Kinder Surprise's Debut in Discrete Optimisation – A Real-World Toy Problem that can be Subadditive

Markus Wagner
University of Adelaide
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 data set that was extracted from the German Kinder Surprise pricing guide for 2019. The data set contains prices and photos of 187 sets of figurines produced between 1979 and 2018. In total 2366 items are included.

We also provide a few first ideas for using this data as a benchmark data set for discrete optimisation, i.e., for subadditive functions with matroid constraints, for combinatorial auctions, and for functions with occasionally violated conditions.

KEYWORDS

Benchmarks; toy problem; combinatorial optimisation

ACM Reference Format:
Markus Wagner. 2019. Kinder Surprise's Debut in Discrete Optimisation – A Real-World Toy Problem that can be Subadditive. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3319619.3326801

1 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. The eggs have been sold billions of times to date across the world. The USA has been a noteworthy exception for many years [17], which prompted Ferrero to produce Kinder Joy, a FDA compliant variant [22].

Typically, several "sets" of toys are released each year, with each set comprising of a number 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

Figure 1: Kinder Surprise Photo [29], Creative Commons photo.

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 incorporate aspects such as shapes, brightness, and colours into complex fitness functions when simulating the preferences of collectors.
(3) Lastly, quite a 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 solution 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 a number of discrete optimisation problems.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326801

This paper is organised as follows. First, in Section 2, we describe how we extracted and cleansed the data. Then we provide a brief characterisation in Section 3. Then, we come to optimisation problems in Section 4, where we (1) establish the connection to submodular and subadditive functions with matroid constraints, (2) provide a few formulations with different characteristics, and (3) propose to use the data set for studies on problems with violated conditions, such as "almost always submodular functions".

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.

2 CREATION OF THE DATA SET

In this section, we outline how we extracted and prepared the data. We also explain how we have dealt with missing and ambiguous information. This "technical" section is necessary as it describes the data set creation and cleansing.

2.1 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 production variants of figurines (in the form of different colourings and common manufacturing defects) exist.

Figure 2: Excerpt from the catalogue. Note that the figurines in the photo are out-of-order.

After exploring our options, we decided to go for a two-way approach. First, we extracted the tables and photos from an HTML export. Then, 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.

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

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 €

Figure 3: Excerpt from the catalogue in HTML format. Note that the location of the tables' titles is not consistent with the PDF (see Figure 2).

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 were not always present, and they did 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.

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, because 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 for 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 look-up.


Olympiade der Schlümpfe D 1983 (Varianten Zubehör)

	Farbunterschiede Zubehör „Olympiade der Schlümpfe"	
A	Stelzen - links hellbraun, rechts dunkelbraun	
B	Hüpfseil - links dünnes Seil, rechts dickes Seil, mind. 3 Farbtöne gut unterscheidbar	
C	Löffel - links hellbraun, rechts dunkelbraun	
D	Ei - links hellrot, rechts dunkelrot	
7a	Stelzen, original am Steg	1500,00 €
1a	Löffel, mit Steg (incl. Ei)	1500,00 €

Figure 4: Excerpt from the catalogue: missing values.

2.2 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.
- No export from the InDesign file provides 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 went from left to right through the image and cut when a column is empty. For some images, shadows and overlaps made the process challenging, 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 we 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.

In the future, should we be allowed to publicly share the group shots in high resolution, then we will make these available in a different sub-directory of the current repository, e.g., for image separation and item association benchmarks.

[2]Following the recommendations of [25] 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.

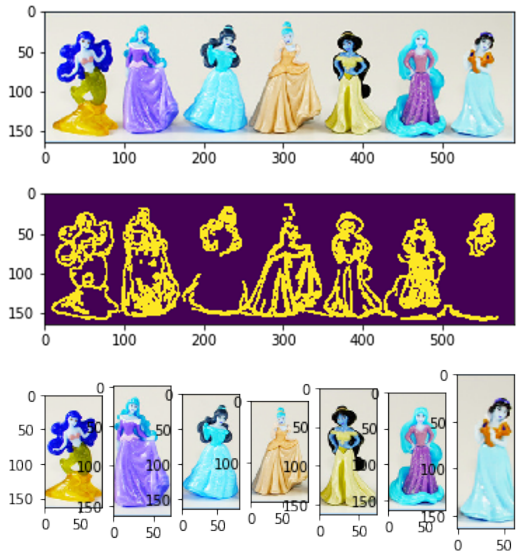

Figure 5: Steps in the image extraction process. From top to bottom: original image, non-tuned edge detection (figurines could not be separated reliably), final result after tuning. Units are pixels.

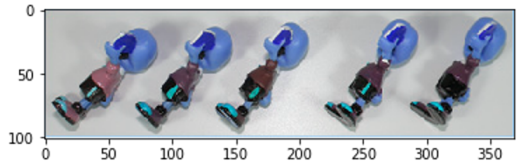

Figure 6: "Overlapping" figurines in a group shot, units are pixels.

2.3 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. Table 1 shows an excerpt from the final table. Further anonymisation could have been performed, e.g., scrambling the items within the sets and the order of the sets, however, the chosen level was sufficient for the publisher to let us publish the data.

The naming convention we used for the images follows the following format: *item_image_prefix :==*
<page number>_<item_in_set>_<time stamp of cut>.<file extention>

For 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.

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

3 CHARACTERISATION

In the following, we provide a brief characterisation of the items in the data set. We focus the following four features of the data set: set_total_price, item_price, set_size, and item_in_set. While

Table 1: Excerpt from `KinderSurprise2019data.csv`. For example, the last set with the number 187 contains 10 figures (each valued at EUR 3.00) and 1 leaflet (valued at EUR 0.50).

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
⋮							
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

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.

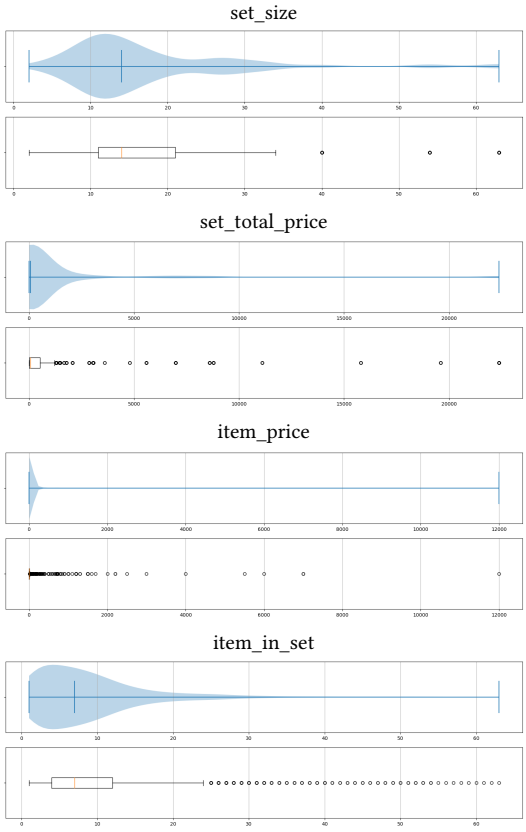

Figure 7: Characteristics of the data set. The violin plot are an alternative to box plots, and they indicate with thickness how common values are.

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

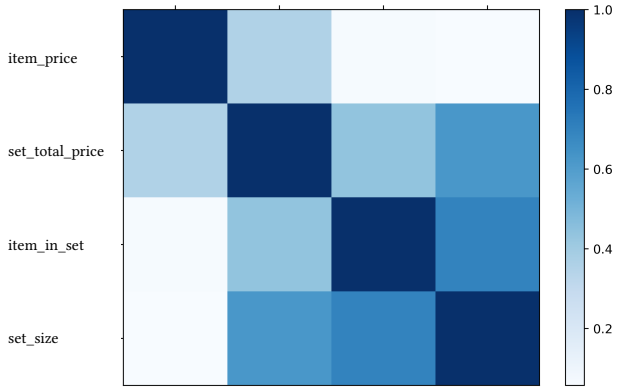

Figure 8: Correlations of features. Darker fields correspond to a larger correlation between the features. X-labels are omitted as they follow the order of the y-labels.

Figure 8 shows the correlations based on Pearson product-moment correlation coefficients between the four features and clustered with Wards hierarchical clustering approach.[4] As expected, and as previously observed, set_total_price and item_price are correlated, as are set_size and item_in_set.

Lastly, let us investigate 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.

[4]Implementation provided by asapy [4], https://github.com/mlindauer/asapy, last accessed on 25 February 2019.

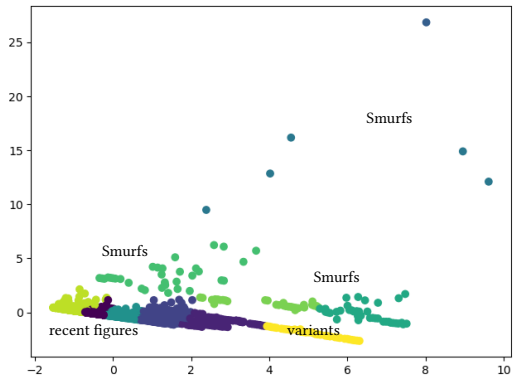

Figure 9: Projection of the 4D space of features. A principal component analysis is used for the projection of the instances from the feature space into 2D.

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 top half of the figure, with $y \geq 10$) are very old and highly-priced "Smurfs" figurines from various sets. The second cluster of Smurfs in the bottom right centre 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 attracted much attention, being some of the earliest sets ever to be included in Kinder Surprise and thus (naturally) among the rarest nowadays.

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. Similarly, in principle, the colouring and shapes of figurines could be correlated with valuations. A characterisation of the photos (and ideas for considering these aspects in optimisation problems) will be added eventually to the repository, however, it is outside the scope of this article.

For the use in discrete optimisation benchmarks, we conjecture that this structure can result in some challenging scenarios, depending on the formulation. For example, the resulting hard-to-solve "cores" of problems can become relatively large when many items have comparable (but not identical) properties, e.g., in the case of knapsack problems [20].

4 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.

4.1 Functions 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. When collecting figurines, we can easily think of scenarios where the task is to distribute a fixed number of figurines among collectors – hence the connection to functions with cardinality constraints.

Most of the functions that we are interested in are submodular functions, as they capture the notion of diminishing returns[5], 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 [13], combinatorial auctions [19], facility location [8], problems in machine learning [11], coverage functions [18], and online shopping [28]. In our setting, diminishing returns can occur when the happiness function of a collector contains aspects such as "the increase of happiness is sub-linear with the number of figurines a collector has".

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. [21] showed that no polynomial time algorithm can achieve a better approximation ratio than $(1 - 1/e)$. While theoretical boundaries like these 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.[6] Subadditivity is a natural property assumed to hold for functions evaluating items sold in combinatorial auctions [2, 3]. Recently, Friedrich et al. [12] 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 – the very same can be said for collectors of Kinder Surprise toys. 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 [3], 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 [12]. This gives us our possible connection to submodular and subadditive functions.

4.2 Combinatorial Auctions

Let us consider some basic combinatorial auctions [23] as another starting point for optimisation problems using our data set.

[5] strictly speaking: non-increasing returns
[6] the family of non-negative submodular functions is strictly contained in the family of subadditive functions

Single-minded bidders [23] – 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), then 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 nor superadditive due to the happiness that is essentially a needle-in-a-haystack.

It is also not submodular, which we can see with a simple counter example. Let U be an allocation that assigns to bidder A one 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 [23] – A bit more lenient are so-called OR-bids. There, bids are fulfilled – and thus contribute to happiness – if at least one 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 happiness-es 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 diminishing returns. The approach that we suggest builds upon OR-bids. 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):[7]

- 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 "uncapped happiness": $\text{Happiness}(i) = g_{i,1}.value * 1 + g_{i,2}.value * 1/2 + g_{i,3}.value * 1/3 + \ldots = \sum_{j=1}^{k} 1/n \cdot g_{i,j}.value$
- Optimisation Problem Variant 2 "capped happiness": $\text{Happiness}(i) = g_{i,1}.value * 1 + g_{i,2}.value * 1/4 + g_{i,3} * 1/9 + \ldots = \sum_{j=1}^{k} 1/n^2 \cdot g_{i,j}.value$
- The total sum across all bidders is then the total sum of each bidder's happiness.

Variant 1 with the divergent Harmonic series is effectively a linear function with coefficients attached to each item. Variant 2 comes close to the linear function that is often referred to as "BinVal" (for "binary values" as coefficients), where decision variables have the potential to dominate the additive 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".

4.3 Unchartered Territory: Breaking submodularity and similar properties

An interesting direction, for both theoretical and empirical studies, is to define functions that are almost everywhere submodular/subadditive/... but with a few sets or points, where this property breaks. This could then maybe give rise to interesting statements for mutation operators and greedy algorithms with restarts. To the best of our knowledge, no works about such occasional violations or prevailing conditions exist yet – and in particular about their effects – for example, about "almost always submodular functions". While there are works characterising, e.g., the degree of submodularity via the so-called *curvature* [7, 30], they do not consider violations.

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". While this saying might go back to Aristotle, he does not seem to have left behind a data set that is easily accessible to support his claims.

Here are five ideas to demonstrate our reasoning process:

Kinder Surprise (again) – One idea here might be to have (some) 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 in order to get an eventual "boost" for free. As this is indeed not attractive for sellers, online shops typically sell collections for less than the total sum of the individual items – for example, the set [10] is sold for EUR 19, but the sum of the individual items is EUR 20. 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 and sentimental reasons.

Profit in manufacturing – For example, some modern smartphones cost about $500 to produce [15], and when the phone is sold for $1,500, then the manufacturer still makes money after subtracting cost for marketing, development, etc. 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". Interestingly, data again is very scarce, with [15] listing prices of only about a dozen components.

LEGO sets – Each LEGO set consists of sometimes thousands of items that can be assembled in a myriad of ways. One interesting aspect here where it differs from our Kinder Surprise data is the "many components to many sets" relationship, as many blocks are used throughout many sets. While lists of parts for sets are available, and so are prices for individual items, our investigations have not revealed a set that would be worth more than the components. For example when considering a particular set [27] and two online shops to source the parts [6, 26], the total cost of the components was EUR 4.41, whereas a complete set (as typically sold in shops) 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 acquiring the data collection is not trivial, and one cannot always find perfect matches.

Algorithm portfolios [14] – 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 adds the most in performance

[7] We use the notation $g_{i,j}.value$ to informally refer to bidder i's valuation of the item j. Each bidder might have a different valuation for item j.

(measured in performance of both solvers together in a portfolio). There is some data available (e.g., [5] contains 30+ different scenarios), however, the number of solvers is typically very small. For example, while [31] contains 10k instances and 55 instance features, the actual search space here would only be the $n = 21$ solvers. We would need to include a cardinality constraint to limit the number of algorithms in the portfolio, otherwise it is not interesting.

Specialisation of other kinds – With the last one, we are digressing a fair bit... For example, we can think of the human race benefiting from the move from "general purpose hunters and gatherers" to "bakers, Beyoncé, and BB-DOB Workshop organisers". Again, while this is intuitively sound, data is unavailable.

5 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. We propose to use it for relatable, discrete optimisation problems.

Our long-term vision is to establish *TOYlib* as a repository – much akin other collections like TSPlib [24] and MIPlib [1] – and to acquire additional data sets with a particular focus on collectables 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. For example, we envision interfaces to optimisation packages and frameworks like IOHprofiler [9].

Licence

This data set is published under the GNU General Public License v3.0 (GNU GPLv3), see Table 2 for an outline.

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

Table 2: 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

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.

(funding sources and further acknowledgements removed for double-blind review)

REFERENCES

[1] T. Achterberg, T. Koch, and A. Martin. 2006. MIPLIB 2003. *Operations Research Letters* 34, 4 (2006), 361–372. Problems available at *http://miplib.zib.de.*

[2] Sepehr Assadi. 2017. Combinatorial Auctions Do Need Modest Interaction. In *EC*. 145–162.

[3] Kshipra Bhawalkar and Tim Roughgarden. 2011. Welfare Guarantees for Combinatorial Auctions with Item Bidding. In *Proc. of SODA*. 700–709.

[4] B. Bischl, P. Kerschke, L. Kotthoff, M. Lindauer, Y. Malitsky, A. Frechétte, H. Hoos, F. Hutter, K. Leyton-Brown, K. Tierney, and J. Vanschoren. 2016. ASlib: A Benchmark Library for Algorithm Selection. *Artificial Intelligence Journal* 237 (2016), 41–58.

[5] Bernd Bischl, Pascal Kerschke, Lars Kotthoff, Marius Thomas Lindauer, Yuri Malitsky, Alexandre Fréchette, Holger H. Hoos, Frank Hutter, Kevin Leyton-Brown, Kevin Tierney, and Joaquin Vanschoren. 2015. ASlib: A Benchmark Library for Algorithm Selection. *CoRR* abs/1506.02465 (2015). arXiv:1506.02465 http://arxiv.org/abs/1506.02465

[6] BrickScout. 2019. Search results for âĂđ6474âĂIJ . https://brickscout.com/product-search?q=6474. (2019). Accessed: 25 February 2019.

[7] Michele Conforti and Gerard Cornuejols. 1984. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Applied Mathematics* 7, 3 (1984), 251 – 274.

[8] Gerard Cornuejols, Marshall L. Fisher, and George L. Nemhauser. 1977. Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms. *Management Science* 23, 8 (1977), 789–810.

[9] Carola Doerr, Hao Wang, Furong Ye, Sander van Rijn, and Thomas Bäck. 2018. IOHprofiler: A Benchmarking and Profiling Tool for Iterative Optimization Heuristics. *arXiv e-prints:1810.05281* (Oct. 2018). arXiv:1810.05281 https://arxiv.org/abs/1810.05281

[10] Eierlei Shop. 2019. Powerpuff Girls Figurinensatz mit allen Beipackzetteln. https://www.eierlei-shop.de/saetze-deutschland/powerpuff-girls-figurinensatz-mit-allen-beipackzetteln.html. (2019). Accessed: 25 February 2019.

[11] Ethan R. Elenberg, Alexandros G. Dimakis, Moran Feldman, and Amin Karbasi. 2017. Streaming Weak Submodularity: Interpreting Neural Networks on the Fly. In *Advances in Neural Information Processing Systems*. 4047–4057.

[12] Tobias Friedrich, Andreas Goebel, Frank Neumann, Francesco Quinzan, and Ralf Rothenberger. 2019. Greedy Maximization of Functions with Bounded Curvature Under Partition Matroid Constraints. In *Conference on Artificial Intelligence (AAAI)*. Accepted for publication.

[13] Michel X. Goemans and David P. Williamson. 1995. Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *Journal of the ACM* 42, 6 (1995), 1115–1145.

[14] Carla P. Gomes and Bart Selman. 2001. Algorithm Portfolios. *Artif. Intell.* 126, 1-2 (Feb. 2001), 43–62.

[15] GSM Arena. 2019. Apple iPhone XS Max components valued at $443. https://www.gsmarena.com/apple_iphone_xs_max_components_value-news-33454.php. (2019). Accessed: 25 February 2019.

[16] LLC Gurobi Optimization. 2018. Gurobi Optimizer Reference Manual. (2018). http://www.gurobi.com

[17] Huffington Post. 2016. 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 February 2019.

[18] Andreas Krause, Ajit Paul Singh, and Carlos Guestrin. 2008. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research* 9 (2008), 235–284.

[19] Takanori Maehara, Yasushi Kawase, Hanna Sumita, Katsuya Tono, and Ken-ichi Kawarabayashi. 2017. Optimal Pricing for Submodular Valuations with Bounded Curvature. In *Proc. of AAAI*. 622–628.

[20] Silvano Martello, David Pisinger, and Paolo Toth. 1999. Dynamic Programming and Strong Bounds for the 0-1 Knapsack Problem. *Management Science* 45, 3 (1999), 414–424.

[21] George L. Nemhauser and Laurence A. Wolsey. 1978. Best Algorithms for Approximating the Maximum of a Submodular Set Function. *Mathematics of Operations Research* 3, 3 (1978), 177–188.

[22] News.com.au. 2017. 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 February 2019.

[23] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. 2007. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA.

[24] G. Reinelt. 1990. *TSPLIB - A Traveling Salesman Problem Library*. Technical Report 250. Universität Augsburg, Institut für Mathematik, Augsburg. 18 pages.

[25] Stack Overflow. 2017. 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 February 2019.

[26] ToyPro. 2019. LEGO Einzelteile. https://www.toypro.com/de/p/einzelteile?search=6474. (2019). Accessed: 25 February 2019.
[27] ToysPeriod. 2019. LEGO 10849 My First Plane. https://www.toysperiod.com/lego-set-reference/duplo/lego-10849-my-first-plane/. (2019). Accessed: 25 February 2019.
[28] Sebastian Tschiatschek, Adish Singla, and Andreas Krause. 2017. Selecting Sequences of Items via Submodular Maximization. In *Proc. of AAAI*. 2667–2673.
[29] United States Customs and Border Protection. 2012. Kinder Surprise Photo. https://commons.wikimedia.org/wiki/File:Kinder_Surprise_Egg.jpg, Creative Commons. (2012). Accessed: 25 February 2019.
[30] Jan Vondrák. 2010. Submodularity and curvature: the optimal algorithm. *RIMS Kokyuroku Bessatsu* B23 (2010), 253–266.
[31] Markus Wagner, Marius Lindauer, Mustafa Mısır, Samadhi Nallaperuma, and Frank Hutter. 2018. A case study of algorithm selection for the traveling thief problem. *Journal of Heuristics* 24, 3 (2018), 295–320.

6 APPENDIX

We have moved the demonstration of a simple mathematically solvable formulation into the appendix, as it serves more the role of a mini-tutorial for the use of our dataset with Gurobi, than to propose new formulations for discrete optimisation.

While the following is a feasible formulation, it might be necessary to employ meta-heuristics in order to tackle aspects such as multi-objectiveness, diversity, and non-linearity of the objective functions.

6.1 An example of a mathematical approach

Depending on the problem formulation, mathematical optimisation might be feasible to provably compute the optimal solution. To facilitate research in this direction, we provide a simple problem formulation and an implementation for the solver Gurobi [16] using a Jupyter Notebook (provided as `KinderSurprise2019gurobi.ipynb`). This can serve interested researchers 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 happiness 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.

When using Gurobi's Python API, 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('KinderSurprise2019')`) easily:

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

Adding the constraints specifying that only one item per set can be allocated 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 (output reformatted and simplified to increase readability):

```
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%
```

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
...
```

We can see that exactly one item per set has been chosen.

For more details, please see the provided `KinderSurprise2019gurobi.ipynb`. At the time of writing, Gurobi 8.1.0 is 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/.