Bin Packing Problem Instances

This document describes two sets of Benchmark Problem Instances for the One Dimensional Bin Packing Problem. The problem instances are supplied as compressed SQLITE database files.

Definition

The One Dimensional Bin Packing Problem is an NP hard Combinatorial Optimisation Problem that has been the subject of much research in the scientific community.The objective is to pack a number of different sized items into as few fixed sized containers, or bins, as possible.The following video shows how a simple deterministic heuristic solves a particular problem instance.

Problem Set A

Available at repository? currently roll web site problem set A consists of 15830 problem instances that were used in the following publication.

Sim, K., Hart, E.: Generating single and multiple cooperative heuristics for the one dimensional bin packing problem using a single node genetic programming island model. In: Proceedings of GECCO 2013. ACM, New York, NY, USA (2013)

Preprint @ napier

Full @ ACM

Problem Set B

Available at repository? currently roll web site problem set B consists of 3968 problem instances that were used in the following publications.

Sim, K., Hart, E., Paechter, B.: *Learning to solve bin packing problems with an immune inspired hyper-heuristic*. In: ECAL 2013 – 12th European Conference on Artificial Life. MIT Press (2013)

Preprint @ napier

Full @ MIT Press

K. Sim, E. Hart, and B. Paechter. *A lifelong learning hyper-heuristic method for bin packing*. Evolutionary Computation 2015, 23, 37-67

Preprint @ napier

Full : MIT Press

The problem instances were generated using a custom designed generator that attempts to create problem instances similar to others produced in the literature. Problem instances are generated using the following characteristics:

- The bin capacity C
- The number of items n
- The ratio of small items of size $\omega <= C/4$
- The ratio of medium items of size $C/4 < \omega <= C/3$

- The ratio of large items of size C/3 < ω <= C/2
- The ratio of huge items of size ω > C/2
- The total free space in the optimal solution summed across all the bins.

All instances have known optimal solutions at the lower bound given by:

$$OPT_{bins} = \left\lceil \frac{\sum_{j=1}^{n} \omega_j}{C} \right\rceil$$

1370 problem instances taken from the literature [1,2] were used to determine the parameters from which to generate new instances. The parameters used to generate these problems are shown in the following table and described in the publications referenced here. All instances are generated using integer values for bin capacity and item weight with the exception of the data set labelled **FalT** which were factored in order to be consistent.

Data Set	capacity (c)	n	ω	#Problems
$ds1$	100,120,150	50,100,200,500	[1,100],[20,100],[30,100]	$36 \times 20 = 720$
$ds3$	100000	200	[20000,30000]	10
$FalU$	150	120,250,500,1000	[20,100]	$4 \times 20 = 80$
$FalT$	1	60,120,249,501	[0.25,0.5]	$4 \times 20 = 80$

Data Set	c	n	ϖ (avg weight)	$\delta(\%)$	# Problems
$ds2$	1000	50,100,200,500	$\frac{c}{3}, \frac{c}{5}, \frac{c}{7}, \frac{c}{9}$	20,50,90	$48 \times 10 = 480$

Each problem instance was sampled and the number of small, medium, large and huge items was calculated and used along with the number of items and the bin capacity to generate new instances. The generator attempts to create problem instances where the free space summed across all bins is zero, a setting that increases the complexity of the problem instances. The correct number of items is generated at random for each of the four size ranges sampled. Items are then packed into the correct number of bins placing each into the bin with the smallest used space with no restriction imposed on the bin capacity. This results in a solution with bins either over or under packed. Items are then adjusted in size so as to exactly fill the bin capacity C. Note that the process is not always successful and some problem instances are discarded. Also the procedure results in some instances with a disproportionate number of items with weights at each of the size range boundaries.

For Problem Set A, consisting of 15,830 problem instances, many more were initially generated. The problem instances were then solved using 4 deterministic heuristics, described in the publications above, along with an implementation of Falkenauer's Hybrid Grouping Genetic Algorithm. Any problem instances for which an optimal solution was found were discarded, resulting in a set of 15,830 hard problem instances.

Problem Set A can be downloaded here as a compressed SQLITE database.

The format of files is shown using the following SQL statement

CREATE TABLE [problem] (

[pr_id] INTEGER,

[pr_dataset] INTEGER,

[pr_opt] INTEGER,

[pr_capacity] INTEGER,

[pr_name] TEXT,

[pr_weights] TEXT,

[isBenchmark] BOOLEAN,

[items] INTEGER,

[solution] TEXT

)

All values are either Integers or text representations of Integer values.

- [pr_id] is a unique identifier set to 1371-17201 inclusive for Problem Set A and to 20000-23967 for Problem Set B.
- The field [pr_dataset] is unused in both of these problem data sets.
- [pr_opt] is the optimal number of bins for the problem instance.
- [pr_capacity] is the bin capacity.
- [pr_name] is a text identifier of the form nnnnxx where nnnn represents the original problem instance, taken from [1,2] and described in [3,4], from which the parameters used for the problem generator were sampled and xx is a sequential identification number indicating the amount of instances generated using those parameters.
- [pr_weights] is a text field storing the individual problem instances weights. These are stored as a string of comma delimited integer values.
- The field [isBenchmark] is unused in both of these problem data sets.
- [items] indicates the number of items in the problem instance.
- [solution] is a text field which stores one known optimal solution. This is stored as a string with each bin delimited using a semicolon and individual items within each bin separated using commas.

The second data set, available for download here, is supplied in the same SQLITE format and consists of 3968 problem instances that were generated using the process outlined above. Three problem instance were generated for the parameters sampled from each of the 1370 problem instances outlined above. All successfully generated problem instances were kept.

References

[1] E. Falkenauer. A hybrid grouping genetic algorithm for bin packing. Journal of Heuristics, 2:5–30, 1996.

[2] A. Scholl, R. Klein, and C. J¨urgens. Bison: a fast hybrid procedure for exactly solving the one-dimensional bin packing problem. Comput. Oper. Res., 24(7):627–645, 1997.

[3] Sim, K., Hart, E.: Generating single and multiple cooperative heuristics for the one dimensional bin packing problem using a single node genetic programming island model. In: Proceedings of GECCO 2013. ACM, New York, NY, USA (2013)

[4] Sim, K., Hart, E., Paechter, B.: Learning to solve bin packing problems with an immune inspired hyper-heuristic. In: ECAL 2013 – 12th European Conference on Artificial Life. MIT Press (2013)