Chapter 5
Computational Experience and Real-World Context

Dealing with non-standard packing problems, and precisely because they are by definition outside the framework of any conventional classification, poses, from the experimental point of view, non-negligible difficulties. Firstly, a remarkable effort is requested to collect, or even generate ex novo, non-trivial instances. They need, actually, to cover an adequate number of scenarios, representative of a sufficiently wide area of real-world applications. Secondly, the elaboration of instances of practical interest is mostly very time consuming. Therefore, an extensive dedicated test campaign is extremely demanding, both in terms of human and computational resources.

In this chapter, an attempt has been made to provide useful insights on the computational aspects, relevant to the various formulations and approaches proposed so far. The author is, nonetheless, aware that a systematic and exhaustive experimental approach could hardly be followed. Outcomes concerning the relevant (ongoing) trial activity are reported hereinafter. The case studies are grouped in separate sections, based on different perspectives, also with the expectation of stimulating possible directions for further dedicated research. In the whole chapter, if not otherwise specified, IBM ILOG Optimizer 12.3 (IBM Corporation 2010) is referred to as the MIP solver adopted, supported by a personal computer, equipped with Core 2 Duo P8600, 2.40 GHz processor; 1.93 GB RAM; and MS Windows XP Professional, Service Pack 2.

5.1 Direct Solutions Obtained from the General MIP Model

The heuristic approaches proposed in Chap. 4 have been introduced to obtain satisfactory (suboptimal) solutions to real-world model instances whilst reducing the computational effort as much as possible. As it is easily gathered, most practical

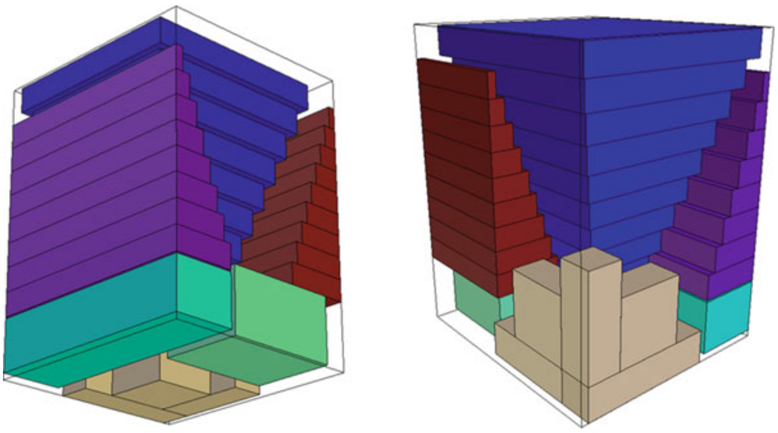
Fig. 5.1 Tetris-like items inside a parallelepiped (Case Study 1.14)

instances can hardly be solved directly, as a matter of fact. This is because of the general MIP model intrinsic difficulties (Sect. 4.1) that are significantly increased when additional conditions (Sect. 2.3) are present. Nonetheless, when relatively small-scale exercises are involved, good-quality results can be obtained as well, by solving the model directly. A set of pertinent case studies (1.1–1.20, see Appendix) has been considered, as a general indication. The packing instances in question are expressed in terms of *feasibility* (i.e., all items have to be loaded). They have been deliberately 'fabricated', in order to deal with cases known a priori for admitting at least one solution.

These case studies contain both single parallelepipeds and actual tetris-like items. All are allowed any possible rotation. The domain is always a parallelepiped, except for Case Study 1.18, for which it is a right prism. The general MIP model of Sect. 2.1 (including some of the auxiliary constraints discussed in Sect. 2.3) has been utilized with a different *objective* function, aimed at minimizing the centre of mass off-centring. For this purpose, the minimum 'virtual' cube, acting as centre of mass domain and 'centred' with respect to the container, is searched for (in Case Studies 1.1–1.4 and 1.7–1.11, the 'virtual' cube has additionally been provided with upper bounds). It is assumed that all items involved are homogeneous and have the same density.

Some details are reported in Tables A.1 and A.2; see Appendix. Figures 5.1, 5.2 and 5.3 represent Case Studies 1.14, 1.18 and 1.20, respectively. As usually understood throughout the whole text, all *components* of each tetris-like item are represented with the same colour.

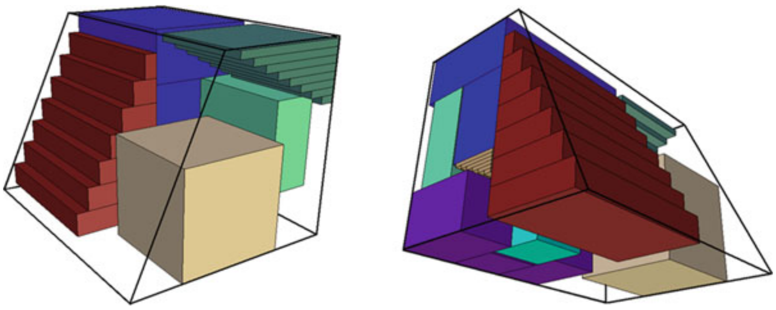

Fig. 5.2 Tetris-like items inside a right prism (Case Study 1.18)

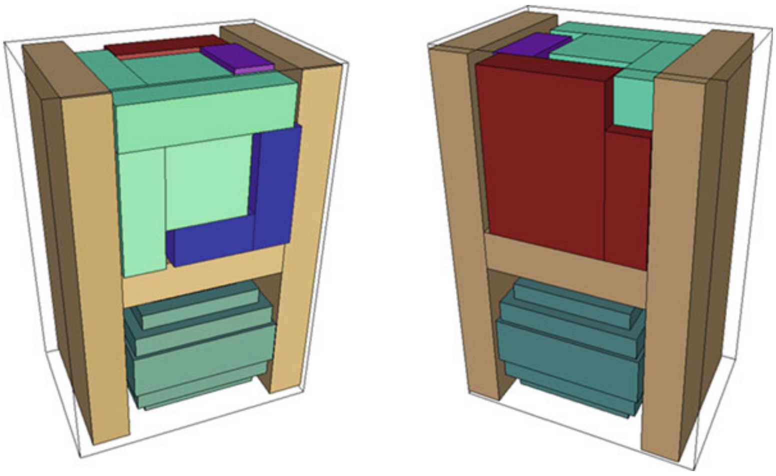

Fig. 5.3 Large tetris-like item acting as a domain (Case Study 1.20)

5.2 Direct Solutions Obtained by Reformulations of the General MIP Model

A number of case studies are considered in this section, focusing on the general MIP model reformulations 3.1.2 and 3.1.3, discussed in Chap. 3. All cases reported in this section focus on the packing of single parallelepipeds, with any possible rotation, into a parallelepiped. No additional conditions have been included (apart from the presence of *separation* planes in Case Study 2.2). All the case studies considered in this section have been solved by utilizing IBM ILOG CPLEX 12.5.1 (supported by a personal computer, equipped with Core 2 Duo P8600, 2.40 GHz processor; 1.93 GB RAM; MS Windows XP Professional, Service Pack 2; and CPLEX 12.5.1 version significantly outperforms 12.3, also referred to in this section).

Table 5.1 Six classes of test parallelepipeds

Classes of single parallelepipeds	L1 side (units)	L2 side (units)	L3 side (units)
A	4	4	4
B	2	3	5
C	1	3	6
D	1	2	6
E	1	3	3
F	1	2	2

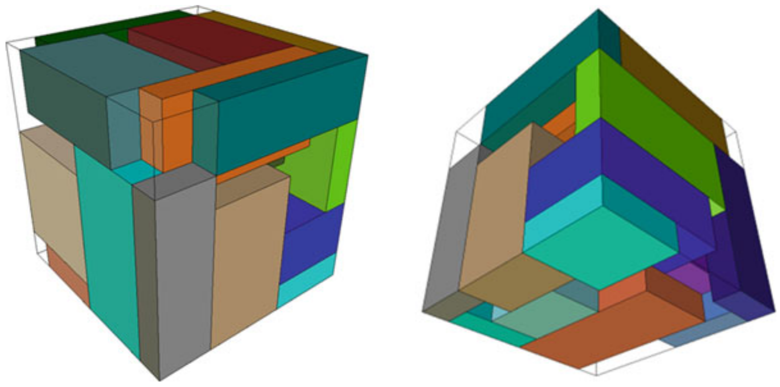

Fig. 5.4 Case Study 2.1

Some interesting case studies, relative to the second linear reformulation for solving the *feasibility* subproblem (Sect. 3.1.2), are reported hereinafter.

Table 5.1 reports six classes of parallelepipeds adopted to execute some tests considered.[1] Case Study 2.1 Contemplates 22 items, extracted from the table: 1 of type A, 6 of B, 6 of C, 5 of D and 4 of E. The domain consists of a cube of eight units. The solution depicted in Fig. 5.4 was found in 30 CPU seconds. The occupied volume reaches 87.5 % of the total available.

Cases Studies 2.2, 2.3 and 2.4 follow. Case 2.2 includes two *separation* planes. Tables A.3, A.4 and A.5, in the Appendix, report, for each case, the item dimensions, whilst Table A.6 those of the relevant domains. The results obtained are summarized here below in Table 5.2. Case Study 2.4 is represented graphically in Fig. 5.5.

The non-restrictive reformulation of the general MIP model reported in Sect. 3.1.3 seems quite suitable to solve the problem directly, when not too large-scale

[1] These classes refer to quite a difficult instance proposed by Jürgen Rietz (Dept. Produção e Sistemas, Centro de Investigação Algoritmi da Universidade do Minho, Escola de Engenharia, Universidade do Minho, 4710–057, Braga, Portugal). It consists of the following: given a cube of 8 units, load 1 item of type A and 6 for all of the remaining types.

Table 5.2 Results of Case Studies 2.2 to 2.4

Case studies	Total number of single parallelepipeds	Loaded volume % (rounded to nearest)	CPU time (s)
Case Study 2.2	31	80.9	2,659
Case Study 2.3	25	84.6	549
Case Study 2.4	17	90.5	834

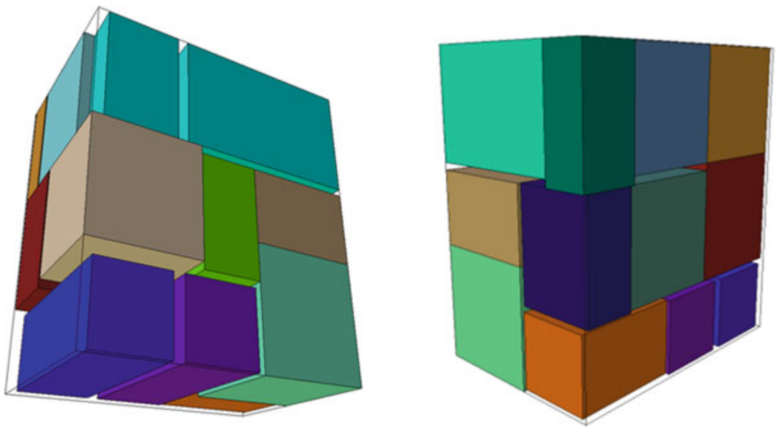
Fig. 5.5 Case Study 2.4

instances are involved. As is easily seen, this reformulated model, differently from the heuristic approaches considered in Chap. 4, allows the optimizer, at least theoretically, to find and to prove the actual optimal solution.

An experimental analysis in this direction is currently ongoing. A significant effort is still expected to confirm the apparent advantage of the use of this reformulation, both as a stand-alone model and in support of the heuristic approaches.

Some successful exercises are reported here. A first indication can be provided by reconsidering Case Studies 2.2–2.4, as solved by the non-restrictive reformulation in question. The same results were obtained in terms of volume occupation, packing all the given items, even if no impositions were made on their loading. Different outcomes, nonetheless, arose, concerning the computational effort: Case Study 2.2 was solved in 280 CPU seconds, Case Study 2.3 in 370 CPU seconds and Case Study 2.4 in 313 CPU seconds (the information available to date, relevant to the non-restrictive reformulation, is however not sufficient to confirm this apparently outperforming trend).

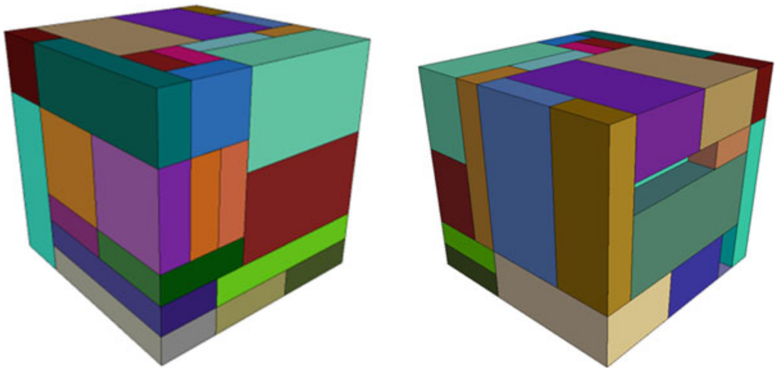
Fig. 5.6 Case Study 3.1

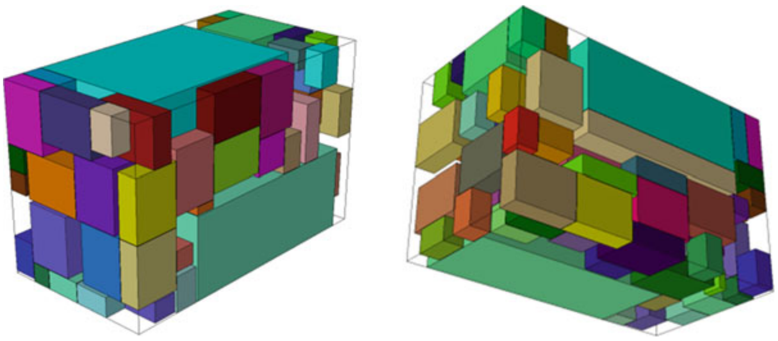
Fig. 5.7 Case Study 3.3

Case Study 3.1 refers to the instance derived from Table 5.1, involving 1 item of class A, 6 of B, 6 of C, 6 of D, 6 of E and 6 of F, considering, as a domain, a cube of 8 units (cf. Note 2). An optimal solution (loading all the given 31 items) was found in 995 CPU seconds. The occupied volume is 98 % of the available one. It is depicted in Fig. 5.6.

Input data relevant to the following Case Studies 3.2 and 3.3 are reported in Tables A.7 and A.8; see Appendix. An optimal solution, including all the 51 items, was found in 757 CPU seconds for Case Study 3.2. The occupied volume is 82.7 % of the available one. An optimal solution, including all the 84 items, was found in 2,727 CPU seconds for Case Study 3.3. The occupied volume is 80 % of the available one. It is worth noticing that the relative instance contains 27,902 constraints and 22,261 variables, of which 21,756 are binary. It is represented by Fig. 5.7.

5.3 Use of the Linear Reformulations to Obtain Approximate Solutions

In the following, some insights are provided, concerning the (*LP-relaxed*) reformulations of Sects. 4.3.1.1 and 4.3.1.2, respectively, both aimed at finding approximate solutions (as initialization steps). Case Studies 2.2–2.4 (cf. Sect. 5.2) are reconsidered here as reference instances.

The first linear reformulation has been utilized to find an approximate solution to Case Study 2.3. The process took only 2 CPU seconds, but 38 intersections were identified (out of 300 pairs of items). The overlap volume is 38.6 % of that associated to the totality of items to load. The graphical results are represented by Fig. 5.8.

The second linear reformulation was adopted by dropping inequalities (3.6) (of Sect. 3.1.2) and including (4.2) (as suggested in Sect. 4.3.1.2). An approximate solution to Case Study 2.4 was obtained within a time limit of 300 CPU seconds, with 23 intersections (out of 136 pairs of items) and 12.7 % of overall volume overlap, considering the actual parallelepipeds. It is shown in Fig. 5.9 (with the occurring intersections). The variation of the second linear reformulation of Sect. 3.1.2 (substituting (3.6) with (3.8), cf. Sect. 4.3.1.2) was, instead, considered for Case Study 2.2. The relative solution (suboptimal for this reformulated model) is represented in Fig. 5.10 (with the occurring intersections). It was found within a time limit of 300 CPU seconds. The number of identified intersections (with respect to the actual items) is 11 (out of 528 pairs of items, considering also the *separation* planes as such), corresponding to 5.5 % of the overlap volume.

The three study cases in question suggest that the first linear reformulation provides quick but quite imprecise solutions, whilst the two versions of the second reformulation are more time consuming but offer better results. This trend seems to be confirmed by the comparative analysis currently ongoing, but a further in-depth experimentation is certainly needed. The approach adopted for Case Study 2.2 presents, for the time being, promising perspectives. From the preliminary outcomes available, indeed, it is usually able to find a number of *integer-feasible* (suboptimal) solutions quite easily.

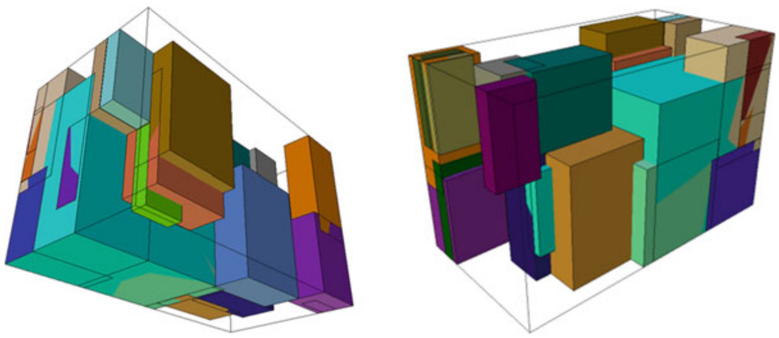

Fig. 5.8 Case Study 2.3 approximate solution (obtained by the first linear reformulation)

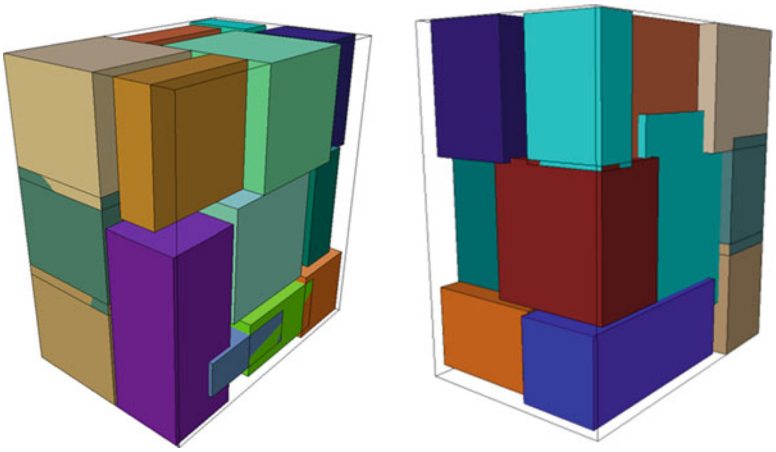

Fig. 5.9 Case Study 2.4 approximate solution (obtained by the second linear reformulation)

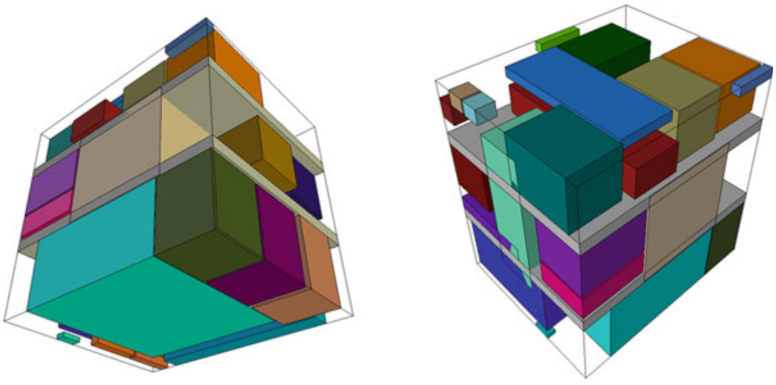

Fig. 5.10 Case Study 2.2 approximate solution (obtained by the second linear reformulation variation)

5.4 Nonlinear Reformulation Approach to Improve Approximate Solutions

A dedicated experimental analysis, focusing on the use of the nonlinear reformulation referred to in Sect. 4.3.1.3, with *objective* function (3.13), is currently ongoing (Fasano and Castellazzo 2013). As outlined there, this approach can be utilized to improve the approximate solutions, obtained either by the first or second linear reformulations; cf. Sects. 4.3.1.1 and 4.3.1.2, respectively (or by any alternative initialization process).

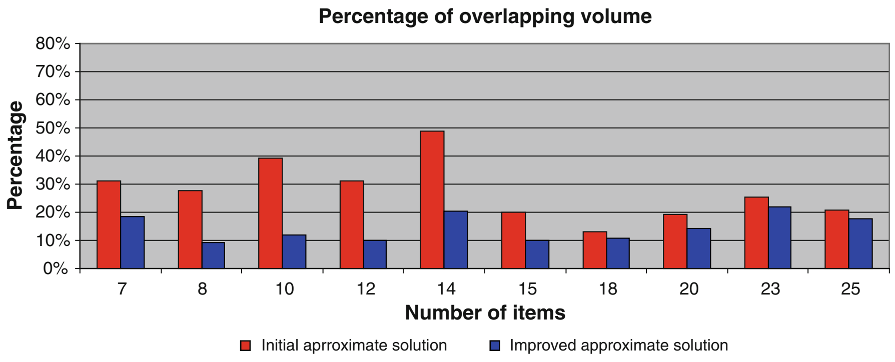

Fig. 5.11 Initial overlap volume reduction

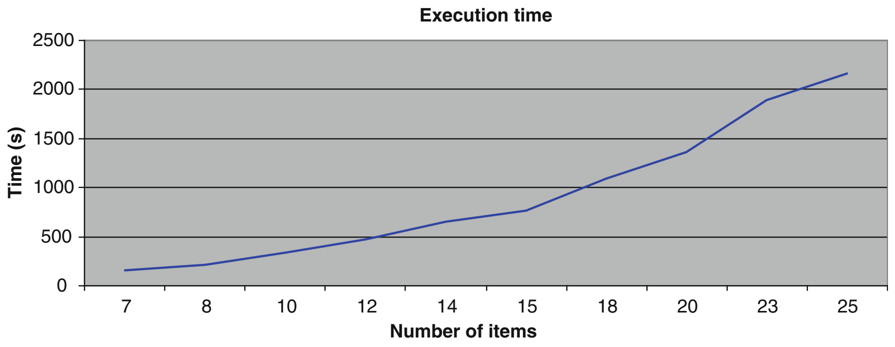

Fig. 5.12 Computational effort

Some preliminary outcomes are briefly reported here. A first rough trend estimate, relative to a sample of 35 case studies (considering, for the time being, just quite a limited number of single parallelepipeds), is suggested by Fig. 5.11 that shows different groupings, based on the number of items involved. For each group, the (average) percentage of overlap volume is displayed, with respect to the initial approximate solution (left column) and the improved one (right column). Some indications, concerning the computational effort, are given in Fig. 5.12. Such an effort is expected to decrease in the near future, by improving the optimization strategies adopted.

Three case studies (4.1–4.3) involving 7, 12 and 18 items are depicted by Figs. 5.13, 5.14 and 5.15, respectively (representing the initial solutions on the left and the improved ones on the right).

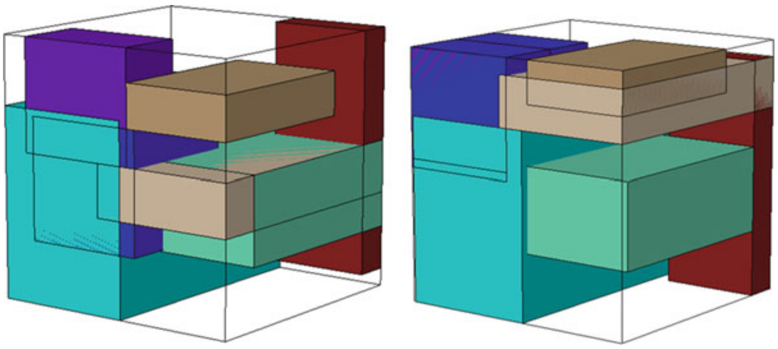
Fig. 5.13 Case Study (4.1) with 7 items

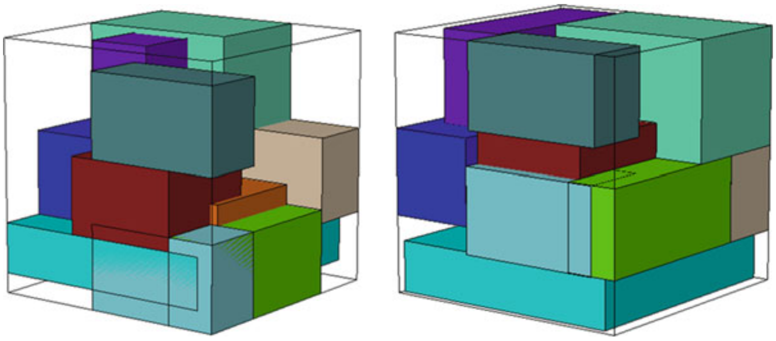
Fig. 5.14 Case Study (4.2) with 12 items

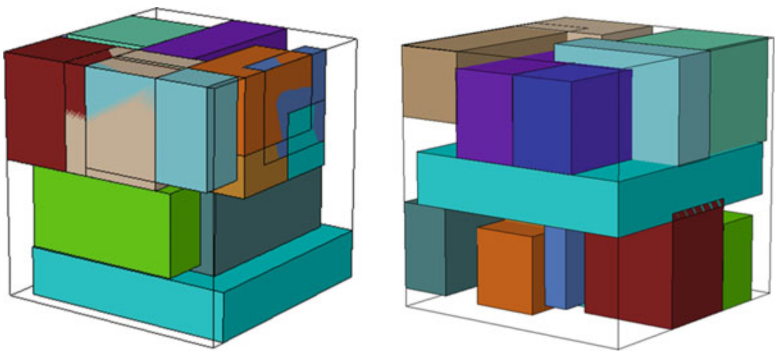
Fig. 5.15 Case Study (4.3) with 18 items

5.5 The Use of Heuristics

This section is devoted to providing some insights on the use of the heuristic approaches proposed in Chap. 4. A significant number of real-world packing issues (more or less complicated, in terms of additional conditions) have been solved successfully in the space engineering context that gave rise to this work (in the framework of the International Space Station, ISS, cf. http://www.nasa.gov, in particular within the CAST project; see (Fasano et al. 2009)). Nonetheless, a substantial commitment is still expected to consolidate the outcomes available to date.

As previously pointed out, differently from other methods, the modeling-based ones, proposed in this volume, try to solve the relevant packing models, taking into account, contemporarily, all the items (or at least subsets of the original instance). This offers the evident advantage of providing a global point of view, allowing, if necessary, for the introduction of overall (i.e. 'transversal') conditions, such as balancing. On the other side, it is easily seen that solving such (MIP/MINLP) models is generally much more complex than carrying out packing algorithms based on sequential placement. It is hence quite obvious that if no additional conditions have to be taken into account, then most non-modeling-based approaches (e.g., Martello et al. 2000) are expected to outperform the modeling-based ones discussed in this monograph.

In addition to the above considerations, a non-trivial issue comes up, since a practical threshold, concerning the scale of the instances to face, is understood. The number of items involved becomes, as a matter of fact, a first limiting factor. It, indeed, directly affects the instance size (that, when the model is formulated in terms of MIP, corresponds to that of the relative matrix), as well as the number of binary variables. Nevertheless, even when the given instance is, as a matter of fact, too large to cope with, a partition into subproblems can be carried out. Quite a successful approach of this type has been applied, in the space engineering context, to the Automated Transfer Vehicle (ATV, ESA; cf. http://www.esa.int). Its extremely challenging cargo accommodation problem has been tackled by developing an ad hoc packing optimization system, decomposing the overall problem at different levels (see Fasano et al. 2009).

5.5.1 Test Instances from the Literature

The description of instances containing a significant number of *tetris*-like items, with several *components* each, represents quite a heavy task indeed. For this reason and in order to provide the reader with quite an easy-to-access experimental framework, it has been decided to concentrate on standard instances.

A dedicated test campaign has been fulfilled for the first heuristic procedure presented in Sect. 4.3.2 (at present, the most consolidated approach, from the

Table 5.3 Results of 670 case studies (heuristic procedure of Sect. 4.3.2)

	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7
Average volume exploitation (%)	78.13	78.48	79.60	79.75	80.04	80.05	80.01
Worst case volume exploitation (%)	69.13	68.94	71.64	72.25	72.56	71.17	72.60
Best case volume exploitation (%)	86.68	85.41	86.79	86.67	86.67	86.61	86.99
Average loaded items	83	80	79	78	80	78	77
Worst case loaded items	42	45	50	47	53	52	56
Best case loaded items	138	122	123	130	118	107	105

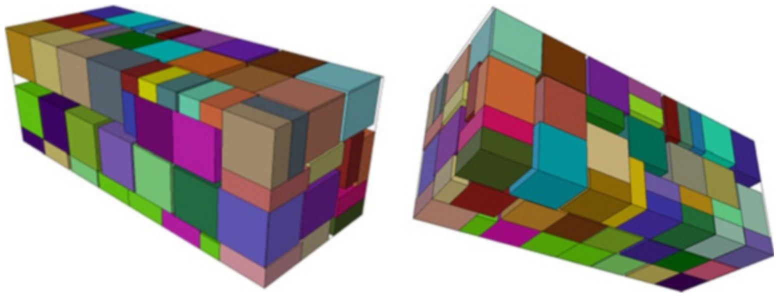
Fig. 5.16 Case Study 5.1.43

experimental point of view), referring to the 'Three Dimensional Cutting and Packing Data Sets—THPACK 1–7 BR' (Bischoff and Ratcliff 1995): http://www.euro-online.org/web/ewg/25/esicup-euro-special-interest-group-on-cutting-and-packing. As it is known, this test bed consists of 7 sets of 100 instances each. They are indicated in the following as Case Studies '5.s.n', where 's' is the progressive number of the set and 'n' the index of the test problem instance. 670 instances out of the 700 available have been tested (excluding all those exceeding 200 items).

Table 5.3 reports the relevant results. All the tests were executed within the limit of 1 h of CPU time. Figure 5.16 shows as an indicative example, Case Study 5.1.43 solution, in which 86.68 % of the available volume has been exploited (loading 94 items out of 141).

The experimental activity relevant to the heuristic procedure described in Sect. 4.3.3 is currently being carried on. Since the computational effort strongly depends on the overall solution strategy followed, it is worth providing some insights on the relevant performances, at each single-phase level. On the basis of the experience acquired to date, Table 5.4 provides quite a consolidated general trend, referring to the process steps separately.

As far as the alternative approach suggested in Sect. 4.3.3 is concerned, a further set of 24 tests from 'Three Dimensional Cutting and Packing Data Sets—THPACK 1–7 BR' was executed, namely, 5.1.17, 5.1.39, 5.1.67, 5.1.68, 5.1.76, 5.1.91, 5.1.100; 5.2.4, 5.2.13, 5.2.39, 5.2.59, 5.2.77, 5.2.79, 5.2.85, 5.2.96; 5.3.39, 5.3.56, 5.3.59, 5.3.77; 5.4.39, 5.4.56, 5.4.79; 5.5.56; 5.6.13.

Table 5.4 Computational trend at single-step level (heuristic procedure of Sect. 4.3.3)

Steps	Involved items	CPU time estimates (s)
Initialization	75–100	45–90 (recursive mode)
Abstract configuration generation	75–100	<5
Packing	75–100	30–60
Hole-filling	10–15	<15
Item exchange	10–15	<5

Table 5.5 Results of 24 case studies (heuristic procedure of Sect. 4.3.3)

Case studies	Total no of items (parallelepipeds)	No of loaded items exploiting 75 % of the available volume	CPU time to reach the 75 % of the available volume
5.1.17	213	116	00:16:36
5.1.39	243	166	00:13:23
5.1.91	238	88	00:07:30
5.1.100	214	142	00:32:41
5.2.4	201	94	00:08:05
5.2.13	228	109	00:22:10
5.2.79	206	128	00:49:53
5.2.85	209	121	00:22:50
5.2.96	202	120	00:31:09
5.3.56	212	112	00:13:49
5.3.77	201	131	01:03:34
5.4.56	233	130	00:31:49
5.4.79	217	119	00:57:58

For all these tests the number of items available is within the range between 201 and 275. To solve this large-scale instance successfully, an ad hoc solution strategy was thought up. A basic cycle, consisting of the following sequence of steps was introduced: *initialization*, *packing*, *item-exchange* and *hole-filling*. A number of basic cycles were allowed to be executed, by extending incrementally the set of items involved, until 75 % of the occupied volume was reached. Afterwards, a final cycle consisting of *hole-filling* steps only was admitted. Within every cycle, each of the above steps was allowed to be repeatedly fulfilled, following appropriate stopping rules. The process had (about) 1 CPU hour as a time limit. Overall results are reported here below:

- Average volume = 76.31 %
- Worst case volume exploitation = 61.19 %
- Best case volume exploitation = 85.41 %
- Average loaded items = 141
- Worst case loaded items = 103
- Best case loaded items = 182

Table 5.5 shows the CPU time requested to attain (when reached) 75 % of volume exploitation and the corresponding number of items loaded.

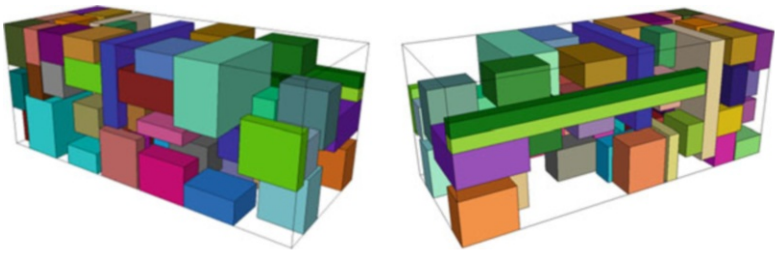

Fig. 5.17 Case Study 6—third cycle solution (50 % of exploited volume)

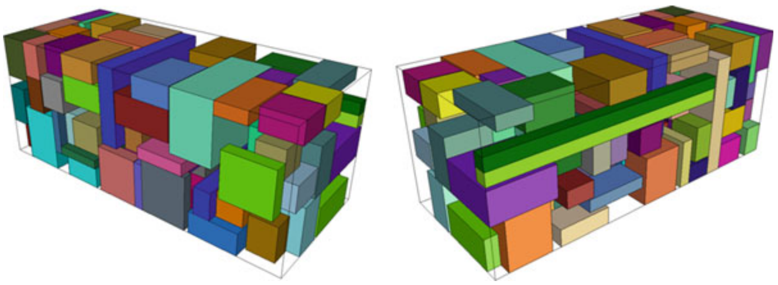

Fig. 5.18 Case Study 6—fifth cycle solution (65 % of exploited volume)

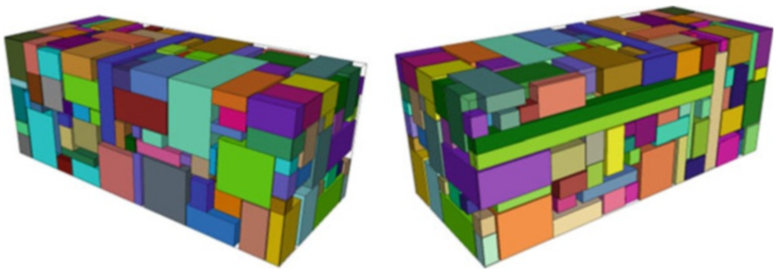

Fig. 5.19 Case Study 6—solution obtained in 1 CPU hour (85.77 % of exploited volume)

5.5.2 *Instance Adopted to Tune the Solution Strategy*

A ‘fabricated’ instance (Case Study 6) was introduced (in addition to others) to tune the solution strategy outlined above. Representing an interesting exercise solved successfully, it is briefly considered hereinafter. This instance consists of the packing of up to 280 parallelepipeds into a parallelepiped of dimensions 540, 225 and 220 units, respectively, maximizing the loaded volume. Results obtained, with 1 CPU hour as a time limit, throughout the whole procedure are illustrated in Figs. 5.17, 5.18 and 5.19, showing different solution levels. Relevant details are reported in Table A.10; see Appendix.

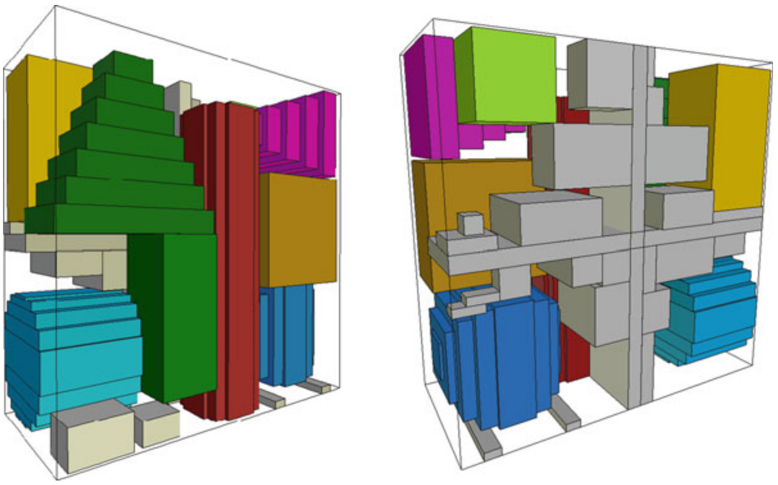

Fig. 5.20 Case Study 7.1 (with *structural* elements and forbidden zones)

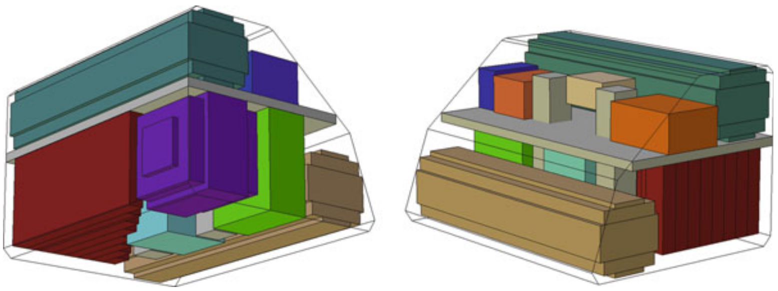

Fig. 5.21 Case Study 7.2 (with a curved domain, a separation plane and *structural* elements)

5.5.3 Close-to-Real-World Instances

Before concluding this section, two similar-to-real-world instances (Case Studies 7.1 and 7.2) successfully solved, in support of the ISS and ATV logistics, are shown in the following; see Figs. 5.20 and 5.21 (the relevant technical details are kept confidential). They were solved by utilizing the heuristic procedure of Sect. 4.3.2, requiring about 500 CPU seconds.

Further applications involving *tetris*-like items are considered in Sect. 6.1.3.