A Real-World Toy Problem – Kinder Surprise's Debut in Optimisation (TOYlib)

Technical Report – Work in Progress

Markus Wagner, *Member, IEEE,*
Optimisation and Logistics Group
School of Computer Science
University of Adelaide
Adelaide, SA, Australia
markus.wagner@adelaide.edu.au

***Abstract*—Kinder Surprise are chocolate eggs that house a small toy. Fans of the toys across the world are collecting and trading these toys, with some toys being very rare and highly priced.**

With this paper, we present and publish a dataset that was extracted from the German Kinder Surprise pricing guide for 2019 – with kind approval of the publisher. The dataset contains prices and photos of 187 sets of figurines produced between 1979 and 2018. In total 2366 items are included.

In addition to a characterisation of the dataset, we provide a few first ideas for using this data as a benchmark dataset.

***Index Terms*—Benchmark, toy problem.**

I. Introduction

Kinder Surprise, sometimes also called Kinder Surprise Egg or Kinder Egg, is a candy manufactured by the Italian company Ferrero since 1974. The eggs are hollow chocolate shells that surround a plastic capsule, which contains a small toy – see Figure 1 for an example.

Fig. 1. Kinder Surprise Photo [1], Creative Commons photo.

The eggs have been sold billions of times to date across the world. The USA has been a noteworthy exception for many years [2], which prompted Ferrero to produce Kinder Joy, an FDA compliant variant [3].

Typically, several "sets" of toys are released each year, with each set comprising of a numer of similarly themed toys. Consequently, it does not come to a surprise that many passionate collectors across the world have been collecting and trading the little toys. Some figurines are extremely rare and much sought after, which can result in high prices. Similarly, rare production variations, e.g., incorrectly coloured parts, can attract much interest. It is important to note that the "fixed" figurines (without any moving parts) are traded the most – although cars, small displays, spinning tops, games, etc. are also very often included in the Kinder Surprise. In this data set, we focus exclusively on the fixed figurines.

For us as researchers, the appeal of the figurines is manifold:

1) For each figurine, the data set provides an anonymised name, a price and a set identifier. This enables the use in set-based and graph-based problems.
2) For each item in the catalog, we include at least one low-resolution photo. This allows us to consider aspects such as shapes and brightness into complex fitness functions.
3) Lastly, quite of few of us can relate to small collectible items, independent of whether these are toys, postal stamps, trading cards, or others.

Interestingly, despite the rather broad presence of collectible items in the real-world, public domain data sets about these are extremely rare. If researchers are interested in particular types of problems that relate to items in sets and their characteristics, and also to the values of sets, then the alternative is often just to produce data sets either randomly or based on some "real-world inspired" process. Whether the characteristics of common instances are captured is often a so-called threat-to-validity.

This is the contribution of this article: the Kinder Surprise 2019 data set forms the beginning of *TOYlib*, and it is available online at https://github.com/markuswagnergithub/TOYlib under the GNU General Public License v3.0 (see the end of this paper for more information). We offer the data, a brief characterisation and a connection to optimisation problems.

Thanks and Disclaimer: We would like to express our sincerest gratitude to André Feiler from the Feiler Verlag https://www.feiler-verlag.de, Germany for providing us with a digital copy of the current pricing guide "O-Ei-A Figuren 2019", and for allowing us to make this data available. The

authors declare no conflict of interest. Neither are they, nor any of their friends or family members involved with Ferrero or the Feiler Verlag.

II. The Dataset

In this section, we first outline how we extracted the extracted and prepared the data. This is followed by a brief characterisation.

A. Table extraction

Our data extraction begun with a copy of the official catalog "O-Ei-A Figuren 2019" in the Adobe InDesign 4 format. The extraction proved to be surprisingly laborious as the data was relatively unstructured and different exports to XML, HTML, and other formats resulted in files that required significant cleanup.[1]

Figure 2 shows an example from a page from the catalogue. In particular, it shows an example where variants of figurines exist.

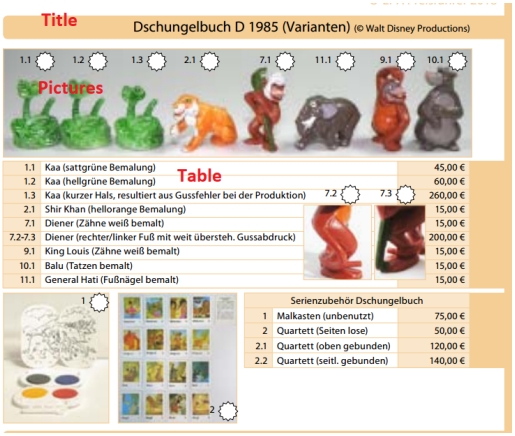

Fig. 2. Excerpt from the catalogue.

After exploring our options, we decided to go for a two-way approach:

- We extracted the tables and photos from an HTML export.
- We used the original PDF to assign tables to pages and to assign tables to sets when unclear, and to assign photos to figurines. We have written a custom tool for this.

We started with the HTML export of the original document (see Figure 3). One challenge that we had to overcome was that the sets' titles are not always present, and they do not always appear before but also after the table. With the help of a few heuristics and some manual cleanup, we extracted and merged all tables of all sets and items, with the associated values. Note that we have not distinguished between figurines and non-figurines, as this could only have been done manually for each item: if a row in a table contained a name and a price, then we added it to the respective set.

[1]As the InDesign file was incompletely tagged, proceeding with the XML export was not an option.

11 General Hati 9,00 €

12 Kleiner Elefant (Schwanz oft abgebrochen!) 10,00 €

Beipackzettel deutsch 170,00 €

Beipackzettel englisch (Jungle-Book) 150,00 €

1.1 Kaa (sattgrüne Bemalung) 45,00 €

1.2 Kaa (hellgrüne Bemalung) 60,00 €

1.3 Kaa (kurzer Hals, resultiert aus Gussfehler bei der Produktion) 260,00 €

2.1 Shir Khan (hellorange Bemalung) 15,00 €

7.1 Diener (Zähne weiß bemalt) 15,00 €

7.2-7.3 Diener (rechter/linker Fuß mit weit übersteh. Gussabdruck) 200,00 €

9.1 King Louis (Zähne weiß bemalt) 15,00 €

10.1 Balu (Tatzen bemalt) 15,00 €

11.1 General Hati (Fußnägel bemalt) 15,00 €

Dschungelbuch D 1985 (Varianten) (© Walt Disney Productions)

Dschungelbuch GB 1990 (© Walt Disney Productions)

Serienzubehör Dschungelbuch

1 Malkasten (unbenutzt) 75,00 €

2 Quartett (Seiten lose) 50,00 €

2.1 Quartett (oben gebunden) 120,00 €

Fig. 3. Excerpt from the catalogue in HTML format.

When we encountered a page in the catalogue focusing on variants of figurines of a set, then we have considered this as a new set. Moreover, when we encountered variants within a set, then we considered each variant to be a new (i.e., additional) item in the set. For example, as the snake Kaa appears in three variants in Figure 2, we added Kaa three times to the set with the respective names.

When we encountered missing values, e.g. in tables highlighting manufacturing variants (see Figure 4), we dropped the items from the set. When we encountered a range or multiple values as a given price, then we decided to proceed with the minimum value of the numbers.

We then calculated the sum of the sets and added this as an entry for each item, for easier lookup.

B. Image extraction and association

The association of images with the individual figurines was challenging for two reasons:

- The individual figurines are normally part of "group shots", and are not available as individual image files.

Fig. 4. Excerpt from the catalogue: missing values

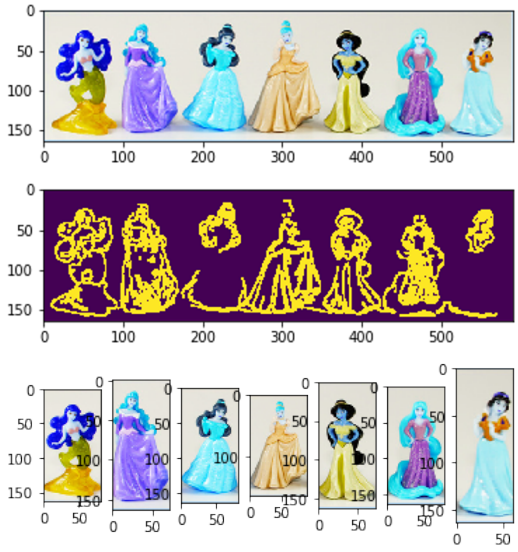

Fig. 5. Steps in the image extraction process. From top to bottom: original image, non-tuned edge detection, final result after tuning. Units are pixels.

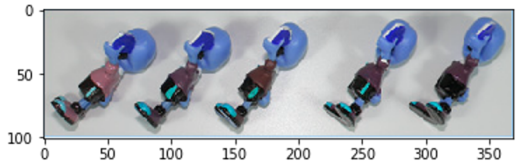

Fig. 6. "Overlapping" figurines in a group shot, units are pixels.

- No original export of the InDesign file would provide a reliable way of connecting images to tables, or even to rows within tables.

The extraction of over 2000 images by hand was not an option, so we sought a semi-automatic process. First, we ran an edge detection algorithm [2] on each group shot, and then go from left to right through the image and cut when a column is empty. For some images, shadows and overlaps make the process tricky, see Figure 5 for an example. However, after some tuning of the parameters, we managed to achieve acceptable results across a wide range of group shots.[3]

Figure 6 shows one of the most difficult situations that we had to deal with: uneven spacing and "overlap" in the sense that vertical cuts would result in the parts of neighbouring figurines to be included. In such cases, we had to accept that overlap cannot easily be avoided and cut the images manually. In addition, we had to manually take care of group shots that involved multiple rows of figurines.

Lastly, we had to manually tend to cases where photos were split too often, where more properly extracted images existed than priced items on a page, and when images appeared on pages different from where they were referenced.

As a result of the semi-automatic process, the image data set contains noise in the sense of incorrect cuts. We see this more as a feature than an issue: real-world data can be noisy, and different approaches can result in different extractions of the individual photos.[4]

[2]Following the recommendations of [4] with adjustments.

[3]Note that this could have been an interesting opportunity for automated parameter tuning of the edge. However, because we were interested in the once-off extraction of parts of the images, and because we did not have ground truth data for a training process but required human intervention, we essentially performed interactive optimisation with a human in the loop.

[4]Should we be allowed to publicly share the group shots in the future, then we will make these available in a different sub-directory, e.g., for image separation and item association benchmarks.

C. Data anonymisation

In order to reduce the risk of commercial loss to the publisher of the catalogue, and also in order to take care of special characters present in the German language, we anonymised the set names and item names. Figure I shows an excerpt from the final table.

The naming convention we used for the images follows the following format:

$item\_image\_prefix :==$ page number_item number_time stamp of cut.

In a few cases, more than one image exists for an item, e.g., due to variants. In such cases, we recommend that the lexicographically first image is associated with an item. In addition

When an item was without an image (but with a price), then we gave this item a white placeholder image of size 150x150 pixels.

III. Characterisation

In the following, we focus the following four features of the data set: set_total_price, item_price, set_size, and item_in_set. While the last one is just a running number within each set, it gives us a slightly different angle at the set characterisation than set_size.

In Figure 7, we start with a few easy-to-compute characteristics. For example, we can see that most sets are comprised

TABLE I
EXCERPT FROM KINDERSURPRISE2019DATA.CSV

item_overall_id	item	item_in_set	item_price	item_image_prefix	set	set_size	set_total_price
1	1_1	1	1200	2_1	1	4	4800
2	1_2	2	1200	2_2	1	4	4800
3	1_3	3	1200	2_3	1	4	4800
4	1_4	4	1200	2_4	1	4	4800
5	2_1	1	200	2_5	2	3	600
6	2_2	2	200	2_6	2	3	600
7	2_3	3	200	2_7	2	3	600
8	3_1	1	1200	2_8	3	3	3600
9	3_2	2	1200	2_9	3	3	3600
10	3_3	3	1200	2_10	3	3	3600
11	4_1	1	400	3_1	4	2	800
12	4_2	2	400	3_2	4	2	800
				...			
2356	187_1	1	3	243_1	187	11	30.5
2357	187_2	2	3	243_2	187	11	30.5
2358	187_3	3	3	243_3	187	11	30.5
2359	187_4	4	3	243_4	187	11	30.5
2360	187_5	5	3	243_5	187	11	30.5
2361	187_6	6	3	243_6	187	11	30.5
2362	187_7	7	3	243_7	187	11	30.5
2363	187_8	8	3	243_8	187	11	30.5
2364	187_9	9	3	243_9	187	11	30.5
2365	187_10	10	3	243_10	187	11	30.5
2366	187_11	11	0.5	243_11	187	11	30.5

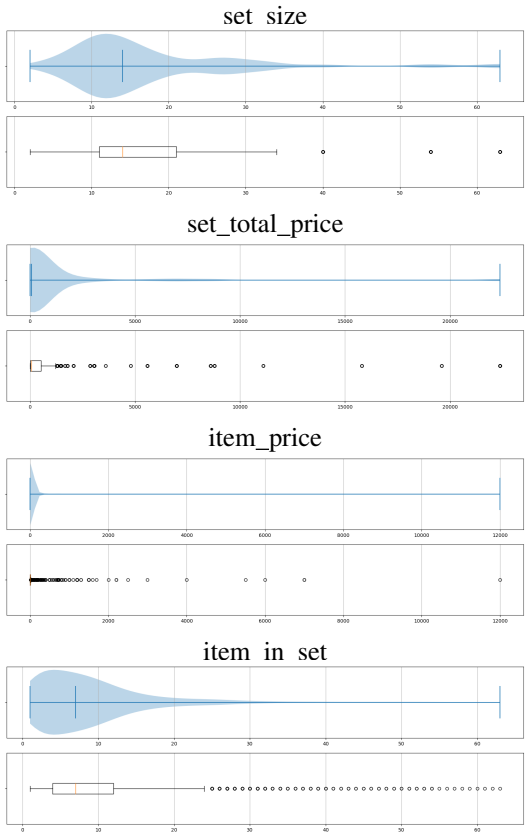

Fig. 7. Characteristics of the data set. Top: three features based on word counts; bottom: two features based on entropy. These violin plot are an alternative to box plots, and they indicate with thickness how common values are.

of just over 10 items this does not come to a big surprise, if one knows that most sets of figurines actually contain 10-12 figurines plus a small leaflet. We can also see that some sets are rather large. The reason for this are the sets there were created for the production variations. We can also observe that most prices are in the order of just a few EUR, which caries over to the values of sets.

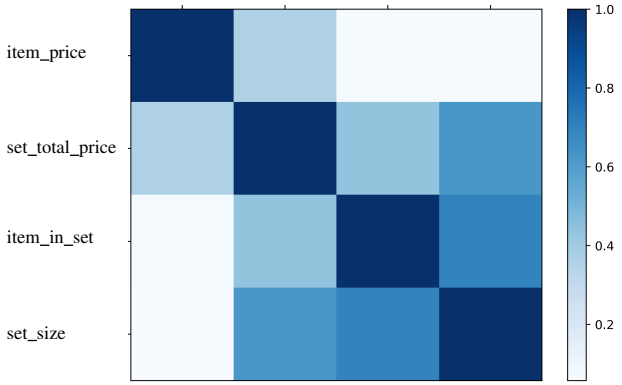

Fig. 8. Correlations of features. Darker fields correspond to a larger correlation between the features. X-labels are omitted as they follow the order of the y-labels.

Figure 8 shows the correlations based on Pearson product-moment correlation coefficients between the four features and clustered with Wards hierarchical clustering approach.[5] As expected, and as previously observed, set_total_price and item_price are correlated, as are set_size and item_in_set.

[5]Implementation provided by asapy [5], https://github.com/mlindauer/asapy, last accessed on 25 January 2019.

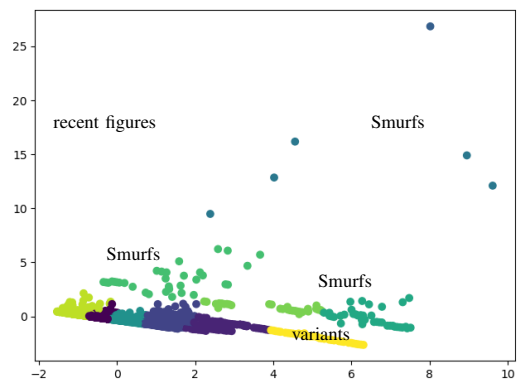

Fig. 9. Projection of the 4D space of features. A principle component analysis is used for the projection of the instances from the feature space into 2D.

Lastly, let us see how well the figurines are distributed in the four-dimensional space. We cluster the corpora in the feature space using a k-means approach. As pre-processing, we use standard scaling and a principal component analysis to two dimensions. To guess the number of clusters, we use the silhouette score on the range of 2 to 12 in the number of clusters. It turns out that many figurines are surprisingly similar. The six "outliers" (consider the y-axis above $y = 10$) are very old and highly-priced "Smurfs" figures from various sets. The second cluster of Smurfs in the bottom right center also consists of highly-priced items. The more we move to the left, the more overlap we see, also because one of the features was item_in_set. Among collectors, the Smurfs have almost attracted much attention, being some of the earliest sets ever to be included in Kinder Surprise.

We can only conjecture that their possibly disproportionally high value makes them stand out in the PCA. As our data set does not contain the year of manufacture, we can also only conjecture that age is correlated with value.

IV. Ideas for Optimisation Problems

After the previous sections have introduced the data set, we now look into real-world problems for which this data set can be used.

A. Function with Matroid Constraints

Much of our current work takes place in the area of combinatorial optimisation. In particular, we are interested in optimising functions under a partition matroid constraint. If the matroid constraints are uniform, then these are also known as cardinality constraints.

Most of the functions that we are interested in are submodular functions, as they capture the notion of diminishing returns[6], i.e. the more we get the less our marginal gain will be. The real world with situation that have this characteristic, thus, the problem of maximizing a submodular function finds applicability in a wide range of situations. Examples include: maximum cut problems [6], combinatorial auctions [7], facility location [8], problems in machine learning [9], coverage functions [10], and online shopping [11].

On the theoretical side, for submodular maximization under matroid and knapsack constraints, the classical result of [8] shows that a greedy algorithm achieves a $1/2$ approximation ratio when maximizing monotone submodular functions under partition matroid constraints. [12] showed that no-polynomial time algorithm can achieve a better approximation ratio than $(1 - 1/e)$.

While theoretical boundaries can often be achieved for certain types of problems and functions, we also deem it important to show in experiments how much closer the analysed algorithms can get than their bound guarantees. Interestingly, suitable real-world data sets that can serve this purpose are surprisingly rare.

Let us now focus on the class of monotone subadditive functions.[7] Subadditivity is a natural property assumed to hold for functions evaluating items sold in combinatorial auctions [13], [14]. Recently, Friedrich et al. [15] have shown that computing the social welfare of a subadditive combinatorial auction is a subadditive function. They considered combinatorial auctions with n players competing for m items, where the items can have different values for each player. Moreover, the value of each item for a player may depend on the particular combination of items allocated to that player. For any given player $i = 1, \ldots, n$, the value of a combination of items is expressed by the *utility function* $u_i \colon 2^{[m]} \to \mathbb{R}_{\geq 0}$. The objective of the social welfare problem (SW) is to find disjoint sets $S_1, \ldots, S_n$ maximizing the total welfare $\sum_{i=1}^{n} u_i(S_i)$. Following [13], they made the following natural assumptions on all utility functions:

1) $u_i(\emptyset) = 0$;
2) $u_i(U \cup T) \leq u_i(U) + u_i(T)$ for all $U, T \subseteq M$;
3) $u_i(U) \leq u_i(T)$ for all $U \subseteq T \subseteq M$.

Friedrich et al. then defined the constraint that each item can be allocated only to one player, and that a function f that, when maximizing f, is equivalent to maximizing a monotone function under a partition matroid constraint. For more technical details, including the proof an improved approximation bound, we refer the interested reader to the article [15].

B. Combinatorial Auctions

Single-minded bidders – Let us consider some basic combinatorial auctions [16] as a starting point for optimisation problems. Here, the bidder is only happy if she gets exactly the items that she is interested in. If a bidder's wishes are not fulfilled (even just partially), and she is not happy. This binary happiness for each bidder can already result in NP-hard problems. For our case, having single-minded bidders makes the problem neither subadditive not superadditive due to the jump.

It is also not submodular, which we can see with a counter example. Let U be an allocation that assigns to bidder A one

[6] strictly speaking: non-increasing returns

[7] the family of non-negative submodular functions is strictly contained in the family of subadditive functions

unwanted item (and thus is unhappy), let V be an allocation that assigns to bidder B one unwanted item (and thus is unhappy), then the union of both allocations can result in a happiness that is greater than zero if the unwanted is in the intersection.

OR-defined happiness [16] – A bit more lenient are so-called OR-bids. There, bids are fulfilled – and thus contribute to happiness – if at least on item from a set of desired items is assigned. In addition, a bidder can specify multiple such sets. The overall happiness here can then be defined as the sum of the individual happinesses assigned to each fulfilled set.

(Un-)capped happiness – A potentially more natural formulation than the single-minded bidder situation is one where bidders get happier and happier, but also with a diminishing return. One approach that we suggest build upon OR-bids and it uses mathematical series and it could look as follows. Assuming n bidders, where each bidder i desires k items $d_{i,1}..d_{i,k}$ (k can vary across bidders):

- For an allocation of items to bidders, sort for each bidder individually the set of items that she actually gets according to their value, largest first, resulting in $g_{i,1}..g_{i,m}$. Undesired but assigned items are simply ignored.
- Optimisation Problem Variant 1: Happiness(i) $= g_{i,1}.value * 1 + g_{i,2}.value * 1/2 + g_{i,3}.value * 1/3 + ... = \sum_{j=1}^{k} 1/n \cdot g_{i,j}.value$
- Optimisation Problem Variant 2: Happiness(i) $= g_{i,1}.value * 1 + g_{i,2}.value * 1/4 + g_{i,3} * 1/9 + ... = \sum_{j=1}^{k} 1/n^2 \cdot g_{i,j}.value$
- The total sum across all bidders is then simply the sum of each bidder's happiness.

Variant 1 with the divergent Harmonic series is effectively a linear function with weights attached to each item. Variant 2 comes close to a linear function that is often referred to as "BinVal" (for "binary values"), where decision variable have the potential to dominate the effect of the other decision variables. Despite this, the variant remains a linear function.

Both happiness functions can be implemented easily, they are inspired by "OR" from combinatorial auctions, and they are "naturally" constructed based on "diminishing returns" and "(un-)capped happiness".

C. A mathematical approach

In some cases, mathematical optimisation might be feasible. To facilitate research in this direction, we provide a so-called a simple problem formulation and an implementation for the solver Gurobi [17] using a Jupyter Notebook (provided as `KinderSurprise2019gurobi.ipynb`). This serves as a starting point for future extension. We outline the technical details to demonstrate that, if the objective function allows it, a mathematical approach can be setup quite quickly.

This simple problem is related to the welfare maximisation problem. Let us assume a situation where a collector can only get one item per set. Moreover, the collector's happyness is identical to the total value of items that she has. This results in a simple optimisation problem:

$$\text{maximise} \sum_{i=1}^{187} \sum_{j} x_{i,j} \cdot x_{i,j}.value$$

such that

$$\sum_{j} x_{i,j} = 1, \forall \text{ sets } 1 \leq i \leq 187$$

where $x_{i,j}$ is the decision variable denoting item j in set i. Note that the sets have varying sizes in our data set.

In python for Gurobi, this looks as follows. Assuming that `items` is a list of tuples

```
<gurobi.tuplelist (2366 tuples, 2 values each):
 ( 1   , 1  )
 ( 1   , 2  )
 ( 1   , 3  )
 ( 1   , 4  )
 ( 2   , 1  )
```

where the first column denotes the set and the second column the respective item within the set, and that `itemvalues` is a dictionary with the corresponding values of the items:

```
{(1, 1): 1200.0,
 (1, 2): 1200.0,
 (1, 3): 1200.0,
 (1, 4): 1200.0,
 (2, 1): 200.0,
```

then we can add the decision variables to a new model m (`m = Model('toys')`) easily:

```
vars = m.addVars(items,
                 vtype=GRB.BINARY,
                 name="x")
```

Adding the constraints is straightforward then:

```
m.addConstrs((vars.sum(i,'*') == 1 for i
    in range(1,188,1)), name='useonce')
```

Our objective function is defined as the product of the decision variables with the respective value:

```
obj = vars.prod(itemvalues)
```

And the optimisation finds an optimal solution[8]:

```
Optimize a model with 187 rows, 2366 columns
                            and 2366 nonzeros
Variable types: 0 continuous, 2366 binary
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [2e-01, 1e+04]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 16339.1
Presolve removed 187 rows and 2366 columns
Presolve time: 0.00s
Presolve: All rows and columns removed

Solution count 2: 53901.8 16339.1

Optimal solution found (tolerance 1.00e-04)
Best objective 5.390185000000e+04,
   best bound 5.390185000000e+04, gap 0.0000%
```

[8]reformatted and simplified to increase readability here

With a solution like the following (omitting non-zero values):

```
x[1,1] 1
x[2,1] 1
x[3,1] 1
x[4,1] 1
x[5,2] 1
x[6,1] 1
x[7,6] 1
...
```

For more details, please see the provided `KinderSurprise2019gurobi.ipynb`. At the time of writing, Gurobi 8.1.0 was available free of charge for academic purposes, see http://www.gurobi.com/academia/for-universities. Jupyter is an open-source application, see https://www.jupyter.org/.

D. Other problem formulations

Based on the idea of welfare, or happiness of agents, we can define many more alternatives that are not straightforward to formulate for mathematical approaches. Hence, the suggestion to use heuristics to solve them.

E. Breaking submodularity

An interesting direction, for both theory and experiments, could be to define functions that are almost everywhere submodular but with a few sets/points, where this breaks. This could then maybe give rise to something interesting about mutation operators and greedy algorithms with restarts.

For this, data is again surprisingly scarce – although one would also expect many real-world scenarios to reflect "The whole is more than the sum of its parts". This saying might go back to Aristotle, however, he does not seem to have left a data set behind that is easily accessible.

Here are five ideas to demonstrate our reasoning process:

Kinder Surprise – One idea here might be to have sets priced higher than the sum of the individual components. What intuitively might make sense at first sight backfires quickly for vendors, as buyers just have to purchase the individual items. As this is indeed not attractive for seller, online shops typically sell collections for less than the total sum of the individual items.[9] In principle, this might still work for rare collections where hardly anyone has individual components, so that having the full set is extremely rare and nobody wants to part with the figurines for little money.

Profit in manufacturing – some modern smartphones costs $500 to produce [18], then add marketing, development, and somewhere between $1,500 and $500 the device manufacturer are still making money. Note that this is different from the economics term "value added", which would be equivalent to a manufacturer's own value added to the product due to software, marketing, infrastructure, etc. What we have in mind is the actual "profit", however, data again is very scarce, with [18] listing prices of only about a dozen components.

[9]For example, the following set is sold for EUR 19, but the sum of the individual items is EUR 20, https://www.eierlei-shop.de/saetze-deutschland/powerpuff-girls-figurinensatz-mit-allen-beipackzetteln.html.

LEGO sets – Each set consist of sometimes thousands of items that can be assembled in a myriad of ways. One iteresting aspect here is the "many components to many sets" relationshop, as many blocks are used throughout many sets. While lists of parts for sets are available, and so are prices for individual items, our tests have not revealed a set that would be worth more than the components. For example when considering a particular set ([19]) and two online shops to source the parts ([20], [21]), the total cost of the components was EUR 4.41, whereas a complete set (as typically sold in shopts) costs EUR 4.95 cents; however, this then also includes the packaging and the building instructions, for which no prices could be sourced. Thus, there is plenty of data, but automation is not trivial, and one cannot always find perfect matches.

Algorithm portfolios [22] – Here, we are referring to the combination with algorithm selection: given a set of algorithms and a new instance, pick one algorithm that you then run. In algorithm portfolios, we have the notions of "single best solver (SB)" (if we could just pick one for the evaluation), "virtual best solver" (assuming we would always pick perfectly), and of course algorithm selectors perform somewhere in-between. One approach to construct a good subset (e.g., given 20 algorithms) is to start with SB and then to add the second solver that add the most in performance (measured in performance of both solvers together in a portfolio). There is some data available (e.g., [23] contains 30+ different scenarios), however, the number of solvers is typically very small. For example, while [24] contains 10k instances and 55 instance features, the actual search space here would only be the $n = 21$ solvers. The search space would be just the $n = 20$ solvers (well, maybe thats not that bad after all, given some measures are computationally costly). We need to include a cardinality constraint to limit the number of algorithms in the portfolio, otherwise it is not interesting.

Specialisation of other kinds – Now we are digressing... For example, we can think of the human race benefiting from the move from "general purpose hunters and gatherers" to "bakers, Beyoncé, and Google". Again, while intuitively sound, data is unavailable.

V. Conclusions and Future Work

The Kinder Surprise 2019 data set forms the beginning of *TOYlib*, and it is available online at https://github.com/markuswagnergithub/TOYlib.

Our long-term vision is establish *TOYlib* as a repository – akin other collections like TSPlib [25] and MIPlib [26] – and to acquire additional data sets with a particular focus on collectibles that are organised in sets. This will include trading cards from sports and trading cards from card games, but also data sets from more serious domains such as the collection of stamps.

As this is an actively maintained data set, variations and additions are likely to happen over time.

Licence

This data set is published under the GNU General Public License v3.0 (GNU GPLv3), see Table II for an outline.

Permissions of this strong copyleft license are conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights.

For more information, we refer the interested reader to https://choosealicense.com/licenses/gpl-3.0/.

TABLE II
OUTLINE OF THE USED GNU GENERAL PUBLIC LICENSE V3.0 (GNU GPLV3)

Permissions	Conditions	Limitations
Commercial use	Disclose source	Liability
Distribution	License and copyright notice	Warranty
Modification	Same license	
Patent use	State changes	
Private use		

Acknowledgments

Our work was supported by the Australian Research Council project DE160100850.

We would like to thank Lujun Weng for his technical support, Tobias Friedrich for coining the term "real-world toy problem", and we would like to thank Andreas Göbel, Timo Kötzing, and Francesco Quinzan for their discussions in the overall project.

Once more, we would like to thank André Feiler from the Feiler Verlag https://www.feiler-verlag.de, Germany for providing us with a digital copy of the current pricing guide "O-Ei-A Figuren 2019", and for allowing us to make this data available.

REFERENCES

[1] United States Customs and Border Protection, "Kinder Surprise Photo," https://commons.wikimedia.org/wiki/File:Kinder_Surprise_Egg.jpg, Creative Commons, 2012, accessed: 25 January 2019.

[2] Huffington Post, "Kinder Surprise USA: Why These Eggs Are Banned South Of The Border," https://www.huffingtonpost.ca/2016/01/26/kinder-surprise-usa_n_9081286.html?ec_carp=1788586484522196048, 2016, accessed: 25 January 2019.

[3] News.com.au, "Americans have been denied the joy of a Kinder Surprise ... until now," https://www.news.com.au/lifestyle/food/eat/americans-have-been-denied-the-joy-of-a-kinder-surprise-until-now/news-story/27190629405fb975b8dc9787ce3c6422, 2017, accessed: 25 January 2019.

[4] Stack Overflow, "Extract object from the image of a box having object," https://stackoverflow.com/questions/43061143/extract-object-from-the-image-of-a-box-having-object/43069349, 2017, accessed: 25 January 2019.

[5] B. Bischl, P. Kerschke, L. Kotthoff, M. Lindauer, Y. Malitsky, A. Frechétte, H. Hoos, F. Hutter, K. Leyton-Brown, K. Tierney, and J. Vanschoren, "Aslib: A benchmark library for algorithm selection," *Artificial Intelligence Journal*, vol. 237, pp. 41–58, 2016.

[6] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.

[7] T. Maehara, Y. Kawase, H. Sumita, K. Tono, and K. Kawarabayashi, "Optimal pricing for submodular valuations with bounded curvature," in *Proc. of AAAI*, 2017, pp. 622–628.

[8] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser, "Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms," *Management Science*, vol. 23, no. 8, pp. 789–810, 1977.

[9] E. R. Elenberg, A. G. Dimakis, M. Feldman, and A. Karbasi, "Streaming weak submodularity: Interpreting neural networks on the fly," in *Advances in Neural Information Processing Systems*, 2017, pp. 4047–4057.

[10] A. Krause, A. P. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, pp. 235–284, 2008.

[11] S. Tschiatschek, A. Singla, and A. Krause, "Selecting sequences of items via submodular maximization," in *Proc. of AAAI*, 2017, pp. 2667–2673.

[12] G. L. Nemhauser and L. A. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Mathematics of Operations Research*, vol. 3, no. 3, pp. 177–188, 1978.

[13] K. Bhawalkar and T. Roughgarden, "Welfare guarantees for combinatorial auctions with item bidding," in *Proc. of SODA*, 2011, pp. 700–709.

[14] S. Assadi, "Combinatorial auctions do need modest interaction," in *EC*, 2017, pp. 145–162.

[15] T. Friedrich, A. Gbel, F. Neumann, F. Quinzan, and R. Rothenberger, "Greedy maximization of functions with bounded curvature under partition matroid constraints," in *Conference on Artificial Intelligence (AAAI)*, 2019.

[16] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. New York, NY, USA: Cambridge University Press, 2007.

[17] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2018. [Online]. Available: http://www.gurobi.com

[18] GSM Arena, "Apple iPhone XS Max components valued at $443," https://www.gsmarena.com/apple_iphone_xs_max_components_value-news-33454.php, accessed: 25 January 2019.

[19] ToysPeriod, "LEGO 10849 My First Plane," https://www.toysperiod.com/lego-set-reference/duplo/lego-10849-my-first-plane/, accessed: 25 January 2019.

[20] BrickScout, "Search results for 6474 ," https://brickscout.com/product-search?q=6474, accessed: 25 January 2019.

[21] ToyPro, "LEGO Einzelteile," https://www.toypro.com/de/p/einzelteile?search=6474, accessed: 25 January 2019.

[22] C. P. Gomes and B. Selman, "Algorithm portfolios," *Artif. Intell.*, vol. 126, no. 1-2, pp. 43–62, Feb. 2001. [Online]. Available: http://dx.doi.org/10.1016/S0004-3702(00)00081-3

[23] B. Bischl, P. Kerschke, L. Kotthoff, M. T. Lindauer, Y. Malitsky, A. Fréchette, H. H. Hoos, F. Hutter, K. Leyton-Brown, K. Tierney, and J. Vanschoren, "Aslib: A benchmark library for algorithm selection," *CoRR*, vol. abs/1506.02465, 2015. [Online]. Available: http://arxiv.org/abs/1506.02465

[24] M. Wagner, M. Lindauer, M. Mısır, S. Nallaperuma, and F. Hutter, "A case study of algorithm selection for the traveling thief problem," *Journal of Heuristics*, vol. 24, no. 3, pp. 295–320, Jun 2018. [Online]. Available: https://doi.org/10.1007/s10732-017-9328-y

[25] G. Reinelt, "TSPLIB - A traveling salesman problem library," Universität Augsburg, Institut für Mathematik, Augsburg, Tech. Rep. 250, 1990.

[26] T. Achterberg, T. Koch, and A. Martin, "MIPLIB 2003," *Operations Research Letters*, vol. 34, no. 4, pp. 361–372, 2006, problems available at *http://miplib.zib.de*.