Means[t]): \(transportation_means_used[t])" | t in Transportation_Means]).\n \n" ++
"Transportation capacity KPI score: \(kpis_scores[Transportation_Capacity_KPI]) / \(kpis_relevances[Transportation_Capacity_KPI])\n" ++ "\t* Transportation capacity used: \([ "\(Transportation_Means[t]): \(transportation_means_assigned_items[t]) / \(transportation_means_capacity[t] * transportation_means_used[t])" | t in Transportation_Means ]).\n \n"
else "" endif,
];
```

REFERENCES

[1] [n.d.]. Brands for Good CarbonCare©. https://www.carboncareasia.com/eng/Carbon_Solutions/BfG_CCL.php.

[2] [n.d.]. CarbonCare©. https://www.carboncareasia.com/eng/Carbon_Solutions/CarbonCare_Label.php.

[3] [n.d.]. Global Recycled Standard (GRS). https://textileexchange.org/standards/recycled-claim-standard-global-recycled-standard/.

[4] [n.d.]. How EU pulp and paper industry can reduce greenhouse gas emissions - EU Science Hub - European Commission. https://ec.europa.eu/jrc/en/news/how-eu-pulp-and-paper-industry-can-reduce-greenhouse-gas-emissions.

[5] [n.d.]. MiniSearch. https://github.com/minizinc/minisearch.

[6] [n.d.]. Reducing CO2 and Greenhouse Gases. https://recycled-papers.co.uk/green-matters/why-use-recycled-papers/co2-and-greenhouse-gases#:~:text=Manufacturing%20recycled%20paper%20produces%20less,in%20the%20average%20European%20car.

[7] [n.d.]. The Sustainable Development Goals - Goal 13. https://unstats.un.org/sdgs/report/2021/goal-13/.

[8] [n.d.]. Types of Transportation in Logistics: Which Is Right for You? https://www.purolatorinternational.com/types-of-transportation-in-logistics/.

[9] [n.d.]. WaterSense. https://www.ecolabelindex.com/ecolabel/watersense.

[10] [n.d.]. Yuck. https://github.com/informarte/yuck.

[11] 2021. *The Sustainable Development Goals Report 2021 - Extended Report - Goal 6*. Technical Report. UNSD, IEA. https://unstats.un.org/sdgs/report/2021/extended-report/Goal%20(6)_final.pdf

[12] 2021. *The Sustainable Development Goals Report 2021 - Extended Report - Goal 7*. Technical Report. UNSD, IEA. https://unstats.un.org/sdgs/report/2021/extended-report/Goal%20(7)_final.pdf

[13] Fabian Lampe and Lukas Wawrla. [n.d.]. Sustainability in Sourcing & Procurements. ([n. d.]). https://assets.website-files.com/5ee61e86d8a739006abca943/608e3f4abe192a44cb0ff762_Archlet%20Sustainability%20White%20Paper.pdf

[14] N.L. Pollesch and V.H. Dale. 2016. Normalization in sustainability assessment: Methods and implications. *Ecological Economics* 130 (2016), 195–208. https://doi.org/10.1016/j.ecolecon.2016.06.018

[15] Roberto Sebastiani and Patrick Trentin. 2018. OptiMathSAT: A Tool for Optimization Modulo Theories. *Journal of Automated Reasoning* (15 Dec 2018). https://doi.org/10.1007/s10817-018-09508-6