A Fast Optimization Approach For A Complex Real-Life 3D Multiple Bin Size Bin Packing Problem

Katrin Heßler[A], Timo Hintsch[A], Lukas Wienkamp[A]

[A] *Global Data & AI, F.LCA - Global Technology and Data, Schenker AG, Kruppstraße 4, 45128 Essen, Germany.*

Abstract

We investigate a real-life air cargo loading problem which is a variant of the three-dimensional Variable Size Bin Packing Problem with special bin forms of cuboid and non-cuboid unit load devices (ULDs). Packing is constrained by additional practical restrictions, such as load stability, (non-)stackable items, and weight distribution constraints. To solve the problem, we present an insertion heuristic embedded into a Randomized Greedy Search. The solution space is limited by only considering certain candidate points (so-called extreme points), which are promising positions to load an item. We extend the concept of extreme points proposed in the literature and allow moving extreme points for non-cuboid ULDs. A special sorting of the items is suggested, which combines a layered structure and free packing. Moreover, we propose dividing the space of each ULD into smaller cells to accelerate the collision, non-floating, and stackability check while loading items. In a computational study, we analyze individual algorithm components and show the effectiveness of our method on adapted real-life instances from the literature.

Key words: packing, air transport, three-dimensional bin packing, heuristics

Email addresses: katrin.hessler@dbschenker.com (Katrin Heßler), timo.hintsch@dbschenker.com (Timo Hintsch), lukas.wienkamp@dbschenker.com (Lukas Wienkamp)

1. Introduction

In the air freight industry, there are several steps from the arrival of individual shipments at the terminal to the fully loaded and ready-to-take-off aircraft. One major step of the operational planning is to load cargo onto special air cargo pallets or containers, so-called *unit load devices* (ULDs), which are then loaded into the aircraft. Air cargo pallets must be covered with a net and straps to secure their cargo. For containers no nets or straps are required. Both ULD types, pallets and containers, can be modeled as bins with a shape that varies with respect to the designated position in the aircraft. When loading cargo into ULDs, the aim is to achieve a high utilization of the ULDs and at the same time ensure load stability. Assuming that the cargo consists of a set of cuboid items, the problem can be modeled as a three-dimensional *Multiple Bin Size Bin Packing Problem* (MBSBPP), according to the typology of Wäscher *et al.* (2007).

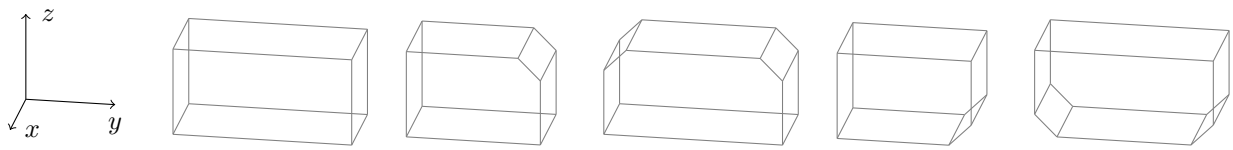
The three-dimensional MBSBPP is a generalization of the Bin Packing Problem (Garey and Johnson 1979) and, therefore, NP-hard. The aim of this work is to develop a solution algorithm that delivers high-quality solutions in short computational times, suitable to be used in operational planning. Due to the problem complexity, we present a greedy-based insertion heuristic and refrain from an exact approach. The problem we consider is a slight variation of the problem presented in Paquay *et al.* (2016), in which the authors propose a mixed-integer program. Other algorithmic approaches are a Relax-and-Fix algorithm and derived matheuristics (Paquay *et al.* 2018a) and a two-phase insertion heuristic (Paquay *et al.* 2018b).

The contributions of this paper are the following:

- We present a new variant of the three-dimensional MBSBPP that takes into account additional constraints and problem characteristics that occur in real-world air transportation situations: the stability of the load, avoiding unused space between items, (non-)stackable items, weight distribution, specially shaped, non-cuboid ULDs, the usage of padding material, reserved edge space on pallets for cargo net locks on which loading is prohibited, and the usage of a substructure to allow for loading large items at the bottom exceeding the ULD edge. The problem extends the one presented in Paquay *et al.* (2016), which does not consider the last three points.

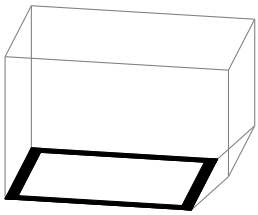
- We suggest a new greedy-based insertion algorithm that sequentially loads the available ULDs. Each ULD is loaded by a Randomized Greedy Search that repeatedly calls an insertion heuristic based on so-called extreme points (Crainic *et al.* 2008), which are promising candidate positions to load an item. The framework is similar to the one proposed by Gajda *et al.* (2022). However, our approach differs from theirs and other approaches as it ensures very fast iterations thanks to various acceleration techniques: A three-dimensional grid, dividing the space of each ULD into smaller cells, is used to accelerate the collision, non-floating, and stackability check while loading items. Moreover, instead of evaluating all generated extreme points with a merit function, a first-fit approach is applied. The resulting very low runtime of the insertion heuristic allows a large number of iterations to be carried out. Note that (Crainic *et al.* 2008) already tested first-fit heuristics but did not perform several iterations with randomly sorted items. A high solution quality is achieved by using an extended set of extreme points and a special sorting of the items which, together with the first-fit approach, combines a layered structure and free packing. Our new approach outperforms the two-phase insertion heuristic of Paquay *et al.* (2018b) with respect to utilization while offering similar solution quality regarding the weight balance. Moreover, our algorithm shows good results for classical three-dimensional Bin Packing Problem instances (Bischoff and Ratcliff 1995).

- New insights on extreme points, that can also be applied to other three-dimensional packing problems, are presented: We allow the moving of extreme points to handle specially shaped, non-cuboid ULDs. This ensures a high loading density underneath the beveled ULD sides at the top. Moreover, we extend the concept of extreme points proposed by Crainic *et al.* (2008) by adapting the projection routine that generates new extreme points. We show that the generation is quadratic in the number of loaded items, which is worse than the linear complexity of generating the extreme point set of Crainic *et al.* (2008). However, the algorithm is still very fast, as the worst-case complexity is almost never reached.

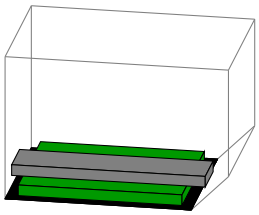


(a) (b) (c) (d) (e) (f)

Figure 1: The underlying coordinate system and different ULD shapes relevant in practice.



(a) Example of a ULD with an edge (in black) on which loading is prohibited.



(b) Example of a ULD with an edge (in black), a substructure (in green), and an item loaded above the ULD's edge (in gray).

Figure 2: Two examples of ULDs with an edge and with/without a substructure.

The remainder of the paper is organized as follows: In the next section, we formally define the three-dimensional MBSBPP. In Section 3, we give an overview of the existing literature on three-dimensional MBSBPPs. The insertion heuristic to load a single ULD is introduced in Section 4. In Section 5, the heuristic is embedded into a Randomized Greedy Search. To load multiple ULDs, the framework is extended further in Section 6. An analysis of individual algorithm components and computational results on various benchmark sets are then reported in Section 7. Section 8 draws final conclusions and discusses future research directions.

2. Problem definition

We first provide some practical insights before formally defining the problem. The underlying coordinate system is depicted in Figure 1a. Note that the length, width, and height correspond to x, y, and z, respectively.

A set of cuboid items must be loaded into a set of ULDs. Some items can be rotated orthogonally and/or tilted while others cannot. There are six possible orientations (rotated: yes/no, tilted: no/across the x-axis/across the y-axis) for rotatable and tiltable items (Paquay *et al.* 2018b; Figure 2). Tiltable items are always rotatable. Furthermore, some fragile items, for example those whose outside is a cardboard, are not stackable, *i.e.*, nothing is allowed to be placed on top.

In case the cargo is loaded onto a pallet (and not into a closed container), the available space for loading is not only defined by the outer shape of the ULD but further restricted. The reason for this is that the cargo must be covered with a net which is attached to the edge of the pallet via net locks. It is not possible to load cargo directly onto these locks. However, if an item is sufficiently far above the ULD edge, for example because it is stacked on top of other items, it may overlap with the edge as access to the locks is guaranteed. To allow for the loading of large items at the bottom that can only be loaded if placed above the ULD's edge, a substructure can be used to artificially raise the ULD's base area. Figure 2 illustrates the ULD's edge and the use of a substructure.

Load stability is crucial to avoid damages during transportation. To prevent items from sliding, it is preferable to avoid holes between items in horizontal direction. To fill small vertical gaps and allow for sufficient support of items, padding material can be placed on top of (stackable) items and the ULD floor.

(a) Feasible region for the CoG.

(b) The CoG is marked with a black square ■. The CoG in x- and y-direction is marked with a red bullet •, respectively. In this example, the CoG is outside the feasible region in x- and y-direction.

Figure 3: Example of how to determine the CoG in x- and y-direction.

We refer to the part of the item that is supported by padding material as *indirectly supported*. The part of the item that rests on another stackable item or the floor is *directly supported*.

It is required that the ULD's *center of gravity* (CoG) in x- and y-direction falls into a pre-defined area around its geometric center (see Figure 3). We assume that the weight of all items is equally distributed so that the center of gravity of each item is its geometrical center.

We can formally define the problem as follows.

Input

Items. A set I of cuboid items. For each item $i \in I$, we have
- size $s^i = (s^i_1, s^i_2, s^i_3)$ with length s^i_1, width s^i_2, height s^i_3,
- weight w_i,
- is rotatable $r_i \in \{0, 1\}$,
- is tiltable $t_i \in \{0, 1\}$,
- is stackable $\vartheta_i \in \{0, 1\}$.

ULDs. A set C of ULDs. For each ULD $c \in C$, we have
- set of vertices V_c,
- set of facets F_c where each facet $f \in F_c$ consists of the vertices defining that facet ($f \subseteq V_c$),
- weight capacity W_c,
- volume capacity $\mathcal{V}_c$,
- edge width e_c,
- vertical edge offset δ_c (minimum vertical distance between an item and the ULD edge to allow for edge overlap),
- use of substructure allowed $\varsigma_c \in \{0, 1\}$.

Packing parameters. Additional parameters defining what constitutes a feasible solution:
- maximum padding height $\hbar$ (maximum height of a gap that can be filled with padding material).
- minimum item overlap o (minimum portion of an item's base area that must be supported by other items, padding material, or the floor).
- maximum center of gravity deviation from a ULD's geometric center $cog^{\max}$.

Constraints

ULD fit. Each loaded item must lay within the ULD.

ULD edge. Items must not be placed on the ULD edge and can only overlap with the edge at a height of at least δ.

Collisions. No two loaded items must collide with each other.

Floating. Each loaded item must sufficiently be supported by either the ULD floor, other loaded items, or padding material. Here, sufficiently supported means that

(i) the item has any level of direct support (*i.e.*, at least one part of the item rests on another item or the floor) and the percentage of the item's base area that is directly or indirectly supported (*i.e.*, rests on another item, padding material, or the floor) amounts to at least the minimum item overlap o, or
(ii) the item's four bottom corner points rest on items or the ULD floor.

Stacking. Neither items nor padding material must be loaded on top of a non-stackable item.

Weight distribution. The CoG of the ULD load must not deviate more than $cog^{\max}$ from the ULD's geometric center.

Objective

Respecting all packing restrictions, the task is to (1) maximize the volume of loaded items and (2) determine a volume-minimal set of ULDs. Both objectives are optimized in a strictly lexicographic sense.

Supported ULD shapes

The definition of a ULD via a set of vertices and facets allows for arbitrary shapes to be expressed. However, only few of them are relevant in practice. Hence, we make the following assumptions on the supported ULD shapes, which are also exploited by the algorithm. Each ULD consists of at least six facets, with exactly two facets parallel to the x-y, x-z, and y-z plane, respectively. We refer to these facets as *non-tilted facets*. In addition, some ULDs may have a special shape with up to two *tilted facets* to fit into the fuselage of the aircraft. These are facets that are not parallel to the x-y, x-z, or y-z plane but orthogonal to the y-z plane. No neighboring tilted facets are allowed, *i.e.*, corner vertices of tilted facets must differ. Also, we assume that each ULD is convex. Figure 1 gives an overview of outer ULD shapes that are relevant in practice.

Further notation

We introduce the following notation, that will be used throughout the remainder of the paper.

- We denote an item's i orientation by

$$\sigma_i = (\text{tilt, rotation}) \in \Theta, \text{where } \Theta = \{\text{not tilted, tilted across } x\text{-axis, tilted across } y\text{-axis}\} \times \{0, 1\}.$$

  Here, the tilt is always applied first, *i.e.*, refers to the item's original size/orientation, and the rotation to the already tilted item. When we apply a certain orientation to an item i, we update its size $s^i = (s^i_1, s^i_2, s^i_3)$ accordingly.
- We identify each facet $f \in F_c$ with its plane equation $n^f_1 x + n^f_2 y + n^f_3 z = a^f$, where the normal vector n^f is assumed to point towards the inside of the ULD.
- We denote the index set of dimensions/directions as $D = \{1, 2, 3\}$ corresponding to $\{x, y, z\}$.
- When we load an item i, its *position* e^i refers to its corner point closest to the coordinate origin, *i.e.*, the corner with the minimum x-, y- and z-coordinates. We refer to the corner point furthest away from the coordinate origin, *i.e.*, the one with the maximum x-, y- and z-coordinates, as *end position*.
- Whenever we focus on only one item $i \in I$ or one ULD $c \in C$, we drop the index i/c from all item/ULD related parameters.

3. Literature review

In the literature, several variants of the three-dimensional MBSBPP have been discussed. In the following, we first provide an overview of constraints relevant in practice and the corresponding three-dimensional packing problems. Afterwards, we focus on heuristic solution approaches for such rich packing problems. With regard to packing problems with practical constraints in the air freight industry, we refer to Bortfeldt and Wäscher (2013); Zhao *et al.* (2014); Brandt (2017); Brandt and Nickel (2019) for a summary of the topic and related problems.

Bischoff and Ratcliff (1995) provide a first overview of packing requirements relevant in practice. Among others, they mention container weight capacity, weight distribution, grouping of items, load stability, and multiple item destinations (multi-drop). The paper focuses on container loading problems, however, several of the restrictions are also relevant for pallet building problems and for loading ULDs in the air freight industry. Table 1 provides an overview of various problem definitions. We have limited the selection to problems considering weight distribution and/or stackability constraints.

Table 1: Overview of real-world three-dimensional packing problem definitions and constraints.

category	problem feature	Bischoff and Ratcliff (1995)	Ratcliff and Bischoff (1998)	Davies and Bischoff (1999)	Terno *et al.* (2000)	Chan *et al.* (2006)	Baldi *et al.* (2012)	Junqueira *et al.* (2012)	Ceschia and Schaerf (2013)	Paquay *et al.* (2016)	Trivella and Pisinger (2016)	our approach
type		CL	CL	CL	PL	AC	KP	CL	CL	AC	BP	AC
bin/ULD	weight capacity	x*			x	x‡			x	x		x
	weight distribution	x*		x	x	x	x	x		x	x	x
	multiple bin sizes					x			x	x		x
	non-cuboid ULDs					x				x		x
	ULD edge											x
item	orientation constraints	x	x	x	x	x	x	x	x	x		x
	stackability	x								x		x
	load bearing capacity		x		x			x	x			
packing	multi-drop	x							x			
	grouping	x*			x	x						
	filling material											x

CL: Container Loading, PL: Pallet Loading, AC: Air Cargo, KP: Knapsack Problem, BP: Bin Packing.
*: Mentioned but not considered in solution approach. ‡: Indirectly considered by objective function.

Weight capacity constraints are relevant in most real-world problems. Weight balance constraints, which ensure the center of gravity falls (as close as possible) to an ideal point or a certain area, are considered in the context of container loading problems (Davies and Bischoff 1999), pallet building problems (Terno *et al.* 2000), air cargo loading problems (Chan *et al.* 2006; Paquay *et al.* 2016), the three-dimensional knapsack problem (Baldi *et al.* 2012), and bin packing problem (Trivella and Pisinger 2016). In practice, these restrictions are important, for example, when lifting a container with a crane or transporting pallets with a forklift. Note that weight balance constraints should not be confused with axle weight limits of trucks (Pollaris *et al.* 2016) or an as evenly as possible distributed weight along truck axles (Gajda *et al.* 2022). Multiple bin sizes

are often considered in air cargo loading problems. The same holds for special non-cuboid bin shapes that are only relevant in air cargo loading problems (Paquay *et al.* 2016; Chan *et al.* 2006). To the best of our knowledge, we are the only ones considering the ULD edge. Some approaches also consider very specific cost functions (Chan *et al.* 2006). However, we do not go into further detail because our approach simply maximizes the volume of loaded items.

Due to its content, an item may only allow a limited number of orientations. In Trivella and Pisinger (2016); Junqueira *et al.* (2012), rotating items is not allowed. Moreover, it can be specified whether an item is stackable, *i.e.*, not fragile. A more detailed stackability definition is proposed by Ratcliff and Bischoff (1998) that define a maximum load bearing capacity depending on the orientation of the item. Other load bearing capacity definitions are introduced in Terno *et al.* (2000); Junqueira *et al.* (2012); Ceschia and Schaerf (2013). Multi drop scenarios, *i.e.*, situations in which items must be delivered to different destinations, generate further restriction on feasible packings. Here, loading patterns that allow for access to the items for the earlier destinations without moving other items around are required. This typically results in solutions following a vertical wall-/stack-building approach to ensure that items with the same destination can be placed on top of each other to avoid spreading them out across the bin floor (Bischoff and Ratcliff 1995; Ceschia and Schaerf 2013). Ensuring that pre-defined groups of items are loaded in the same bin is considered in Terno *et al.* (2000); Chan *et al.* (2006). The problem definition that comes closest to ours is the one of Paquay *et al.* (2016), which consider the same problem without the ULD edge and filling material.

Most of the heuristic approaches rely on the seminal works of Martello *et al.* (2000); Crainic *et al.* (2008) that introduce ways to generate promising positions to load items. Martello *et al.* (2000) propose the concept of so-called corner points and, based on that, develop an exact branch-and-bound algorithm to fill a single bin. Later on, Crainic *et al.* (2008) extend the idea of corner points to the concept of extreme points allowing for the creation of additional candidate points for loading items. While in Martello *et al.* (2000) all corner points have to be recalculated when an additional item is loaded, Crainic *et al.* (2008) suggest a procedure that only needs to calculate additional extreme points upon loading an item. Hence, despite the potential larger number of candidate points, the complexity of updating the extreme points is only $\mathcal{O}(|L|)$ whereas recalculating the corner points has $\mathcal{O}(|L|^2)$ complexity (where $|L|$ denotes the number of already loaded items). Crainic *et al.* (2008) present a constructive heuristic based on the extreme point concept.

Several other heuristics rely on local search. Lodi *et al.* (2002; 2004) present a unified tabu search framework *TSpack*, that is applicable to various multi-dimensional bin-packing problem variants. Faroe *et al.* (2003) propose a guided local search that iteratively removes one bin from a feasible solution. It uses memory to guide the search to promising regions of the solution space. A two-level tabu search TS^2pack is proposed by Crainic *et al.* (2009), where the first level aims to reduce the number of bins and the second level optimizes the packing of the bins. A local search approach for a multi-drop multi-container loading problem is introduced by Ceschia and Schaerf (2013). Trivella and Pisinger (2016) present a multi-level local search heuristic for a load-balanced multi-dimensional bin-packing problem. The algorithm uses an implicit representation of multi-dimensional packings by means of interval graphs (Fekete and Schepers 2004) and iteratively improves the load balancing using different search levels.

Parreño *et al.* (2008) present a *Greedy Randomized Adaptive Search Procedure* (GRASP) that is based on a constructive block heuristic using the concept of maximal free space. Later, the authors propose a hybrid GRASP and Variable Neighborhood Descent algorithm (Parreño *et al.* 2010), where the constructive phase is based on their previous work. In Alvarez-Valdes *et al.* (2013), the best solutions found by a GRASP algorithm are combined into a Path Relinking procedure to intensify and diversify the search. Other heuristic approaches for three-dimensional packing problems can be found in Chan *et al.* (2006); Egeblad and Pisinger (2009); Baldi *et al.* (2012).

The solution approach that is closest to ours is the randomized constructive heuristic of Gajda *et al.* (2022) that combines items, sorts the combined items, partially perturbs the sorting randomly, and finally constructs the packing. Within the constructive packing phase, the location at which an item is placed is determined by iterating over potential points, which are a subset of extreme points. The supposedly best location is selected using a merit function considering the proportion of the contact surface with the underlying item. The algorithm performs well on large-scale industry instances.

Returning to the problem definition of Paquay *et al.* (2016), which is closest to ours, two heuristic

approaches to this problem have been proposed: a MIP-based constructive heuristic (Paquay *et al.* 2018a) and a tailored two-phase constructive heuristic (Paquay *et al.* 2018a). We will compare our algorithm with the latter in Section 7.5.

Another common approach for three-dimensional packing problems with practical constraints is to model it as a two-level problem where the lower level creates layers and the higher level combines them. Packing horizontal layers is often desirable, especially to assure a high load stability. Approaches are presented in Mack and Bortfeldt (2010); Elhedhli *et al.* (2019); Gzara *et al.* (2020); Calzavara *et al.* (2021).

4. Insertion Heuristic

In this section, we introduce the insertion heuristic to load a single ULD. Its sole objective is to maximize the utilized volume of the given ULD. The weight balance will only be considered in Section 5, when we embed the heuristic into a *Randomized Greedy Search* (RGS). The framework will be extended further to allow for the loading of multiple ULDs in Section 6.

Our algorithm is based on the seminal extreme point insertion heuristic introduced by Crainic *et al.* (2008), who extended the corner point heuristic of Martello *et al.* (2000). The fundamental idea is to load the items one by one according to a predefined sequence, beginning at one of the ULD's corners. Newly loaded items are tried to be placed close to already loaded ones in a way that avoids fragmenting the remaining space. To that end, whenever an item is loaded, a set of so-called extreme points is computed. These are candidate points at which further items can be loaded.

Aside from supporting ULDs with blocked edge space for cargo net locks, one of the main difference of our approach to the one of Crainic *et al.* (2008) (and Paquay *et al.* (2018b)) is an extension of the extreme points concept by generating more points (Section 4.5) and allowing some of them to be moved (Section 4.3). Additionally, we do not evaluate all generated extreme points to decide where the next item is loaded but perform a first fit approach. As a result, our insertion heuristic is very fast so that it can be called multiple times by the RGS.

On a high level, Algorithm 1 outlines the insertion heuristic we propose. The selection of the sorting criterion and whether a substructure is used is dealt with by the RGS. After initializing the set of extreme points, the ULD is adapted to handle the ULD's edge and a possible substructure. Then, the items are grouped and sorted resulting in an ordered list of items and orientations. In this ordered list, each item can appear multiple times with different orientations. Afterwards, the next item to be loaded and a set of orientations are selected iteratively, and the item is loaded at the first extreme point that does not violate any loading restriction. In the following, we will elaborate in greater detail

(line 2) in Section 4.1, how the ULD is adapted to handle the ULD's edge and a possible substructure,
(lines 3–4) in Section 4.2, how items are grouped and sorted, and how the next item and the set of orientations are chosen,
(line 5) in Section 4.3, how the next extreme point is chosen and potentially moved,
(line 7) in Section 4.4, how to check whether an item can be loaded at an extreme point, and
(line 10) in Section 4.5, how the set of extreme points is updated.

4.1. ULD adaption

For ULDs with edge width $e > 0$ and vertical edge offset $\delta > 0$, the available space for loading is not only defined by the outer shape of the ULD, but needs to be further restricted. To ensure that the space above the ULD edge up until the height δ is blocked, we load four non-stackable dummy items of height $\delta - 1$ on the edge of the ULD floor. Here, the non-stackability guarantees that the dummy items cannot be used as support for any other actual item loaded on top. Setting the height of the dummy item to $\delta - 1$ allows items to overlap with the edge at height δ without touching the dummy items (and thus violating the stackability constraint).

Using a substructure can also be modelled by dummy items. Next to the four dummy items blocking the space above the edge, we additionally add one stackable dummy item of height δ covering the whole ULD floor except the edges.

Algorithm 1: Insertion heuristic

Input: Set of items I and a ULD $c \in C$,
sorting criterion $\mathcal{C}$,
degree of randomization ρ,
whether a substructure is used b

Output: Loaded ULD

```
1  Define set of extreme points E = {(0,0,0)}
2  Adapt ULD (dependent on b)
3  Determine ordered list L of items and orientations by means of sorting criterion C
4  for (i, O) ∈ L do
5      while e = getNextExtremePoint(E) exists and i is not loaded do
6          for orientation σ ∈ O do
7              if i can be loaded at the (potentially moved) extreme point e with orientation σ then
8                  Load i at the (potentially moved) extreme point e with orientation σ
9                  Remove i from L
10                 Update set of extreme points E
11                 break
12             end
13         end
14     end
15 end
16 return loaded ULD
```

4.2. Sorting and selecting the next item

Crainic *et al.* (2008) propose different two-level sorting criteria that are based on the volume, height, and area of the items. We adopt some of their ideas and extend them by introducing different kinds of item groups to ensure that similar or even identical items are loaded in close proximity for increased packing density. Here, a particular focus lies on ensuring that items with the same height are loaded next to each other to create packing layers that allow for maximum support of further items loaded on top. Additionally, we consider an item's stackability when selecting the next item to avoid loading non-stackable items close to the ULD floor and thus blocking the loadable space above.

In line 3 of Algorithm 1, we create a list of items and orientations defining the order in which items will be loaded. This is done by creating two sets of item groups, ordering them according to a provided sorting criterion, and then determining the corresponding item orientations.

4.2.1. Item groups

We begin by clustering all items into groups of *identical items*. Two items i, k are considered identical if (i) they have the same weight $i = k$, (ii) they have the same loading characteristics (rotatability, tiltability, stackability) $r_i = r_k$, $t_i = t_k$, $\vartheta_i = \vartheta_k$, and (iii) they have the same set of dimensions $\{s_1^i, s_2^i, s_3^i\} = \{s_1^k, s_2^k, s_3^k\}$.

Then we compute groups of *similar items* based on the possible item heights and the item stackabilities. Two items i, k are considered similar if (i) they can be tilted in such a way that they have the same height and (ii) they have the same stackability characteristic $\vartheta_i = \vartheta_k$

We create a group of similar items $S_{h,\vartheta}$ for each possible item height and stackability indication $\vartheta \in \{0, 1\}$:

$$S_{h,\vartheta} = \left\{ i \in I : \vartheta_i = \vartheta \wedge \left[s_3^i = h \vee (\text{item } i \text{ is tiltable } \wedge (s_1^i = h \vee s_2^i = h)) \right] \right\}.$$

By definition, all members of a given identical item group belong to the same similar item groups. Hence, we will treat each $S_{h,\vartheta}$ as a set of identical item groups instead of a set of items, *i.e.*, $S_{h,\vartheta} \subseteq \mathcal{P}(I)$ instead

of $S_{h,\vartheta} \subseteq I$, where $\mathcal{P}(I)$ denotes the power set of I. Note that depending on the dimensions and tiltability, an identical item group can belong to up to three different sets $S_{h,\vartheta}$.

4.2.2. Sorting

We select one of the following five sorting criteria and sort (*i*) the set of all similar item groups and (*ii*) within each similar item group $S_{h,\vartheta}$ the set of identical item groups accordingly in non-ascending order:

- `cumulated volume`: The cumulative volume of all items in a group.
- `highest volume`: The volume of the group member with the highest volume.
- `stackability-cumulated volume`: Items are sorted lexicographically based on `stackability` first (stackable items are preferred over non-stackable ones) and on `cumulated volume` second.
- `stackability-highest volume`: Analogous to `stackability-cumulated volume` but using `highest volume` as second order criterion.
- `random`.

Additionally, we introduce the parameter $\rho \in (0, 1]$, which allows to define a degree of randomization applied to the item sorting. Based on that we rearrange both sorted sets of item groups. Given a sorted list of item groups $\mathcal{M} = [M_1, M_2, \ldots, M_m]$, we create a new sorting $\bar{\mathcal{M}} = [\bar{M}_1, \bar{M}_2, \ldots, \bar{M}_m]$. Item groups are drawn one-by-one from the original sorting $\mathcal{M}$, where the j-th item group $\bar{M}_j$ corresponds to the item group at position $\left\lceil y_j^{\frac{1}{\rho}} (m - j + 1) \right\rceil$ in $\mathcal{M} \setminus [\bar{M}_1, \bar{M}_2, \ldots, \bar{M}_{j-1}]$, where $y_j \in [0, 1]$ is a uniformly distributed random number. This is an adaption of the random distribution proposed by Shaw (1997), which is often used in metaheuristics. The higher the parameter ρ, the more randomized the sorting is. For $\rho \to 0$, the randomness is close to zero, while the maximum value of $\rho = 1$ corresponds to complete randomness.

For the sorting policies `stackability-cumulated volume` and `stackability-highest volume`, we maintain the strict prioritization of stackable item groups over non-stackable ones, *i.e.*, the randomization is applied separately to the sorted (sub-)sets of stackable and non-stackable item groups.

If we now unpack the sorted item groups, we end up with an ordered list of the individual items defining the loading order:

$$\underbrace{\overbrace{i_{\ell_1}, i_{\ell_1+1}, \ldots,}^{\text{IdenticalItems 1}} \overbrace{i_{\ell_2}, i_{\ell_2+1}, \ldots, \ldots,}^{\text{IdenticalItems 2}}}_{\text{SimilarItems 1}} \underbrace{\overbrace{i_{\ell_j}, i_{\ell_j+1}, \ldots, \ldots,}^{\text{IdenticalItems } j}}_{\text{SimilarItems 2}} \ldots$$

In this ordered list, each item can appear up to three times, depending on the dimensions and tiltability.

4.2.3. Determining item orientation

As the idea is to load items with the same height next to each other, we need to ensure that we not only consecutively process all items contained in the same similar item group, but also that the items's orientations are aligned with the heights defined by the respective similar item groups. That means, for an item $i \in S_{h,\vartheta}$, we need to determine the possible orientations O such that $s_3^i = h$. To do so, it suffices to consider one tilt: If the item is not tiltable, there is nothing to show. If the item is tiltable, we consider the following three cases.

(i) The item size is the same in all dimensions. All tilts yield the same item size, we can just pick one.

(ii) The item size is different in all dimensions. There can only be one tilt where the item height matches the similar item group's height h.

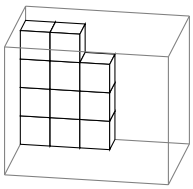
(iii) Two dimensions of the item size are the same.
   - The one unique dimension matches the height h. There is only one applicable tilt.
   - The two identical dimensions match the height h. We can just pick one of the two tilts where the item height matches h. Since we assume each tiltable item to be rotatable (see Section 2), for either tilt we can achieve the same item length and width.

Now, the sorted item list from Section 4.2.2 combined with the corresponding relevant item orientations, constitutes list $\mathcal{L}$ from line 3 of Algorithm 1 with

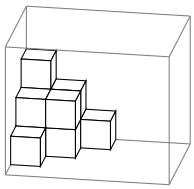
$$\mathcal{L} = \Bigg(\overbrace{\underbrace{(i_{\ell_1}, O_{\ell_1}), (i_{\ell_1+1}, O_{\ell_1+1}), \ldots}^{\text{IdenticalItems 1}}, \underbrace{(i_{\ell_2}, O_{\ell_2}), (i_{\ell_2+1}, O_{\ell_2+1}), \ldots}^{\text{IdenticalItems 2}}, \ldots}^{\text{SimilarItems 1}}, \overbrace{\underbrace{(i_{\ell_j}, O_{\ell_j}), (i_{\ell_j+1}, O_{\ell_j+1}), \ldots}^{\text{IdenticalItems } j}, \ldots, \ldots}^{\text{SimilarItems 2}}, \ldots \Bigg) .$$

4.3. Selecting and potentially moving the next extreme point

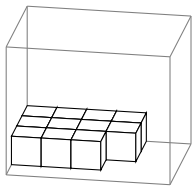
Figure 4 presents three different loading strategies we tested that affect the selection of the next extreme point. To ensure stable loading of a ULD in accordance with the item selection process from Section 4.2, the loading from bottom to top is preferred over the other ones. The set of extreme points is therefore sorted in ascending lexicographic order by z, y, x and the next extreme point is chosen by iterating over the sorted list.



(a) Load in a manner of stacks by sorting the extreme points in ascending lexicographic order by x, y, z.



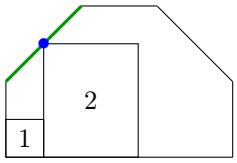
(b) Load around the origin by selecting the point closest to the origin as next extreme point.



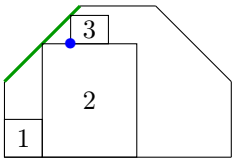
(c) Load the ULD from bottom to top by sorting the extreme points in ascending lexicographic order by z, y, x.

Figure 4: Different loading strategies.

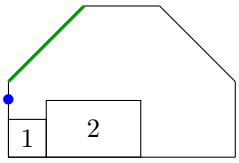
In case the ULD has tilted facets at the top (see Figure 1d), we may generate extreme points at which no item can ever be loaded (for a detailed description of how extreme points are generated, we refer to Section 4.5). An example is shown in Figure 5a. After loading item 2, we generate the extreme point •. As each item has a non-zero height, clearly no item can ever be loaded at this position. Similarly, when item 2 is loaded in the way depicted in Figure 5c, only very few items could ever be loaded at the extreme point •. We call such an extreme point *critical extreme point* and the corresponding tilted facet limiting the loading space for the extreme point *critical facet*.



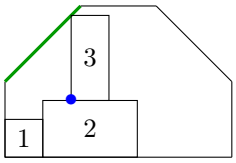
(a) Critical extreme point • generated at a tilted facet.



(b) Moved extreme point • when loading item 3.



(c) (Potentially) critical extreme point • generated at a non-tilted facet.

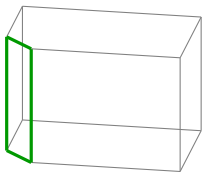


(d) Moved extreme point • when loading item 3.

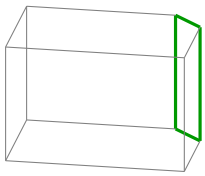
Figure 5: Extreme points at which (potentially) no item can be loaded (5a and 5c) and the corresponding moved extreme points (5b and 5d). The critical facet is colored green/marked in **bold**.

To avoid having these useless extreme points and unused space in the ULD, we introduce a procedure to move them away from the ULD wall according to the size of the item that is supposed to be loaded at the given point.

An extreme point is only allowed to be moved if it is directly on a ULD facet (tilted or non-tilted). Moreover, we only allow moving away from the origin as the way we generate extreme points (see Section 4.5) already ensures sufficient alternative extreme points towards the origin. Considering the relevant ULD shapes



(a) Critical facet with $n_1 < 0, n_2 > 0, n_3 = 0$ and move direction y.



(b) Critical facet with $n_1 > 0, n_2 < 0, n_3 = 0$ and move direction x.

Figure 6: Other shapes where the extreme points can be moved. The critical facet is colored green/marked in **bold**.

(Figure 1), moving in x-direction would only move an extreme point parallel to any potential critical facet and can be neglected. Finally, moving an extreme point in z-direction (relevant for ULD shapes (e) and (f) in Figure 1) very likely results in loading positions at which the item would float. Hence, the only way in which we allow an extreme point to be moved is away from the origin in y-direction. Again, given the relevant ULD shapes, this implies that an extreme point only needs to be considered for moving if it lies on one of the two facets with critical extreme points depicted in Figures 5a and 5c. More formally, if it lies on a facet with plane equation $n_1x + n_2y + n_3z = a$, where $n_1 = 0, n_2 > 0, n_3 \leq 0$.

If we have such a critical extreme point $e = (e_1, e_2, e_3)$, the corresponding critical facet is always given by $n_1x + n_2y + n_3z = a$ with $n_1 = 0, n_2 > 0, n_3 < 0$. To compute the minimum value ν by which we have to move an item of size s so that it fits at extreme point e, we need to place the top left corner of the item directly at the critical facet (see Figure 5b and 5d). Thus, we solve $n_1(e_1 + s_1) + n_2(e_2 + \nu) + n_3(e_3 + s_3) = a$ and obtain

$$\nu = \frac{a - n_2e_2 - n_3(e_3 + s_3)}{n_2}.$$

Now, if $\nu > 0$, we know that we have to move the extreme point in order for the item to fit into the ULD (regarding the critical facet). The new moved extreme point is $(e_1, e_2 + \nu, e_3)$.

Note that the approach can be extended to other shapes, not relevant for the air freight context, as shown in Figure 6.

4.4. Checking whether an item can be loaded at an extreme point

An item can be loaded at an extreme point if it does not exceed the ULD walls, it does not collide with other items, it is supported (non-floating), and it is not stacked on a non-stackable item. In the following, we will explain how the individual checks are performed. The section closes with a description of an acceleration technique.

4.4.1. ULD wall exceedance

Each ULD facet $n_1x + n_2y + n_3z = a$ defines a half-space corresponding to the inside of the ULD via $\{(x, y, z) \in \mathbb{R}^3 : n_1x + n_2y + n_3z \geq a\}$. To check whether an item loaded at a given extreme point lies within the ULD, we need to check for each ULD facet if all of the item's corner points lie within that half-space. We say that the item lies on the inside of that facet. We can apply knowledge about the shape of the ULD/facet and the way that we generate extreme points to simplify and speed up these checks.

Let's consider an item of size $s = (s_1, s_2, s_3)$ that is supposed to be inserted at position $e = (e_1, e_2, e_3)$. First, we assume that we always generate valid extreme points, *i.e.*, points within the ULD. Hence, we never need to check whether the item's corner point corresponding to e lies within the ULD. Secondly, we can distinguish between tilted and non-tilted facets. For the latter case, we can perform the checks for all facets of this kind at once: We simply compute the ULD's bounding box, *i.e.*, the smallest cuboid that covers the ULD. We assume that, by definition, the ULD is shifted towards the origin in such a way that the minimum x-, y-, and z-coordinate over all ULD vertices are 0. Hence, we can store the bounding box as its size $B = (B_1, B_2, B_3)$ along the x-, y-, and z-axis. Now it is sufficient to check whether $e_i + s_i \leq B_i$ for $i \in \{1, 2, 3\}$ to ensure that the item lies on the inner side of all non-tilted facets.

For a tilted facet defined by the plane equation $n_1x + n_2y + n_3z = a$, an item lies on the inside of or on the facet if $n \cdot (e_1 + \delta_1 s_1, e_2 + \delta_2 s_2, e_3 + \delta_3 s_3) \geq a$, where $\delta_d = 1$ if $n_d < 0$ and $\delta_d = 0$ otherwise, for $d \in D$.

4.4.2. Collision check

Two items i and k of size s^i and s^k loaded at positions e^i and e^k collide if $e^k_d < e^i_d + s^i_d$ and $e^i_d < e^k_d + s^k_d$ for all $d \in D$. For the collision check, we just iterate over already loaded items. If any loaded item k collides with the item i to be loaded, item i cannot be loaded at the position in the given orientation.

4.4.3. Non-floating and stackability check

The checks whether an item i is floating or placed on a non-stackable item when loaded at position e are performed in combination in Algorithm 2.

The set of loaded items is extended by an artificial item to model the ULD's base area. In line 2, we first compute a reduced set of loaded items R that are relevant for supporting i: These items have to overlap with i in the x- and y-direction, *i.e.*, their base areas overlap. Moreover, their surface area has to be at an height $\mathcal{H}$ suitable for direct support or indirect support via padding material, *i.e.*, $\mathcal{H} \in [e_3 - \hbar, e_3]$. This includes a dummy item of height 0 modelling the ULD floor if applicable according to the previous condition. Subsequently, we sort this reduced item set by non-ascending z end position. We then iterate over the ordered set to determine whether the item i has any direct support, its total supported area, and the number of supported corner points. If the item i directly rests on a non-stackable item, stackability requirements are violated (see line 10) and we can stop. If it directly rests on a stackable item, the direct support criterion is fulfilled (see line 16). In lines 14–24, the total area and the number of corner points supported by stackable items are updated. The non-floating and stackability check returns 'true', if the item i to load is directly supported and (i) the four bottom corner points are supported or (ii) at least o percent of the item's base area is supported.

In case of a padding height $\hbar > 0$, calculating the total supported area (see lines 18-20) results in the following problem. Since $\hbar > 0$, we can have multiple items in R stacked on top of each other, *i.e.*, we cannot simply compute the supported area of i for each individual loaded item $\ell \in R$ and take the sum. Instead, we would have to calculate the union of the base areas of all stackable items in R, subtract the area potentially blocked by non-stackable items in R on top, and then finally compute the overlap with i. Solving this problem is computationally expensive. Therefore, we apply a simplification: If the current iteration's item ℓ is stackable, we calculate the overlap of its base area with i's base area and add it to the cumulated supported area (see line 18). Subsequently, for each item $\jmath$ considered in the previous iterations, we calculate the overlap of the base areas of ℓ, $\jmath$, and i and subtract it from the cumulated supported area (see line 20). This way, we assure that item $\jmath$'s base area is not counted towards the support for item i if there is another item ℓ between $\jmath$ and i (vertically). However, this procedure only works accurately if there are no more than two potential support items in R stacked on top of each other.

In Figure 7, we show an example where the calculated total supported area is too small. If calculated accurately, the full base area of item 4 should be considered supported (directly or indirectly). To calculate the support, we iterate over the items 3, 2, and 1 (in that order). In the first iteration, we add the base area of item 3. In the second iteration, we add the base area of item 2 and subtract the overlap of the base areas of item 2 and 3, which amounts to item 3's base area in total. Finally, in the third iteration, we add the base area of item 1 and subtract the base areas of items 2 and 3. Hence, we end up with an overall supported area equal to only the base area of item 3.

However, since in realistic instances the padding height $\hbar$ is typically significantly smaller than the item heights, the scenario outlined above only occurs very rarely. Moreover, while our simplification might sometimes rule out otherwise feasible solutions, it never allows for infeasible solutions to be accepted: We only ever overestimate the blocked supported area, *i.e.*, underestimate the actual item support. Hence, we consider the simplification acceptable.

4.4.4. Acceleration techniques

Performing the collision, non-floating, and stackability check for all already loaded items can be very time consuming, especially if the instance consists of many items. However, for these checks, we only have

Algorithm 2: Non-floating and stackability check.

Input: Loaded items L, new item i to load at position e^i
Output: Whether the item i is not floating and meets the stackability requirements

Add artificial item to L to model the ULD's base area
Define $R = \{\ell \in L : \text{item } \ell \text{ intersects with } (e_1^i, e_1^i + s_1^i) \times (e_2^i, e_2^i + s_2^i) \times [e_3^i - \hbar, e_3^i]\} \subseteq L$
Sort loaded items by non-ascending end position $e_3^\ell + s_3^\ell$ to get $R = [\ell_1, \ell_2, \ldots]$
`directlySupported` = false
`totalSupportedArea` = 0
`numberSupportedCornerPoints` = 0
for $\ell \in R$ **do**
  if ℓ *is not stackable* **then**
    if i *directly rests on* ℓ **then**
      return false
    end
    continue
  end
  if i *directly rests on* ℓ **then**
    Update `numberSupportedCornerPoints`
    `directlySupported` = true
  end
  `additionalSupportedArea` = baseAreaOverlap(ℓ, i)
  for $\jmath \in \{\ell_1, \ell_2, \ldots, \ell\}$ **do**
    `additionalSupportedArea` −= baseAreaOverlap$(\ell, \jmath, i)$
  end
  if `additionalSupportedArea` > 0 **then**
    `totalSupportedArea` += `additionalSupportedArea`
  end
end
return `directlySupported` and (`numberSupportedCornerPoints` = 4 or `totalSupportedArea` $\geq o \cdot s_1^i \cdot s_2^i$)

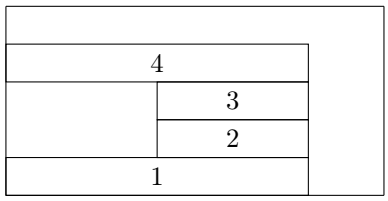


Figure 7: Inaccuracy for stacked support items: Item 4 needs to be loaded, items 1-3 are already loaded. The padding height $\hbar$ is equal to the (vertical) distance between item 4 and item 1.

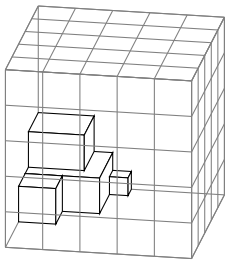


Figure 8: A $5 \times 5 \times 5$ grid for a ULD.

to consider a small subset of the loaded items within close proximity to the position of the next item to be loaded. To identify the relevant items quickly, in the preprocessing we divide the ULD space into a grid of equally sized cubes: First, the average item edge size $\bar{s}$ (considering all 3 dimensions) is calculated as

$$\bar{s} = \frac{1}{3|I|} \sum_{i}^{|I|} (s_1^i + s_2^i + s_3^i).$$

Then, a grid with cells of size $(\bar{s}, \bar{s}, \bar{s})$ is created. Here, each cell is defined as a three-dimensional half-open interval. For example, the grid cell at the origin is defined as $[0, \bar{s}) \times [0, \bar{s}) \times [0, \bar{s})$. Figure 8 shows an example of such a grid.

Whenever an item item i of size s^i is loaded at position e^i, we register it in all grid cells that intersect with $[e_1^i, e_1^i + s_1^i) \times [e_2^i, e_2^i + s_2^i) \times [e_3^i, e_3^i + s_3^i)$. Similarly, if we try to load a new item i at position e^i, we can easily determine the grid cells relevant for our checks. For the collision check, these are again all grid cells which the item intersects with if loaded at the given position. For the non-floating and stackability check, we need to identify all the items below i that can potentially provide support considering the maximum padding height $\hbar$. These are precisely the items in the grid cells that intersect with $[e_1, e_1 + s_1^i) \times [e_2, e_2 + s_2^i) \times [e_3 - \hbar - \varepsilon, e_3)$ with $\varepsilon > 0$.

Determining the intersecting grid cells is computationally inexpensive: Given an $n_1 \times n_2 \times n_3$ grid consisting of the cells

$$[\ell_1 \bar{s} - \bar{s}, \ell_1 \bar{s}) \times [\ell_2 \bar{s} - \bar{s}, \ell_2 \bar{s}) \times [\ell_3 \bar{s} - \bar{s}, \ell_3 \bar{s}) \text{ for } \ell \in \{1, ..., n_1\} \times \{1, ..., n_2\} \times \{1, ..., n_3\}$$

and item i, we can calculate for all $d \in D$

$$\ell_d^{min} = \left\lfloor \frac{e_d^i}{\bar{s}} \right\rfloor \text{ and } \ell_d^{max} = \left\lfloor \frac{e_d^i + s_d^i - \varepsilon}{\bar{s}} \right\rfloor \text{ with } \varepsilon > 0.$$

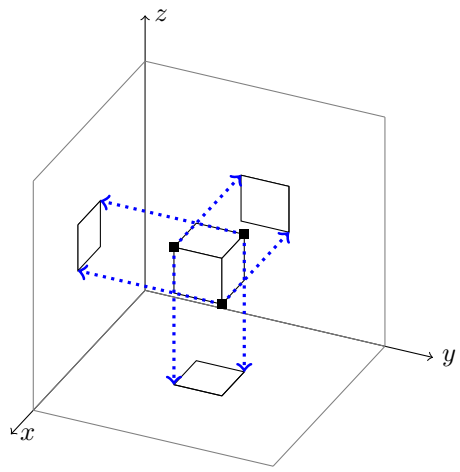
The grid cells intersecting with item i are then given by

$$[\ell_1 \bar{s} - \bar{s}, \ell_1 \bar{s}) \times [\ell_2 \bar{s} - \bar{s}, \ell_2 \bar{s}) \times [\ell_3 \bar{s} - \bar{s}, \ell_3 \bar{s}) \text{ for } \ell \in \{\ell_1^{min}, ..., \ell_1^{max}\} \times \{\ell_2^{min}, ..., \ell_2^{max}\} \times \{\ell_3^{min}, ..., \ell_3^{max}\}.$$

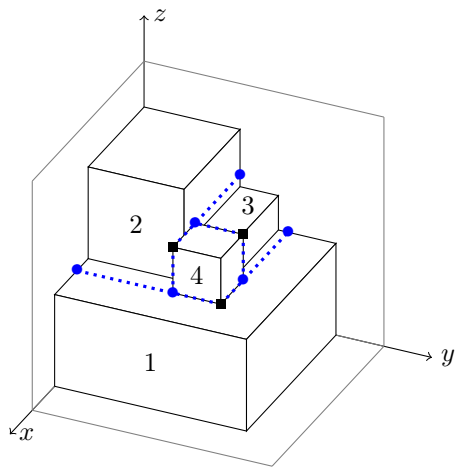
This way, we can significantly reduce the number of items to consider for our checks when loading a new item.

4.5. Generating extreme points

Crainic *et al.* (2008) present an efficient way to determine candidate points where to place the items. The calculation takes the already loaded cargo into account and aims to identify sufficiently many and suitable positions such that a dense packing can be achieved and sufficiently few positions to ensure that the problem can be solved within a reasonable time. This set of extreme points is updated as soon as an item is added to a given packing. In the following, we describe in detail how extreme points are generated. We use the main ideas of Crainic *et al.* (2008) but adapt their approach with the advantage of generating more extreme points without significantly affecting the runtime.



(a) The six different projections for generating extreme points (for illustration purposes, we ignore that this is invalid as the item is floating). They result from projecting the black corner points ■ of the item along the orthogonal axes of the ULD.



(b) Assuming that there are three already loaded items 1, 2, and 3 in the ULD, the same loading position for item 4 results in the blue extreme points •.

Figure 9: Generating extreme points by projection.

Whenever an item is loaded, we are interested in positions which, for at least one dimension, cannot be shifted further towards the ULD origin. Otherwise, the remaining space would be fragmented unnecessarily. To obtain these positions (the extreme points), for each newly inserted item, Crainic *et al.* (2008) project certain item corner points along the orthogonal axis of the ULD. Here, projecting means to 'walk along a given axis' starting from a given point towards the coordinate origin until another item or a ULD wall is hit as shown in Figure 9.

For a given item i of size s loaded at position e, Crainic *et al.* (2008) suggest the following corner points as starting positions for the projections:

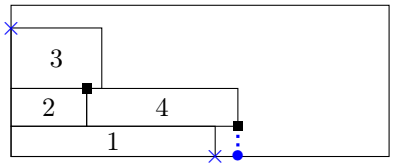
- $(e_1, e_2, e_3 + s_3)$ for projection in x- and y-direction,
- $(e_1, e_2 + s_2, e_3)$ for projection in x- and z-direction, and
- $(e_1 + s_1, e_2, e_3)$ for projection in y- and z-direction.

We slightly modify this approach to start the projection further away from the origin and, hence, potentially include more already loaded items in our projection routine (see Figure 9):

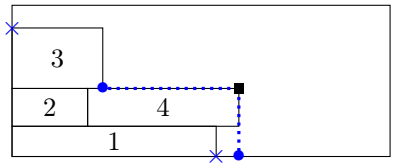
- $(e_1 + s_1, e_2 + s_2, e_3)$ for projection in x- and y-direction,
- $(e_1, e_2 + s_2, e_3 + s_3)$ for projection in x- and z-direction, and
- $(e_1 + s_1, e_2, e_3 + s_3)$ for projection in y- and z-direction.

Figure 10 shows a two-dimensional example for which this modification yields additional viable candidate points.

For an efficient packing, only creating the extreme points that are obtained from *directly* hitting either the ULD wall or another item is often insufficient: Figures 11a and 11b depict the simple, described way of generating extreme points (as two-dimensional example). Now, if we try to load item 7, we can observe that it would fit best at the position illustrated in Figure 11c. However, it cannot be loaded in this spot as no corresponding extreme point has been generated. We resolve this issue by extending Crainic *et al.* (2008) procedure to also generate extreme points for *all* positions at which the projection has not yet directly hit another item, but if a sufficiently large item were to be loaded at this position, it would rest on another already loaded item. This is done by artificially extending the item's surface towards the origin along the non-projection directions before performing the projection as illustrated in Figure 11d.

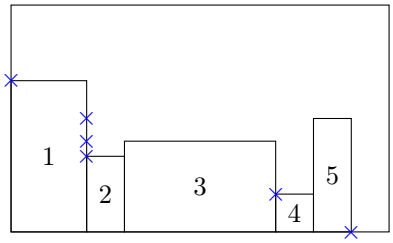


(a) Extreme points according to Crainic *et al.* (2008).

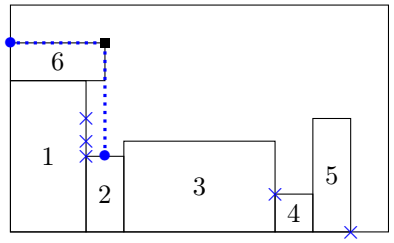


(b) Extreme points according to our procedure.

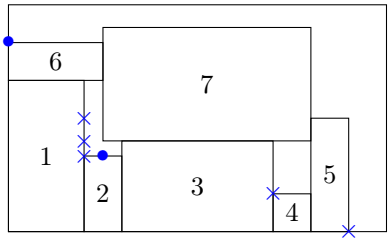
Figure 10: Items 1, 2, and 3 are loaded already. Blue crosses × denote the already existent, not blocked extreme points. New extreme points created by loading item 4 are highlighted by • and the corresponding projection starting point by ■.



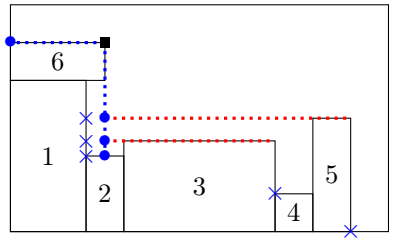
(a) Current status. Extreme points are denoted by ×.



(b) Item 6 is loaded. When only considering directly hit items and the ULD wall, only two new extreme points • are generated when projecting from ■.

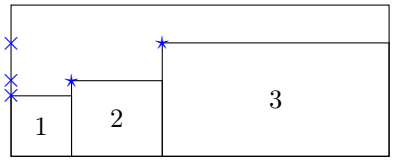


(c) Item 7 would fit well at the depicted position but cannot be loaded since no extreme point has been generated.

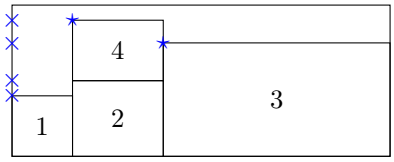


(d) Generate additional extreme points for positions where a newly loaded item might rest on already loaded items 3 or 5.

Figure 11: Example of additional extreme points when loading a new item for a two-dimensional packing problem.



(a) Example with three loaded items. Extreme points are denoted by $\times$ and extreme points on top of items (added in line 15) are denoted by the star $\star$.



(b) Item 4 can only be loaded on top of item 2.

Figure 12: Example of a special case where the extreme point on top of the item is needed to load item 4 (see Algorithm 3 line 15).

We now formalize the procedure to generate new extreme points. Algorithm 3 defines the six projections that must be performed and adds an additional extreme point on top a newly loaded stackable item. Algorithm 4 is the projection subroutine that is called by Algorithm 3 in line 11.

Algorithm 3: Generate new extreme points.

Input: Loaded items L and an item i of size s loaded at position e
Output: Set E of new extreme points

```
1  Define set of new extreme points E = ∅
   /* Add extreme points via projection */
2  Define set of directions D = {1, 2, 3}
3  for j ∈ D do
4  |  if i non-stackable and j = 3 then
5  |  |  continue
6  |  end
7  |  for d ∈ D with j ≠ d do
8  |  |  p = e
9  |  |  p_j = p_j + s_j
10 |  |  p_d = p_d + s_d
11 |  |  E = E ∪ Projection(p, d, L)
12 |  end
13 end
   /* Add extreme point on top of item */
14 if i stackable then
15 |  E = E ∪ {(e_1, e_2, e_3 + s_3)}
16 end
17 return E
```

In lines 8-10 of Algorithm 3, we compute the described starting points for our projections depending on the projection direction $d \in D$. If the newly inserted item is non-stackable, we do not want to generate any extreme points that would require items loaded at these points to directly rest on the non-stackable item at hand. Therefore, lines 4–6 ensure that we do not perform any projection in x- or y-direction starting from a corner point on top of the item. Next to the extreme points of the six projections, the additional extreme point $(e_1, e_2, e_3 + s_3)$ is added in line 15 as it turned out to improve the solution quality. Figure 12 shows a two-dimensional example where this special extreme point is useful.

We explain the projection subroutine (Algorithm 4) with the example depicted in Figure 13. Item 8 has just been loaded and we want to calculate new extreme points by projecting in x-direction. We iterate over all already loaded items in non-ascending order of the items' end positions' x-coordinates (see line 5). In lines 7–9, we then filter out items that cannot be hit by the projection (either directly or by surface

Algorithm 4: Projection.

Input: Projection starting point p, projection direction d, and loaded items L with loading position c^ℓ and size s^ℓ for $\ell \in L$

Output: Set of new extreme points E

Initialize newly generated extreme point $e = p$
Define non-projection directions θ and η such that $\{\theta, \eta, d\} = \{1, 2, 3\}$
Define set of blocking items $\mathcal{B} = \varnothing$
Define set of extreme points $E = \varnothing$
Sort items $\ell \in L$ in non-ascending end position order according to the projection direction, *i.e.*, $c^\ell_d + s^\ell_d$
for $\ell \in L$ **do**
    /* Filter out items that are irrelevant for projection */
    if $c^\ell_d \geq p_d$ *or* $c^\ell_\theta + s^\ell_\theta \leq p_\theta$ *or* $c^\ell_\eta + s^\ell_\eta \leq p_\eta$ **then**
        continue
    end
    /* Filter out items whose projection is blocked by others */
    if $(c^\ell_\theta \geq c^b_\theta$ *or* $p_\theta \geq c^b_\theta)$ *and* $(c^\ell_\eta \geq c^b_\eta$ *or* $p_\eta \geq c^b_\eta)$ *for any* $b \in \mathcal{B}$ **then**
        continue
    end
    /* Add extreme point */
    if $c^\ell_d + s^\ell_d \leq p_d$ *and* $(d \neq 3$ *or* ℓ *is stackable)* **then**
        $e_d = c^\ell_d + s^\ell_d$
        $E = E \cup \{e\}$
    end
    /* Stop if projection directly hits the item (all following items are blocked) */
    if $p_\theta \geq c^\ell_\theta$ *and* $p_\eta \geq c^\ell_\eta$ **then**
        return E
    end
    /* Add item to list of potentially blocking items */
    $\mathcal{B} = \mathcal{B} \cup \{\ell\}$
end
/* Project to ULD wall if projection is not blocked by previous items */
$e_d = 0$
$E = E \cup \{e\}$
return E

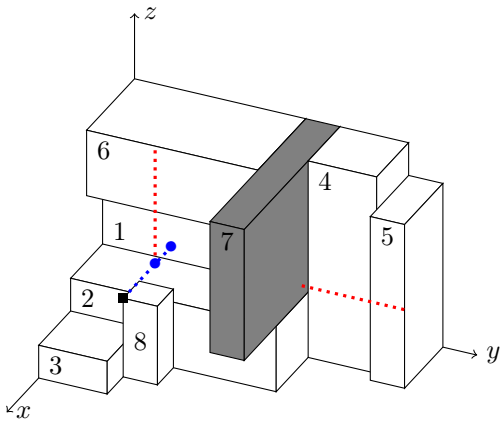


Figure 13: Example of calculating extreme points in x-direction when loading the item 8. The starting point of the projection is denoted by the black square ■. New extreme points are denoted by blue bullets •. Item 7 is a blocking item.

extension). These are all items ℓ that are beyond the projection's starting point p in projection direction ($c_d^\ell \geq p_d$) or are placed on the origin side of p in at least one of the two non-projection directions ($c_\theta^\ell + s_\theta^\ell \leq p_\theta$ or $c_\eta^\ell + s_\eta^\ell \leq p_\eta$). In the example, item 3 is filtered out as it's on the origin side of p in y-direction.

In the next step (lines 10–12), we filter out those items that, in theory, are relevant for the projection but whose surface extension is blocked by other items. In our example, item 5 is blocked by item 7 and the surface extension is highlighted by the red dotted line.

Line 13 ensures that no extreme points resulting from a projection in z-direction that – directly or by extension – hit a non-stackable item are added. If an item is not filtered out by any of the above criteria, the corresponding extreme point is added in line 15. In the example, extreme points are added via projection onto items 6 and 1. We terminate the algorithm as soon as an item is hit directly since all other projections are blocked by this item (line 18). If no item is hit directly, an extreme point at the ULD wall is added (line 23). In the example, the algorithm stops in line 18 when item 1 is hit. Note that the projection to the ULD wall in line 22 is only valid for non-tilted facets. In case the projection hits a tilted facet, the corresponding coordinate can be calculated easily via the corresponding plane equation.

Since Crainic *et al.* (2008) only create extreme points that are obtained from directly hitting either the ULD wall or another item, their approach generates a maximum of six extreme points for each newly loaded item. In our algorithm, (significantly) more extreme points can be generated for each newly loaded item. Nevertheless, our projection routine is fast. With the sorting of L at the beginning and two nested for-loops, one running over the loaded items L and the other one running over a subset of the loaded items $\mathcal{B} \subseteq L$, the overall time complexity is $\mathcal{O}(|L|^2)$. Note that the set of blocking items is typically small ($|\mathcal{B}| \ll |L|$).

5. Randomized Greedy Search

In order to improve the results of the insertion heuristic and to find a loading pattern for a given ULD that simultaneously maximizes the volume of loaded items and considers the weight balance, we apply a *Randomized Greedy Search* (RGS).

The RGS repeatedly calls the insertion heuristic with different item sorting criteria $\mathcal{C}_1, \ldots, \mathcal{C}_5$ (see Section 4.2). The selection of the sorting criteria is evenly distributed over all M iterations, *i.e.*, each sorting criterion is performed $M/5$ times. If the use of a substructure is allowed, all sorting criteria are applied twice – once for the ULD with and once for the ULD without the substructure. After each iteration, the quality of the newly generated solution is evaluated and the best solution is updated accordingly. Finally, after all iterations have been performed, we improve the load stability of the best found solution by trying to close potential gaps between the loaded items. The procedure is summarized in Algorithm 5.

Algorithm 5: Randomized Greedy Search.

Input: Set of items I and a ULD $c \in C$,
number M of RGS iterations,
different sorting criteria,
degree of randomization ρ

Output: Loaded ULD c

```
1  Best found solution S_best = ∅
2  for b ∈ {0, 1} (whether a substructure is used) do
3      for C ∈ {C_1, ..., C_5} (iterate over all sorting criteria) do
4          for j ∈ {1, 2, ..., M/5} do
5              Set degree of randomization δ = ρ
6              if j = 1 then
7                  δ = 0
8              end
9              New solution S = InsertionHeuristic(I, c, C, δ, b)
10             if S is better than S_best then
11                 S_best = S
12             end
13         end
14     end
15 end
16 Refine the solution S_best by avoiding holes
17 return S_best
```

For simplicity, we assume that the use of a substructure is permitted, *i.e.*, $\varsigma = 1$ and $b \in \{0, 1\}$. Otherwise, we can simply omit the for loop in line 2 ($b = 0$). When calling the insertion heuristic in line 9, we ensure that for the first iteration of each sorting policy, we do not apply any randomization. The user-defined degree of randomization will only be considered in the later iterations.

In the following, we explain the solution selection and the avoidance of holes in the load plan.

5.1. Solution selection

The comparison between different solutions is based on a score value that takes the volume utilization, weight balance, and whether the items would also fit into the remaining ULDs into account. Besides the maximum center of gravity deviation $cog^{\max} \in [0, 1]$ that defines in which area the CoG must be (see Figure 3), we introduce the parameter $\mathcal{I}_w \in [0, 1]$ which specifies how important the weight balance is during solution selection. The volume utilization importance is defined by $\mathcal{I}_v = 1 - \mathcal{I}_w \in [0, 1]$.

For dimension $d \in \{1, 2\}$ (only the horizontal weight balance is considered), we define the center of gravity deviation

$$cog_d^{\text{dev}} = \begin{cases} \left| \frac{\text{CoG}_d - B_d/2}{B_d/2} \right| & \text{if } \left| \frac{\text{CoG}_d - B_d/2}{B_d/2} \right| > cog^{\max}, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where B denotes the bounding box of the ULD. Then, the weight balance score is

$$\mathcal{S}_w = 1 - \frac{cog_1^{\text{dev}} + cog_2^{\text{dev}}}{2} \tag{2}$$

and the volume utilization score is

$$\mathcal{S}_v = \frac{\sum_{i \in L} s_1^i s_2^i s_3^i}{\mathcal{V}}. \tag{3}$$

In order to estimate how difficult it will be to load the remaining items in the remaining ULDs, we define the set R of items that could not be loaded yet. Moreover, let U be the set of ULD groups (identical ULDs

constitute one ULD group) and U_i the set of ULD groups into which item $i \in I$ fits. The score of a solution is then defined by

$$\mathcal{S} = \mathcal{I}_w \mathcal{S}_w + \mathcal{I}_v \mathcal{S}_v - \frac{\sum_{i \in R}(|U| - |U_i|)v_i}{\sum_{i \in I}(|U| - |U_i|)v_i}. \tag{4}$$

Solutions with higher scores are preferred.

5.2. Avoiding holes

To prevent items from sliding, we try to remove horizontal holes between the loaded items by moving (sets of) items to the center of the ULD in a post-processing step. For algorithmic performance reasons, the algorithm described below is only performed at the end of the RGS. As a result, the weight balance score (2) of a solution may improve or even worsen, but this is not taken into account when selecting a solution within the RGS.

The idea of the algorithm is to perform the two steps

1. determine a set of items to be moved together
2. move the items as far as possible towards the center of the ULD

repeatedly until no items are moved or a maximum number of iterations is reached.

In step 1, we iterate over all loaded items $\ell \in L$ for which there is a hole in direction x or y. A hole is defined as empty space next to item ℓ in direction of the ULD center with at least one (blocking) item between the item and the ULD wall in this direction. Afterwards, we determine a set Q of neighboring items that should be moved together with ℓ. Note that as illustrated in Figure 14a, the set of neighboring items does not have to directly rest on the floor. Starting with item ℓ, the set of neighboring items is determined by Algorithm 6. In line 4, the bounding box B of Q is iteratively extended and subsequently, items are added to Q that either intersect with B or are supported (directly or indirectly) by B. The algorithm returns an empty set in line 7 to avoid generating several identical sets Q (see Figure 14a).

As soon as a non-empty set Q is found, step 2 is performed. Only considering collisions with other items, it would be straight forward to simply move the item set as far as required to close the hole. However, due to non-floating and stackability restrictions, this would frequently result in infeasible load plans. In these cases, we still aim to move the items as far as possible towards the center of the ULD to reduce the size of the holes and increase load stability. Hence, we perform a binary search on the interval defined by the farthest position to which B can be moved only considering collisions and the current position of the item set bounding box B to determine a feasible new position for the item set that minimizes the hole (see Figure 14b).

Note that we only move items in one direction at a time. In case there are holes for an item in both x- and y-direction, we first perform the above procedure for one direction and then check, after moving the corresponding items, if there are still holes in the other direction.

6. Loading multiple ULDs

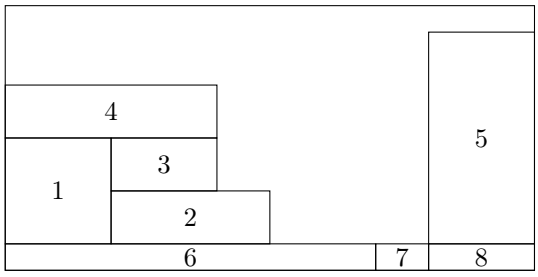
To load items into multiple ULDs, we choose a simple sequential approach (see Algorithm 7). We introduce a ULD selection routine to determine, based on the available items, which ULD should be loaded next/first. Once no more items fit into the currently selected ULD, the remaining items are tried to be loaded into the next ULD. As our ULD selection approach generally prefers larger ULDs over smaller ones, at the end of the algorithm, the items of the last loaded ULD are tried to be reloaded into a smaller available ULD (line 7) to avoid unused space.

We will now explain the selection of the next ULD from line 2. Depending on the instance at hand, certain items are harder to load than others in the sense that they only fit into a limited subset of the available ULDs. As we aim to load all items, this means in turn that some of the ULDs are required to be included in the solution while an efficient packing can render others obsolete. Now, the idea is to first load the ULDs that are required anyway and ensure that we utilize them as good as possible:

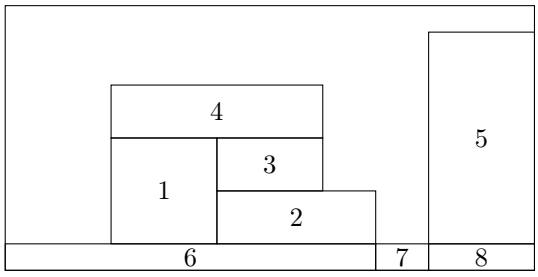
Algorithm 6: Determine set of movable items.

Input: Set of all loaded items L and an item $\ell \in L$ to be moved
Output: A set of movable items $Q \subset L$

```
1  Initialize Q = {ℓ}
2  Define p_3 as the z-position at which item ℓ is loaded
3  repeat
4  |  Determine cuboid bounding box B of Q
5  |  if An item ȷ ∈ L \ Q exists that either intersects with the bounding box B or is supported by B
   |   then
6  |  |  if e_3 < p_3 where e_3 is the z-position at which item ȷ is loaded then
7  |  |  |  return Q = ∅
8  |  |  end
9  |  |  Q = Q ∪ {ȷ}
10 |  end
11 until No item is added to Q
12 return Q
```



(a) The set of items $Q = \{1, 2, 3, 4\}$ is moved together to avoid holes. When determining Q, start items 2 and 3 both would yield the identical set Q without lines 6-8 in Algorithm 6.



(b) New position of the the item set Q determined by the binary search. Item 7 is non-stackable. Only considering collisions, Q could be moved all the way to item 5. As this would violate non-stackability, the items are only moved to the edge of item 7.

Figure 14: Example of moving a set of items.

Algorithm 7: High-level algorithm overview

Input: Set of items to be loaded I and set of available ULDs C
Output: Feasible solution of MBSBPP

```
1 while I ≠ ∅ and C ≠ ∅ do
2 |  Select next ULD c ∈ C
3 |  Load ULD c with available items I
4 |  Update I and C
5 end
6 if C ≠ ∅ then
7 |  Reload items of the last loaded ULD into the smallest available ULD in which all these items fit
8 end
```

Let U be the set of all ULD groups (identical ULDs constitute one ULD group) and $U_i \subseteq U$ the set of ULD groups into which item $i \in I$ fits. Then, the smallest number of fitting ULD groups per item is

$$m = \min_{i \in I} |U_i| \,. \tag{5}$$

We define the set of potential next ULD groups as

$$\mathcal{U} = \bigcup_{i \in I : |U_i| = m} U_i. \tag{6}$$

We select the ULD group $u \in \mathcal{U}$ with the highest cumulative volume of items that can (individually) be loaded in one of the ULDs belonging to the ULD group. Ties are broken by the highest ULD volume.

7. Computational results

In this section, we first give an overview of the benchmark instances and then describe details of the implementation and the computational setup. Next, algorithm components are evaluated. The section closes with a comparison to approaches from the literature and detailed results for an instance set adapted to our presented problem.

7.1. Benchmark instances

We consider three types of benchmark sets: three-dimensional Bin Packing Problem instances proposed by Bischoff and Ratcliff (1995), three-dimensional MBSBPP with transportation constraints instances proposed by Paquay *et al.* (2018b), and an adaption of the Paquay *et al.* (2018b) instance set.

The first instance set by Bischoff and Ratcliff (1995) consists of seven subsets of instances grouped by the number of types of different items $(3, 5, 8, 10, 12, 15, 20)$, each one containing 100 instances. For all instances, one type of cuboid ULD (or bin) is available. To define feasible item orientations, it is specified for each dimension whether the item can be tilted across the corresponding axis. This yields a more detailed item specification compared to ours, which only allows to define on a general level whether an item can be tilted. Item weights and ULD weight capacities are not considered.

The second benchmark set proposed by Paquay *et al.* (2018b) consists of ten subsets of instances grouped by the number of items $(10, 20, \ldots, 100)$, each containing 30 instances. For all instances, an unlimited number of ULDs of six types, which can be loaded in a Boeing 777, is available. The item sets are based on real-world data. However, the authors had to manipulate orientations and stackablity characteristics because these parameters were missing. Note that we only consider the final data set and not the training data set.

The ULD definition of Paquay *et al.* (2018b) differs slightly from ours. In particular, they do not consider the length/width of the edge on which loading at the base area is prohibited. As a consequence, loading a substructure is also not reasonable. Therefore, we additionally test our algorithm on an adapted instance set of the Paquay *et al.* (2018b) instances. The item sets remain unchanged and for each ULD we define the length/width of the edge $e = 10$ and the vertical edge offsest $\delta = 10$. Moreover, we not only consider instances with an unlimited number of available ULDs, but also instances where each ULD type is available only once.

7.2. Details of the implementation and computational setup

The RGS is implemented in `C++` and compiled into a single-thread code. The computational study is carried out on a MacBook Pro with an Apple M2 Pro chip and 32 GB of RAM.

We have conducted extensive preliminary tests to tune our algorithm and determine good parameter settings. Particularly, we had to pay attention to ensuring that the solution quality was good and that the runtime was short at the same time. Pretests were conducted on randomly generated and real-world instances. We were in close contact with our business colleagues to assess the solution quality.

The parameter settings are summarized in Table 2. We distinguish between algorithm-specific and problem-specific parameters. Instead of an overall time limit, the number of extreme point checks (see line 7

in Algorithm 1) is limited, making the stop criterion deterministic. As most of the runtime can be attributed to these checks, it scales proportionally to the number of extreme point checks. If a huge number of items fit into a ULD, the maximum number of extreme point checks can be reached within the first RGS iterations. In this case, we nevertheless execute at least 10 RGS iterations. A maximum number of RGS iterations is additionally set to ensure that small instances are solved in short time. The calculation time for instances with up to 100 items is typically no more than 1 second.

Table 2: Overview of parameter settings.

Type	Parameter	Value
algorithm-specific	maximum number of extreme point checks	20,000,000
	minimum number $M^{\min}$ of RGS iterations	10
	maximum number $M^{\max}$ of RGS iterations	500
	degree of randomization ρ	0.5
	ascending lexicographic extreme point sorting	z, y, x
problem-specific	minimum item overlap o	0.9
	maximum padding height $\hbar$	10
	maximum center of gravity deviation $cog^{\max}$	0.1
	weight balance importance $\mathcal{I}_w$	0.5

7.3. Evaluation of algorithm components

In this section, we analyze individual algorithm components. In particular, we evaluate the proposed grid acceleration, variations of our extreme point generation routine, and different sorting criteria.

The projection in Algorithm 4 differs from that in (Crainic *et al.* 2008) as it not only relies on directly hit items but also items that are hit indirectly via surface extension, resulting in potentially more than one extreme point per projection. We also consider blocking items and different projection starting points. Moreover, we allow to move extreme points (see Section 4.3). To evaluate these changes and our grid acceleration, we tested our algorithm without the respective extensions on adapted Paquay *et al.* (2018b) instances, where each ULD type is just available once (cf. *1 ULD* in Section 7.5). In detail, we have changed the algorithm as follows:

- *No grid acceleration* (`No grid`): We do not use the grid presented in Section 4.4.4 to accelerate the collision, non-floating, and stackability check when loading the next item.
- *No set of blocking items* (`No blocking`): In the projection routine, we filter out items that cannot be hit by the projection since they are blocked by others. We omit this feature by discarding lines 10–12 and 20 of Algorithm 4.
- *No moving of extreme points* (`No moving`): The moving of extreme points as described in Section 4.3 is not performed.
- *Mimic extreme point generation of Crainic* et al. *(2008)* (`Crainic et al`): Only one extreme point per iteration is generated by projecting either on an already loaded item or the container wall. In this variant, lines 10–12 and 20 of Algorithm 4 are discarded. Moreover, the *if*-statements in lines 13–19 are merged by removing lines 16–17 and replacing the *if*-condition in line 13 by $p_\theta \geq c^\ell_\theta$ *and* $p_\eta \geq c^\ell_\eta$ *and* $c^\ell_d + s^\ell_d \leq p_d$ *and* ($d \neq 3$ *or* ℓ *is stackable*). In addition, the projection start point is adapted and no additional extreme point on top of the newly loaded item is generated by discarding lines 10 and 14–16 of Algorithm 3. The moving of extreme points as described in Section 4.3 is also not performed. This approach corresponds to the one of Crainic *et al.* (2008).

Results are summarized in Table 3. For each instance set and algorithm variant, we report the average ratio of the variant runtime to the default runtime t_D, *i.e.*, our algorithm without any changes. Similarly, the ratios of the utilization are denoted. In particular, all values $\frac{t_{NG}}{t_D}$, $\frac{t_{NB}}{t_D}$, $\frac{t_{CR}}{t_D}$, $\frac{t_{NM}}{t_D} > 1$ indicate that the

default approach is faster. Moreover, all values $\frac{u_{NG}}{u_D}$, $\frac{u_{NB}}{u_D}$, $\frac{u_{CR}}{u_D}$, $\frac{u_{NM}}{u_D} < 1$ indicate that the default approach has higher average utilization.

Using a grid accelerates the algorithm by an average of 18.5%. The grid is particularly advantageous for instances with many items. For the variant `No blocking`, the solution time is significantly higher while the improvement in utilization by 0.1% is negligible. This shows that the use of the set of blocking items, in fact, prohibits the generation of irrelevant extreme points and allows for lower runtimes without diminishing solution quality. As expected, the other two variants `No moving` and `Crainic et al` are faster than the default approach since the algorithm has been simplified. However, the utilization is on average 0.7% and 2.2% worse for the variant `No moving` and `Crainic et al`, respectively.

Table 3: Results for different definitions of the set of extreme points and omitted grid acceleration. The solution times t_{NG}, t_{NB}, t_{NM}, t_{CR} and utilizations u_{NG}, u_{NB}, u_{NM}, u_{CR} correspond to the variants `No grid`, `No blocking`, `No moving`, and `Crainic et al`, respectively.

	`No grid`		`No blocking`		`No moving`		`Crainic et al`	
$\|I\|$	$\frac{t_{NG}}{t_D}$	$\frac{u_{NG}}{u_D}$	$\frac{t_{NB}}{t_D}$	$\frac{u_{NB}}{u_D}$	$\frac{t_{NM}}{t_D}$	$\frac{u_{NM}}{u_D}$	$\frac{t_{CR}}{t_D}$	$\frac{u_{CR}}{u_D}$
10	0.947	1.000	0.970	1.000	0.999	1.000	0.920	1.008
20	1.004	1.000	1.003	1.000	0.956	0.995	0.872	0.982
30	1.068	1.000	1.043	1.012	0.940	0.997	0.777	0.956
40	1.095	1.000	1.081	0.998	0.939	0.990	0.788	0.980
50	1.151	1.000	1.109	0.999	0.921	0.983	0.797	0.970
60	1.203	1.000	1.139	0.998	0.898	1.001	0.775	0.956
70	1.257	1.000	1.159	1.003	0.912	0.991	0.771	1.002
80	1.319	1.000	1.196	1.010	0.913	0.985	0.766	0.957
90	1.377	1.000	1.228	0.999	0.919	0.986	0.757	0.986
100	1.430	1.000	1.245	0.993	0.923	1.000	0.756	0.979
Total	1.185	1.000	1.117	1.001	0.932	0.993	0.798	0.978

We also evaluated how often which sorting criterion leads to the best solution for a ULD. Table 4 summarizes results for the adapted Paquay *et al.* (2018b) benchmark groups *unlimited* and *1 ULD* (see Section 7.5). The sorting policy `stackability-cumulated volume` leads to the best solution for most of the ULDs. Randomly drawing the next item is only useful for 37 ULDs in total. However, we point out that in line 10 of Algorithm 5, the best solution is only updated if the new solution is better (and not equal to) the current best solution. Since we apply the different sorting criteria in the order shown in Table 4, a bias towards the earlier called policies is to be expected. It is therefore not surprising that `stackability-cumulative volume` is considered the best for most ULDs. In turn, `random` is better than all other sorting criteria for 37 ULDs. Note that we have also tested other common sorting criteria such as the base area. However, these other sorting criteria did not improve the solution quality.

Table 4: Evaluation of sorting policies.

	unlimited	1 ULD
`stackability-cumulated volume`	347	367
`stackability-highest volume`	99	123
`cumulated volume`	100	99
`highest volume`	93	116
`random`	14	23

7.4. Results for Bischoff and Ratcliff (1995) instances

In order to study the general packing density of our approach, we compare the RGS with two methods of Bischoff and Ratcliff (1995) for the classical three-dimensional bin packing problem. Both methods focus

on maximizing the volume utilization when loading a single bin. The first method additionally takes loading stability into account by loading in a kind of layer structure. The second method addresses multi-drop situations by building stacks. Their approach is optimized to handle weakly heterogeneous items.

To ensure a fair comparison between the different approaches, we adapt the problem-specific parameters to minimum item overlap $o = 1$, padding height $\hbar = 0$, and weight balance importance $\mathcal{I}_w = 0$. However, we do not handle tiltable items in the same way. For items that can only be tilted across one axis, we define that the item is not tiltable at all. Hence, the solution space considered in our approach is smaller. In addition, the set of extreme points is sorted in ascending lexicographic order by x, y, z to mimic the multi-drop situation by predominantly building stacks. Note also that the edge width $e = 0$ and no substructure is allowed ($\varsigma = 0$).

Results grouped by the number $\mathcal{J}$ of different types of items are summarized in Table 5. For a single ULD, we compare the average, minimum, and maximum utilization in percent, denoted by $\bar{u}$, $u^{\min}$, and $u^{\max}$, respectively. The solution time t is reported in milliseconds. The average number of loaded items is denoted by $\overline{|L|}$. The RGS algorithm outperforms the two methods of Bischoff and Ratcliff (1995). For the RGS, the average utilization is 85.0% and decreases when the number $\mathcal{J}$ of item types increases. This is different for the two methods of Bischoff and Ratcliff (1995). The average utilization rate of the RGS is higher in all but one case.

Our results lag behind the tree-search-based approach of Araya *et al.* (2017). Next to the smaller solution space, the main reason for this performance issues is most likely that our approach is optimized for instances with a more heterogeneous set of items and instances with fewer items because 90 or more items per ULD are very rare in practice. Note also that the solution time is below 1 second for all but two instances, which is significantly lower than the 30 seconds of Araya *et al.* (2017). For these two instances, more than 350 items are loaded into one ULD.

Table 5: Comparison of Bischoff and Ratcliff (1995) (BR) and the proposed RGS.

		Method 1 of BR			Method 2 of BR			RGS					
$\mathcal{J}$	$\overline{\lvert I\rvert}$	$\bar{u}$	$u^{\min}$	$u^{\max}$	$\bar{u}$	$u^{\min}$	$u^{\max}$	$\bar{u}$	$u^{\min}$	$u^{\max}$	$\bar{t}$	$t^{\max}$	$\overline{\lvert L\rvert}$
3	150.4	81.8	65.0	94.4	83.8	72.1	93.6	**87.5**	75.3	93.6	218	1505	125.8
5	136.7	81.7	66.7	93.8	84.4	68.0	91.9	**87.3**	78.3	92.2	212	609	112.6
8	134.3	83.0	66.9	92.6	83.9	75.3	89.7	**86.2**	81.2	91.1	257	619	108.9
10	132.9	82.6	66.5	88.9	83.7	73.1	90.1	**85.0**	80.0	90.2	278	695	106.7
12	132.9	82.8	70.4	90.4	83.8	74.9	89.9	**84.2**	78.7	88.4	298	755	106.3
15	131.5	81.5	64.9	89.2	82.4	72.3	88.4	**83.0**	78.4	86.8	323	669	102.9
20	130.3	80.5	70.5	88.3	**82.0**	75.6	86.9	81.5	78.2	84.9	360	608	100.2

7.5. Results for Paquay et al. (2018b) instances

For the comparison with the approach of Paquay *et al.* (2018b), we use the following problem-specific parameters: padding height $\hbar = 0$ and weight balance importance $\mathcal{I}_w = 100$. For these instances, the maximum CoG deviation is between 5% and 10% but differs in x- and y-direction for some containers. We therefore set the maximum center of gravity deviation $cog^{\max} = 0.05$. Moreover, we ignore the minimum item overlap o and only consider items non-floating if the four bottom corner points rest on stackable items or the ULD base area. Note also that the edge width $e = 0$ and no substructure is allowed ($\varsigma = 0$) for all ULDs.

Table 6 displays the comparison between the two approaches grouped by the number of items $|I|$ in an instance. We report the (average) number of ULDs $|C|$ ($\overline{|C|}$) and the median of the utilization u^{med}. For the RGS, we additionally denote the number G^{vio} (percentage $\text{G}^{\text{vio}}(\%)$) of ULDs for which the CoG is violated. Again, the solution time t is reported in milliseconds.

Table 6: Comparison of Paquay *et al.* (2018b) and the RGS.

	Paquay *et al.* (2018b)			RGS							
$\|I\|$	$\|C\|$	$\overline{\|C\|}$	u^{med}	$\|C\|$	$\overline{\|C\|}$	u^{med}	$\bar{u}$	G^{vio}	$\text{G}^{\text{vio}}(\%)$	$\bar{t}$	$t^{\max}$
10	47	1.6	33.2	30	1.0	**34.3**	36.1	0	0.0	26.0	42
20	54	1.8	34.6	35	1.2	**42.1**	40.3	3	8.6	69.0	142
30	72	2.4	36.6	47	1.6	**44.7**	42.7	6	12.8	116.4	281
40	83	2.8	37.9	60	2.0	**46.5**	44.1	4	6.7	88.9	105
50	90	3.0	42.2	66	2.2	**47.5**	45.7	5	7.6	130.5	161
60	108	3.6	41.5	80	2.7	**49.6**	46.4	10	12.5	172.7	206
70	126	4.2	45.7	82	2.7	**54.0**	51.6	6	7.3	228.0	294
80	129	4.3	47.0	93	3.1	**54.5**	51.1	12	12.9	280.6	345
90	141	4.7	47.5	102	3.4	**53.5**	50.3	13	12.7	355.8	422
100	160	5.3	50.2	113	3.8	**53.0**	50.3	20	17.7	429.9	513
Total	1010			708				79			

All in all, the RGS outperforms the two-phase constructive heuristic of Paquay *et al.* (2018b). The median utilization is higher for all instance groups. Moreover, the number of used ULDs $|C|$ is significantly smaller with 708 instead of 1010 in total. The maximum solution time of the RGS is 0.5 seconds. Paquay *et al.* (2018b) report that the solution time is no longer than 12 seconds. The comparison of the solution time should be treated with caution, as different machines were used. For the RGS, the acceptable CoG deviation is violated in 79 of 708 ULDs, which corresponds to 11.2%. The approach of Paquay *et al.* (2018b) is slightly worse in this respect with 12.9% of the loaded ULDs violating the acceptable CoG deviation. To ensure that the values are comparable, we have analyzed the violation in the same way as Paquay *et al.* (2018b), *i.e.*, the maximum CoG deviation is different for some containers in x- and y-direction (which was ignored during solution generation). Note that Paquay *et al.* (2018b) propose two different merit functions MF1 and MF2. Our approach is compared to MF1, while MF2 actually yields better results for the CoG restriction. Instance-by-instance results are shown in the Online Appendix.

7.6. Results for adapted Paquay et al. (2018b) instances

In order to evaluate our algorithm with all its features, we tested the RGS on adapted Paquay *et al.* (2018b) instances by setting the edge width $e = 10$ and the vertical edge offset $\delta = 10$. Table 7 reports results for instances with an unlimited number of available ULDs (denoted by *unlimited*) and instances where each ULD type is available only once (denoted by *1 ULD*). The column #sub. denotes for how many ULDs a substructure is used.

The average utilization is between 37.1% for instances with $|I| = 10$ items and 58.7% for instances with $|I| = 100$ items. The acceptable CoG deviation is violated for 7.2% of the ULDs if each ULD type is available indefinitely. The utilization differs only slightly if each ULD type is available once. However, the violation of the CoG deviation increases to 10.1% of the ULDs. Using a substructure is beneficial for around 27% of the ULDs. All instances are solved within 1.3 seconds. All in all, the results look very promising due to a high utilization of ULDs. Instance-by-instance results are shown in the Online Appendix.

8. Conclusions and outlook

We have introduced an insertion heuristic embedded into a Randomized Greedy Search to solve a three-dimensional MBSBPP with transportation constraints that is relevant in the air freight industry. In particular, the problem considers load stability, (non-)stackable items, weight distribution, specially shaped ULDs, padding material, and the usage of a substructure. The insertion heuristic that is repeatedly called and based on extreme points follows a first fit approach resulting in a very fast algorithm. An underlying

Table 7: Results for adapated Paquay *et al.* (2018b) instances with an unlimited number of ULDs and only one available ULD of each ULD type.

	RGS (unlimited)								RGS (1 ULD)							
|I|	$\|C\|$	$\overline{\|C\|}$	$\bar{u}$	G^{vio}	$G^{vio}(\%)$	#sub.	$\bar{t}$	t^{max}	$\|C\|$	$\overline{\|C\|}$	$\bar{u}$	G^{vio}	$G^{vio}(\%)$	#sub.	$\bar{t}$	t^{max}
10	30	1.0	37.1	3	10.0	2	68.6	107	30	1.0	37.1	4	13.3	2	69.8	110
20	33	1.1	45.9	1	3.0	1	158.4	293	33	1.1	45.9	2	6.1	2	160.6	292
30	45	1.5	44.3	2	4.4	11	274.1	627	45	1.5	44.4	4	8.9	9	279.2	669
40	57	1.9	47.3	5	8.8	17	237.8	860	57	1.9	48.1	2	3.5	14	239.5	854
50	60	2.0	50.7	6	10.0	15	303.8	548	62	2.1	52.3	8	12.9	14	301.6	548
60	72	2.4	51.5	9	12.5	20	392.3	465	77	2.6	54.5	8	10.4	27	383.4	450
70	78	2.6	54.9	4	5.1	22	495.6	686	85	2.8	56.9	10	11.8	24	477.4	580
80	86	2.9	55.1	6	7.0	20	620.0	850	97	3.2	57.5	10	10.3	41	595.1	731
90	92	3.1	56.1	7	7.6	19	763.1	960	109	3.6	58.5	10	9.2	35	746.5	868
100	100	3.3	58.7	4	4.0	29	923.1	1239	134	4.5	56.5	16	11.9	53	888.4	1049
Total	653			47		156			729			74		221		

grid structure is introduced for further acceleration. To ensure a high loading density, the sorting of items with corresponding orientations is based on a grouping procedure that enables both layer and free packing patterns. Moreover, new findings on extreme points were presented by extending the set of extreme points proposed in the literature (Crainic *et al.* 2008). An algorithm for generating extreme points that almost never reaches the quadratic worst-case complexity in the number of loaded items was proposed. In a computational study, we demonstrated that our algorithm is competitive with the state of the art approaches and outperforms them on most instances realistic for the air freight industry. In comparison to the approach of Paquay *et al.* (2018b), the median utilization is higher for all and at least five percent higher for all but two instance groups, while the center of gravity deviation is almost the same. Moreover, results of adapted Paquay *et al.* (2018b) instances are promising for improving the utilization of ULDs in practice. We have also shown that using a substructure is beneficial for around 27% of the ULDs for this benchmark set.

Our work, in particular the grid acceleration, the special item sorting, and the definition, generation, and moving of extreme points, can also be extended to many other packing problems, e.g., two-dimensional packing problems or problem variants with costs (Chan *et al.* 2006). Moreover, the approach can be adapted to assure building stacks, which is relevant in land transportation to easily load and unload (groups of) items. Another research avenue is to focus not only on a high utilization per ULD, but to consider several ULDs simultaneously in order to balance the weight throughout the aircraft. A further practical requirement to tackle is to ensure that predefined groups of items are loaded into the same ULD.

Acknowledgement

The authors would like to thank Michaela Babl and Dennis Schmidt for their detailed answers to all practical questions on the problem and the colleagues from the BinPACKER Air team Ryszard Balewski, Mariusz Ciepluch, Holger Köhler and Paweł Krzemiński for fruitful discussions. Special thanks go to Ivo Hedtke, who provided support with every problem and every question.

References

Alvarez-Valdes, R., Parreño, F., and Tamarit, J. (2013). A grasp/path relinking algorithm for two- and three-dimensional multiple bin-size bin packing problems. *Computers & Operations Research*, **40**(12), 3081–3090.

Araya, I., Guerrero, K., and Nuñez, E. (2017). Vcs: A new heuristic function for selecting boxes in the single container loading problem. *Computers & Operations Research*, **82**, 27–35.

Baldi, M. M., Perboli, G., and Tadei, R. (2012). The three-dimensional knapsack problem with balancing constraints. *Applied Mathematics and Computation*, **218**(19), 9802–9818.

Bischoff, E. and Ratcliff, M. (1995). Issues in the development of approaches to container loading. *Omega*, **23**(4), 377–390.

Bortfeldt, A. and Wäscher, G. (2013). Constraints in container loading – a state-of-the-art review. *European Journal of Operational Research*, **229**(1), 1–20.

Brandt, F. (2017). *The air cargo load planning problem*. Ph.D. thesis, Dissertation, Karlsruhe, Karlsruher Institut für Technologie (KIT), 2017.

Brandt, F. and Nickel, S. (2019). The air cargo load planning problem - a consolidated problem definition and literature review on related problems. *European Journal of Operational Research*, **275**(2), 399–410.

Calzavara, G., Iori, M., Locatelli, M., Moreira, M. C. O., and Silveira, T. (2021). Mathematical models and heuristic algorithms for pallet building problems with practical constraints. *Annals of Operations Research*, pages 1–32.

Ceschia, S. and Schaerf, A. (2013). Local search for a multi-drop multi-container loading problem. *Journal of Heuristics*, **19**(2), 275–294.

Chan, F. T., Bhagwat, R., Kumar, N., Tiwari, M., and Lam, P. (2006). Development of a decision support system for air-cargo pallets loading problem: A case study. *Expert Systems with Applications*, **31**(3), 472–485.

Crainic, T. G., Perboli, G., and Tadei, R. (2008). Extreme point-based heuristics for three-dimensional bin packing. *Informs Journal on Computing*, **20**(3), 368–384.

Crainic, T. G., Perboli, G., and Tadei, R. (2009). Ts2pack: A two-level tabu search for the three-dimensional bin packing problem. *European Journal of Operational Research*, **195**(3), 744–760.

Davies, A. and Bischoff, E. E. (1999). Weight distribution considerations in container loading. *European Journal of Operational Research*, **114**(3), 509–527.

Egeblad, J. and Pisinger, D. (2009). Heuristic approaches for the two- and three-dimensional knapsack packing problem. *Computers & Operations Research*, **36**(4), 1026–1049.

Elhedhli, S., Gzara, F., and Yildiz, B. (2019). Three-dimensional bin packing and mixed-case palletization. *INFORMS Journal on Optimization*, **1**(4), 323–352.

Faroe, O., Pisinger, D., and Zachariasen, M. (2003). Guided local search for the three-dimensional bin-packing problem. *INFORMS Journal on Computing*, **15**(3), 267–283.

Fekete, S. P. and Schepers, J. (2004). A combinatorial characterization of higher-dimensional orthogonal packing. *Mathematics of Operations Research*, **29**(2), 353–368.

Gajda, M., Trivella, A., Mansini, R., and Pisinger, D. (2022). An optimization approach for a complex real-life container loading problem. *Omega*, **107**, 102559.

Garey, M. and Johnson, D. (1979). *Computers and Intractability : A guide to the theory of NP-Completeness*, volume 174. San Francisco: W.H. Freeman.

Gzara, F., Elhedhli, S., and Yildiz, B. C. (2020). The pallet loading problem: Three-dimensional bin packing with practical constraints. *European Journal of Operational Research*, **287**(3), 1062–1074.

Junqueira, L., Morabito, R., and Sato Yamashita, D. (2012). Three-dimensional container loading models with cargo stability and load bearing constraints. *Computers & Operations Research*, **39**(1), 74–85.

Lodi, A., Martello, S., and Vigo, D. (2002). Heuristic algorithms for the three-dimensional bin packing problem. *European Journal of Operational Research*, **141**(2), 410–420.

Lodi, A., Martello, S., and Vigo, D. (2004). Tspack: A unified tabu search code for multi-dimensional bin packing problems. *Annals of Operations Research*, **131**(1–4), 203–213.

Mack, D. and Bortfeldt, A. (2010). A heuristic for solving large bin packing problems in two and three dimensions. *Central European Journal of Operations Research*, **20**(2), 337–354.

Martello, S., Pisinger, D., and Vigo, D. (2000). The three-dimensional bin packing problem. *Operations Research*, **48**(2), 256–267.

Paquay, C., Schyns, M., and Limbourg, S. (2016). A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application. *International Transactions in Operational Research*, **23**(1-2), 187–213.

Paquay, C., Limbourg, S., Schyns, M., and Oliveira, J. F. (2018a). Mip-based constructive heuristics for the three-dimensional bin packing problem with transportation constraints. *International Journal of Production Research*, **56**(4), 1581–1592.

Paquay, C., Limbourg, S., and Schyns, M. (2018b). A tailored two-phase constructive heuristic for the three-dimensional multiple bin size bin packing problem with transportation constraints. *European Journal of Operational Research*, **267**(1), 52–64.

Parreño, F., Alvarez-Valdes, R., Tamarit, J. M., and Oliveira, J. F. (2008). A maximal-space algorithm for the container loading problem. *INFORMS Journal on Computing*, **20**(3), 412–422.

Parreño, F., Alvarez-Valdes, R., Oliveira, J. F., and Tamarit, J. M. (2010). A hybrid grasp/vnd algorithm for two- and three-dimensional bin packing. *Annals of Operations Research*, **179**(1), 203–220.

Pollaris, H., Braekers, K., Caris, A., Janssens, G. K., and Limbourg, S. (2016). Capacitated vehicle routing problem with sequence-based pallet loading and axle weight constraints. *EURO Journal on Transportation and Logistics*, **5**(2), 231–255.

Ratcliff, M. S. W. and Bischoff, E. E. (1998). Allowing for weight considerations in container loading. *OR Spektrum*, **20**(1), 65–71.

Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report, Department of Computer Science, University of Strathclyde, Glasgow, Scotland, UK.

Terno, J., Scheithauer, G., Sommerweiß, U., and Riehme, J. (2000). An efficient approach for the multi-pallet loading problem. *European Journal of Operational Research*, **123**(2), 372–381.

Trivella, A. and Pisinger, D. (2016). The load-balanced multi-dimensional bin-packing problem. *Computers & Operations Research*, **74**, 152–164.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European*

Journal of Operational Research, **183**(3), 1109–1130.

Zhao, X., Bennell, J. A., Bektaş, T., and Dowsland, K. (2014). A comparative review of 3d container loading algorithms. *International Transactions in Operational Research*, **23**(1–2), 287–320.

Online Appendix

In this Appendix, we present instance-by-instance results. Moreover, we present the results for each loaded ULD. The entries in the Tables 8–13 have the following meaning:

$\|I\|$:	number of items;
No.:	instance number;
t:	solution time in milliseconds;
$\|L\|$:	number of loaded items;
$\|C\|$:	number of loaded ULDs;
u^{total}:	percentage of the total utilization (loaded volume / volume of loaded ULDs);
ID:	container id (or type);
$\mathcal{C}$:	sorting criterion of the best solution;
u:	utilization (loaded volume / ULD volume);
cog_1^{dev}:	center of gravity deviation in x-direction in percent;
cog_2^{dev}:	center of gravity deviation in y-direction in percent.

Tables 8–9 display the results for Paquay *et al.* (2018b) instances, Tables 10–11 for adapted Paquay *et al.* (2018b) instances with an unlimited number of available ULDs, and Tables 12–13 for adapted Paquay *et al.* (2018b) instances where each ULD type is available only once.

Table 8: Detailed results for the Paquay *et al.* (2018b) instances.

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
10	0	15	10	1	24.9
10	1	38	10	1	27.9
10	2	30	10	1	41.6
10	3	34	10	1	27.4
10	4	21	10	1	56.1
10	5	8	10	1	13.1
10	6	23	10	1	32.7
10	7	22	10	1	58.5
10	8	31	10	1	38.3
10	9	30	10	1	48.2
10	10	20	10	1	26.4
10	11	42	10	1	26.0
10	12	21	10	1	46.6
10	13	14	10	1	55.0
10	14	22	10	1	62.6
10	15	23	10	1	48.1
10	16	38	10	1	35.9
10	17	34	10	1	25.8
10	18	34	10	1	29.7
10	19	21	10	1	42.5
10	20	24	10	1	25.0
10	21	34	10	1	38.4
10	22	16	10	1	15.8
10	23	35	10	1	27.9
10	24	35	10	1	26.1
10	25	16	10	1	23.3
10	26	38	10	1	16.1
10	27	23	10	1	51.8
10	28	22	10	1	50.5
10	29	16	10	1	41.9
20	0	142	20	1	50.8
20	1	25	20	1	39.4
20	2	48	20	1	42.4
20	3	108	20	1	47.2
20	4	76	20	1	48.1
20	5	25	20	1	33.3
20	6	43	20	1	27.2
20	7	118	20	1	61.5
20	8	69	20	1	43.4
20	9	117	20	1	42.4
20	10	34	20	2	42.4
20	11	39	20	1	47.2
20	12	25	20	1	40.0
20	13	24	20	2	39.1
20	14	75	20	1	35.0
20	15	38	20	1	30.5
20	16	108	20	1	44.5
20	17	110	20	1	55.9
20	18	47	20	1	35.7
20	19	24	20	2	41.6
20	20	27	20	1	46.8
20	21	54	20	1	46.0
20	22	32	20	2	38.3
20	23	120	20	1	24.1
20	24	116	20	1	37.8
20	25	120	20	1	50.0
20	26	24	20	1	30.5
20	27	128	20	1	46.1
20	28	127	20	1	48.5
20	29	28	20	2	37.3
30	0	53	30	2	46.0
30	1	281	30	1	42.0
30	2	247	30	1	61.0
30	3	55	30	2	39.5
30	4	49	30	2	50.9
30	5	54	30	2	50.7
30	6	48	30	2	49.5
30	7	250	30	1	52.7
30	8	249	30	1	50.3
30	9	48	30	2	47.0
30	10	52	30	1	38.3
30	11	246	30	1	34.8
30	12	52	30	2	44.9
30	13	254	30	1	36.1
30	14	49	30	2	48.6
30	15	262	30	1	42.2
30	16	54	30	2	39.7
30	17	51	30	2	42.2
30	18	51	30	2	47.8
30	19	60	30	2	39.8
30	20	47	30	2	49.9
30	21	58	30	2	35.7
30	22	151	30	1	45.5
30	23	52	30	1	45.1
30	24	247	30	1	45.1
30	25	52	30	2	50.1
30	26	56	30	2	39.8
30	27	52	30	2	46.5
30	28	262	30	1	64.6
30	29	50	30	1	43.0
40	0	100	40	2	48.5
40	1	79	40	2	50.3
40	2	95	40	2	46.7
40	3	80	40	2	37.9

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
40	4	86	40	2	43.8
40	5	77	40	2	50.8
40	6	91	40	2	45.3
40	7	95	40	2	35.3
40	8	99	40	2	57.4
40	9	90	40	2	57.0
40	10	81	40	2	41.4
40	11	96	40	2	45.1
40	12	89	40	2	41.5
40	13	86	40	2	47.9
40	14	102	40	2	42.6
40	15	83	40	2	46.0
40	16	92	40	2	38.4
40	17	88	40	2	52.4
40	18	78	40	2	45.0
40	19	104	40	2	41.3
40	20	85	40	2	49.7
40	21	80	40	2	48.2
40	22	97	40	2	53.0
40	23	87	40	2	49.9
40	24	80	40	2	47.9
40	25	105	40	2	43.5
40	26	95	40	2	52.5
40	27	68	40	2	48.2
40	28	99	40	2	55.4
40	29	80	40	2	41.1
50	0	144	50	2	54.3
50	1	130	50	2	53.8
50	2	115	50	3	46.4
50	3	143	50	2	50.0
50	4	118	50	3	43.2
50	5	112	50	3	43.3
50	6	122	50	2	51.5
50	7	118	50	3	46.4
50	8	143	50	2	50.4
50	9	126	50	2	47.4
50	10	148	50	2	47.0
50	11	122	50	2	45.1
50	12	125	50	2	52.4
50	13	136	50	2	48.8
50	14	131	50	2	45.5
50	15	138	50	2	44.2
50	16	138	50	2	50.1
50	17	128	50	2	53.2
50	18	135	50	2	45.1
50	19	161	50	2	54.5
50	20	134	50	2	43.5
50	21	130	50	2	51.9
50	22	124	50	3	40.6
50	23	136	50	2	51.2
50	24	128	50	2	51.0
50	25	122	50	3	44.8
50	26	116	50	2	51.3
50	27	137	50	2	50.8
50	28	133	50	2	50.9
50	29	123	50	2	47.7
60	0	205	60	3	44.1
60	1	177	60	3	44.5
60	2	177	60	3	47.3
60	3	150	60	3	49.7
60	4	206	60	3	48.0
60	5	170	60	3	50.9
60	6	140	60	3	48.8
60	7	184	60	3	44.1
60	8	174	60	2	48.9
60	9	178	60	3	49.8
60	10	161	60	3	45.4
60	11	180	60	2	56.9
60	12	139	60	3	51.2
60	13	193	60	3	49.8
60	14	158	60	2	50.4
60	15	142	60	3	56.4
60	16	141	60	3	55.7
60	17	192	60	2	52.3
60	18	185	60	3	45.8
60	19	190	60	2	48.0
60	20	157	60	3	48.3
60	21	199	60	2	46.6
60	22	190	60	2	49.6
60	23	195	60	2	60.2
60	24	189	60	2	51.3
60	25	146	60	3	53.3
60	26	177	60	2	55.0
60	27	182	60	3	43.0
60	28	147	60	3	52.3
60	29	156	60	3	52.2
70	0	218	70	3	49.0
70	1	260	70	2	63.4
70	2	226	70	3	52.3
70	3	259	70	3	46.2
70	4	211	70	3	54.0
70	5	223	70	3	49.8
70	6	264	70	3	52.2
70	7	294	70	2	61.4

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
70	8	205	70	3	50.6
70	9	210	70	3	54.3
70	10	223	70	3	51.6
70	11	216	70	3	55.7
70	12	193	70	3	51.5
70	13	227	70	3	53.2
70	14	257	70	2	53.0
70	15	210	70	3	48.3
70	16	245	70	3	52.0
70	17	194	70	3	54.1
70	18	234	70	2	55.2
70	19	244	70	3	46.0
70	20	190	70	3	53.2
70	21	272	70	2	56.1
70	22	238	70	3	54.1
70	23	211	70	3	52.3
70	24	215	70	2	54.2
70	25	245	70	2	60.4
70	26	188	70	3	53.7
70	27	232	70	3	51.0
70	28	177	70	3	56.2
70	29	259	70	2	51.8
80	0	268	80	3	55.7
80	1	264	80	3	62.1
80	2	261	80	3	55.0
80	3	300	80	3	55.1
80	4	256	80	3	55.9
80	5	233	80	4	56.1
80	6	300	80	3	51.6
80	7	262	80	4	45.4
80	8	269	80	3	53.8
80	9	308	80	3	52.7
80	10	300	80	3	52.0
80	11	289	80	3	59.5
80	12	345	80	3	43.8
80	13	252	80	3	60.3
80	14	314	80	2	57.5
80	15	260	80	3	56.0
80	16	278	80	3	50.5
80	17	288	80	3	49.5
80	18	240	80	3	59.7
80	19	272	80	3	58.3
80	20	334	80	3	54.2
80	21	267	80	4	50.4
80	22	297	80	3	51.2
80	23	271	80	3	51.7
80	24	327	80	3	52.5
80	25	307	80	3	49.1
80	26	279	80	3	55.2
80	27	254	80	3	51.9
80	28	285	80	3	51.4
80	29	239	80	4	50.0
90	0	397	90	3	49.0
90	1	360	90	3	49.8
90	2	379	90	4	54.7
90	3	405	90	3	52.5
90	4	333	90	4	48.5
90	5	422	90	3	52.3
90	6	385	90	4	52.3
90	7	327	90	3	59.3
90	8	391	90	3	50.1
90	9	285	90	3	60.0
90	10	364	90	4	51.4
90	11	333	90	3	54.3
90	12	328	90	3	57.3
90	13	417	90	3	54.9
90	14	368	90	4	54.8
90	15	342	90	3	56.9
90	16	348	90	3	55.1
90	17	359	90	3	57.6
90	18	329	90	3	54.0
90	19	404	90	3	57.6
90	20	323	90	4	47.7
90	21	288	90	4	51.5
90	22	372	90	3	56.1
90	23	333	90	3	56.2
90	24	326	90	4	49.6
90	25	378	90	3	51.3
90	26	399	90	4	46.4
90	27	341	90	4	51.7
90	28	303	90	4	51.2
90	29	336	90	4	52.6
100	0	484	100	3	53.2
100	1	406	100	4	55.4
100	2	385	100	4	50.8
100	3	407	100	4	52.9
100	4	413	100	4	47.6
100	5	513	100	3	49.5
100	6	362	100	4	58.2
100	7	419	100	3	62.0
100	8	462	100	4	52.9
100	9	394	100	4	50.5
100	10	436	100	4	48.5
100	11	494	100	4	47.2

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
100	12	399	100	4	55.1
100	13	490	100	3	58.4
100	14	448	100	4	52.9
100	15	444	100	4	49.8
100	16	429	100	4	51.7
100	17	413	100	3	59.1
100	18	457	100	3	56.4
100	19	455	100	4	50.5
100	20	431	100	3	56.1
100	21	435	100	4	51.8
100	22	440	100	4	48.4
100	23	394	100	3	65.3
100	24	404	100	4	50.6
100	25	409	100	4	49.3
100	26	414	100	4	52.3
100	27	443	100	4	53.1
100	28	398	100	4	56.2
100	29	420	100	5	43.7

Table 9: Detailed ULD results for Paquay *et al.* (2018b) instances.

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
10	0	5	s-cv	24.9	1.4	0.7
10	1	4	s-cv	27.9	0.7	0.5
10	2	4	s-cv	41.6	3.4	1.2
10	3	4	s-cv	27.4	1.4	3.2
10	4	3	s-cv	56.1	2.6	3.7
10	5	5	s-cv	13.1	1.5	0.0
10	6	3	s-cv	32.7	2.0	1.7
10	7	3	s-cv	58.5	5.2	0.8
10	8	4	s-cv	38.3	5.5	4.5
10	9	4	s-cv	48.2	1.6	0.1
10	10	5	s-cv	26.4	4.3	2.0
10	11	4	s-cv	26.0	1.3	0.3
10	12	3	s-cv	46.6	1.2	0.9
10	13	3	s-cv	55.0	4.6	1.2
10	14	3	hv	62.6	2.3	1.8
10	15	3	s-cv	48.1	1.7	4.0
10	16	4	s-cv	35.9	2.6	2.9
10	17	4	s-cv	25.8	1.0	0.4
10	18	4	s-cv	29.7	2.5	0.2
10	19	3	s-cv	42.5	1.3	0.4
10	20	3	s-cv	25.0	0.3	4.0
10	21	4	s-cv	38.4	5.7	0.2
10	22	5	s-cv	15.8	1.4	0.0
10	23	4	s-cv	27.9	0.1	3.2
10	24	4	s-cv	26.1	0.2	3.7
10	25	4	s-cv	23.3	2.3	0.0
10	26	4	s-cv	16.1	0.0	3.4
10	27	3	cv	51.8	3.8	0.4
10	28	3	s-cv	50.5	2.7	2.9
10	29	3	s-cv	41.9	0.0	0.2
20	0	6	s-hv	50.8	4.7	4.7
20	1	5	s-cv	39.4	3.6	0.8
20	2	4	hv	42.4	4.8	0.1
20	3	4	s-cv	47.2	1.4	1.0
20	4	5	r	48.1	2.4	2.1
20	5	5	s-cv	33.3	2.3	12.5
20	6	5	s-cv	27.2	2.5	0.1
20	7	6	hv	61.5	2.1	3.1
20	8	3	s-cv	43.4	2.9	3.7
20	9	5	s-cv	42.4	0.7	0.1
20	10	5	hv	44.0	10.2	2.9
20	10	4	s-cv	39.7	0.0	1.4
20	11	5	s-cv	47.2	2.6	0.6
20	12	5	s-cv	40.0	1.4	0.4
20	13	5	hv	42.1	2.4	1.7
20	13	1	s-cv	21.2	0.0	10.0
20	14	6	s-cv	35.0	1.7	0.2
20	15	5	s-cv	30.5	0.7	3.9
20	16	4	s-cv	44.5	2.3	3.4
20	17	4	cv	55.9	2.7	3.0
20	18	5	s-cv	35.7	2.4	1.9
20	19	5	hv	48.6	3.2	2.9
20	19	3	s-cv	13.8	0.0	0.0
20	20	5	s-cv	46.8	2.0	4.9
20	21	4	s-cv	46.0	2.3	0.4
20	22	5	hv	37.5	2.4	0.2
20	22	4	s-cv	39.5	0.7	2.1
20	23	5	s-cv	24.1	0.6	0.1
20	24	5	s-cv	37.8	2.0	0.3
20	25	6	s-cv	50.0	3.2	0.6
20	26	5	s-cv	30.5	0.8	0.2
20	27	6	s-cv	46.1	3.1	2.4
20	28	6	s-cv	48.5	1.8	3.3
20	29	5	cv	38.8	4.6	1.9
20	29	3	s-cv	31.4	0.0	0.0
30	0	5	s-cv	50.8	8.2	2.6
30	0	3	s-cv	26.7	3.0	0.0
30	1	5	s-cv	42.0	2.5	1.3
30	2	6	s-hv	61.0	2.2	1.0
30	3	5	s-cv	49.5	2.6	2.3
30	3	4	s-cv	23.6	5.1	22.7
30	4	5	cv	53.2	2.8	1.1
30	4	3	s-cv	41.5	2.1	0.0
30	5	5	cv	51.8	1.4	1.4
30	5	3	s-cv	46.4	1.8	0.0
30	6	5	hv	50.8	1.1	0.7
30	6	3	s-cv	44.2	0.0	0.6
30	7	5	s-hv	52.7	2.6	1.0
30	8	5	s-cv	50.3	3.0	0.5
30	9	5	s-cv	49.6	0.7	0.6
30	9	3	s-cv	36.3	0.1	0.0
30	10	5	s-cv	38.3	1.3	1.6
30	11	5	s-cv	34.8	4.3	0.5
30	12	5	s-cv	49.6	1.6	1.5
30	12	2	s-cv	30.0	0.1	4.6
30	13	5	s-hv	36.1	1.3	5.3
30	14	5	s-cv	50.5	1.3	2.0
30	14	3	s-cv	41.2	0.0	0.0
30	15	5	s-cv	42.2	2.1	3.6
30	16	5	s-hv	44.7	4.2	0.6
30	16	4	s-cv	31.6	0.0	1.9
30	17	5	hv	46.1	1.8	3.9
30	17	3	s-cv	26.6	0.0	0.0
30	18	5	hv	50.6	1.6	4.7

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
30	18	3	s-cv	36.3	0.1	3.9
30	19	5	s-cv	48.5	1.5	3.4
30	19	4	s-cv	26.0	0.6	0.6
30	20	5	hv	52.6	3.5	2.3
30	20	3	s-cv	39.0	0.1	3.8
30	21	5	s-hv	36.4	14.8	2.7
30	21	4	s-cv	34.4	0.0	0.0
30	22	5	hv	45.5	3.3	12.8
30	23	5	s-hv	45.1	4.5	2.3
30	24	5	s-cv	45.1	0.5	3.3
30	25	5	hv	53.0	2.3	0.0
30	25	1	s-cv	32.6	0.0	9.1
30	26	5	s-hv	50.3	4.8	1.5
30	26	4	s-cv	23.2	0.0	0.0
30	27	5	hv	49.5	1.9	3.3
30	27	1	s-cv	28.5	0.9	9.1
30	28	6	s-hv	64.6	2.9	1.6
30	29	5	s-cv	43.0	3.4	2.3
40	0	5	s-cv	50.2	4.0	0.9
40	0	6	s-cv	46.2	0.1	0.5
40	1	5	s-cv	54.2	2.0	2.5
40	1	3	s-cv	34.4	0.0	0.0
40	2	5	s-hv	55.3	1.9	0.0
40	2	4	s-cv	32.9	0.0	0.6
40	3	5	hv	50.9	3.6	1.6
40	3	4	cv	17.3	0.0	0.0
40	4	5	s-hv	47.1	0.9	1.2
40	4	3	s-cv	30.9	0.1	0.0
40	5	5	s-hv	61.6	2.6	4.4
40	5	3	s-cv	7.1	0.0	0.0
40	6	5	s-hv	42.6	2.9	3.9
40	6	6	s-hv	49.1	0.7	4.0
40	7	5	s-hv	46.9	1.2	3.6
40	7	4	s-cv	16.9	4.5	0.0
40	8	5	hv	60.2	1.4	5.1
40	8	6	s-cv	53.6	4.8	2.9
40	9	5	s-hv	61.7	2.6	2.3
40	9	6	s-cv	50.6	0.0	0.2
40	10	5	cv	42.5	2.2	4.1
40	10	3	s-cv	37.1	0.0	13.8
40	11	5	s-cv	52.0	0.8	4.5
40	11	4	s-cv	34.0	1.8	0.0
40	12	5	hv	53.4	0.3	4.1
40	12	4	s-cv	22.5	0.0	4.1
40	13	5	cv	52.8	0.7	1.9
40	13	1	s-cv	18.6	0.0	10.0
40	14	5	cv	44.3	2.7	0.6
40	14	5	s-cv	40.9	1.1	3.3
40	15	5	s-cv	48.5	4.9	2.7
40	15	4	s-cv	41.9	0.0	0.0
40	16	5	hv	42.8	2.7	1.1
40	16	4	s-cv	31.3	2.9	6.1
40	17	5	s-hv	55.5	0.5	2.2
40	17	3	s-cv	40.2	0.1	3.2
40	18	5	cv	55.2	2.2	1.9
40	18	4	s-cv	28.8	1.6	4.4
40	19	5	s-hv	42.7	1.2	1.2
40	19	3	s-cv	35.8	0.1	3.0
40	20	5	cv	58.0	1.2	3.2
40	20	4	s-cv	36.7	0.0	4.6
40	21	5	hv	53.3	3.2	3.7
40	21	4	s-cv	40.1	0.7	0.1
40	22	5	s-cv	52.8	2.1	0.8
40	22	6	s-cv	53.3	6.0	0.6
40	23	5	s-hv	51.2	3.1	4.1
40	23	3	s-cv	44.6	3.3	0.9
40	24	5	s-hv	50.5	1.7	2.5
40	24	4	s-cv	43.7	1.0	0.0
40	25	5	s-hv	45.3	13.0	1.0
40	25	5	s-cv	41.7	0.8	0.0
40	26	5	cv	49.3	6.5	2.9
40	26	4	s-cv	57.6	0.0	0.9
40	27	5	s-cv	52.1	1.8	3.5
40	27	3	s-cv	32.8	0.0	0.0
40	28	5	s-cv	50.9	0.2	3.2
40	28	6	s-cv	61.5	0.0	4.4
40	29	5	hv	49.6	1.9	0.8
40	29	4	s-cv	27.4	0.0	4.8
50	0	5	s-hv	62.3	1.8	2.7
50	0	5	s-cv	46.3	3.8	4.7
50	1	5	cv	55.4	2.1	4.1
50	1	3	s-cv	47.1	0.0	2.5
50	2	5	cv	52.4	5.0	2.8
50	2	5	s-hv	44.7	1.9	1.8
50	2	1	s-cv	21.2	0.0	10.0
50	3	5	s-hv	63.4	2.3	2.6
50	3	4	s-cv	28.8	0.0	1.2
50	4	5	cv	48.6	0.4	1.1
50	4	5	s-cv	50.4	3.5	2.1
50	4	4	s-cv	23.2	0.0	0.0
50	5	5	cv	46.0	12.3	5.1
50	5	5	s-hv	51.4	0.0	4.4
50	5	4	s-cv	26.2	0.7	0.0
50	6	5	s-cv	66.4	2.4	3.7

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
50	6	4	s-cv	28.0	1.8	0.1
50	7	5	s-hv	48.4	4.9	0.1
50	7	5	s-hv	50.1	3.5	4.8
50	7	1	s-cv	13.3	0.0	0.6
50	8	5	cv	44.3	5.2	4.7
50	8	4	s-hv	60.0	4.3	3.7
50	9	5	s-hv	55.8	1.0	0.2
50	9	3	s-cv	13.8	0.0	0.0
50	10	5	s-cv	47.1	0.5	1.6
50	10	5	s-cv	47.0	0.6	4.9
50	11	5	s-hv	47.9	7.2	2.4
50	11	5	s-cv	42.3	0.0	0.8
50	12	5	cv	58.6	4.5	2.3
50	12	3	s-cv	27.7	0.1	0.0
50	13	5	s-hv	48.6	1.6	0.6
50	13	5	hv	48.9	5.2	0.2
50	14	5	s-hv	58.0	4.6	2.8
50	14	4	s-cv	25.5	2.1	1.4
50	15	5	s-cv	39.2	4.6	4.3
50	15	6	hv	51.2	0.0	1.0
50	16	5	hv	50.7	3.1	6.0
50	16	6	s-hv	49.4	1.2	2.4
50	17	5	cv	59.4	1.2	0.5
50	17	5	s-hv	47.1	2.0	0.0
50	18	5	s-hv	52.2	2.4	2.4
50	18	4	s-cv	33.9	3.7	1.3
50	19	5	s-cv	47.4	3.5	0.2
50	19	6	hv	64.2	1.0	5.1
50	20	5	s-cv	45.0	0.6	0.2
50	20	3	s-cv	37.4	0.0	0.0
50	21	5	s-cv	54.6	1.7	1.8
50	21	4	s-cv	47.5	0.9	0.0
50	22	5	cv	41.2	2.9	1.1
50	22	5	hv	52.8	2.4	3.8
50	22	4	s-cv	20.1	1.9	0.0
50	23	5	cv	56.2	0.9	1.9
50	23	4	s-cv	43.4	3.9	4.8
50	24	5	s-hv	63.1	2.9	2.3
50	24	4	cv	31.8	0.0	4.4
50	25	5	r	37.8	2.5	4.4
50	25	5	s-cv	51.8	1.1	3.2
50	25	3	s-cv	44.9	0.1	0.0
50	26	5	s-cv	58.5	3.3	0.5
50	26	4	s-cv	39.8	0.0	3.1
50	27	5	cv	45.9	3.4	5.5
50	27	4	s-cv	58.6	1.1	0.1
50	28	5	hv	58.5	0.7	1.7
50	28	5	s-cv	43.3	0.8	1.6
50	29	5	s-cv	57.2	3.7	1.3
50	29	4	s-cv	32.6	0.0	2.6
60	0	5	hv	38.5	7.5	6.8
60	0	5	hv	54.7	2.6	0.4
60	0	4	s-cv	36.2	2.1	4.1
60	1	5	s-cv	50.7	4.4	1.0
60	1	5	r	51.8	0.4	4.7
60	1	4	s-cv	23.1	29.2	0.0
60	2	5	cv	49.1	1.3	3.2
60	2	5	r	48.4	4.5	0.1
60	2	1	s-cv	30.3	0.0	9.1
60	3	5	hv	50.2	1.8	0.2
60	3	5	s-cv	53.9	3.8	2.7
60	3	3	s-cv	31.0	0.1	0.0
60	4	5	s-cv	46.6	1.8	9.9
60	4	5	s-hv	51.9	1.6	3.3
60	4	3	s-cv	38.3	0.0	0.0
60	5	5	cv	56.6	2.9	0.1
60	5	5	cv	51.7	2.9	3.1
60	5	1	s-cv	12.6	0.0	9.1
60	6	5	cv	42.6	19.9	4.7
60	6	5	r	65.3	3.0	3.3
60	6	6	s-cv	34.6	6.8	5.1
60	7	5	hv	47.4	7.3	0.9
60	7	5	hv	51.9	0.8	3.9
60	7	4	s-cv	26.5	2.6	3.1
60	8	5	s-hv	52.5	1.3	1.4
60	8	4	s-cv	43.2	0.0	0.5
60	9	5	hv	55.7	1.3	5.0
60	9	5	s-cv	49.3	2.0	1.1
60	9	1	s-cv	17.6	0.0	1.8
60	10	5	s-cv	46.2	1.9	0.5
60	10	5	hv	54.1	2.2	4.4
60	10	4	s-cv	30.2	1.4	2.3
60	11	5	s-hv	54.5	1.9	2.4
60	11	4	s-hv	60.7	0.0	1.8
60	12	5	s-cv	58.8	2.0	3.3
60	12	5	hv	48.6	3.0	3.5
60	12	1	s-cv	21.2	0.0	10.0
60	13	5	hv	41.7	2.2	2.2
60	13	5	cv	55.9	1.2	2.1
60	13	4	s-cv	53.2	0.1	0.8
60	14	5	cv	57.9	0.7	1.8
60	14	4	s-hv	38.6	0.8	0.2
60	15	5	s-cv	59.1	5.7	3.2
60	15	5	hv	54.6	1.8	1.2

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
60	15	3	s-cv	53.4	0.0	4.9
60	16	5	s-hv	63.0	1.2	4.1
60	16	5	cv	52.5	3.1	1.3
60	16	2	s-cv	41.9	0.0	4.3
60	17	5	cv	54.8	4.9	1.9
60	17	5	s-hv	49.8	0.3	2.2
60	18	5	cv	47.4	8.2	4.1
60	18	5	s-cv	50.8	2.2	4.4
60	18	4	s-cv	35.1	3.2	4.6
60	19	5	hv	51.3	0.7	4.1
60	19	5	hv	44.7	0.3	3.4
60	20	5	s-hv	48.4	4.6	5.2
60	20	5	cv	52.7	0.8	3.4
60	20	1	s-cv	21.4	0.0	9.1
60	21	5	hv	49.4	2.8	6.2
60	21	5	s-cv	43.8	0.2	3.9
60	22	5	hv	56.5	2.0	0.4
60	22	4	s-cv	38.6	2.2	0.0
60	23	5	hv	64.7	0.6	2.4
60	23	6	s-hv	54.0	0.0	0.4
60	24	5	cv	59.2	1.7	5.3
60	24	4	s-cv	38.8	1.9	2.0
60	25	5	s-hv	59.7	0.5	2.2
60	25	5	hv	52.4	1.0	1.9
60	25	1	s-cv	21.0	4.8	0.6
60	26	5	hv	60.0	1.4	1.8
60	26	6	s-cv	48.1	0.0	2.2
60	27	5	cv	34.2	3.4	0.3
60	27	5	s-hv	55.3	3.1	2.6
60	27	4	s-cv	37.6	0.2	3.4
60	28	5	s-hv	48.6	1.1	2.4
60	28	5	hv	61.3	4.1	4.6
60	28	1	s-cv	20.7	0.0	0.6
60	29	5	s-hv	54.2	1.1	3.7
60	29	5	hv	53.4	2.1	4.2
60	29	3	s-cv	39.1	0.1	4.8
70	0	5	s-cv	45.6	6.2	4.7
70	0	5	cv	58.5	3.2	2.6
70	0	4	s-cv	39.3	2.9	0.0
70	1	5	s-hv	60.6	0.4	1.6
70	1	6	hv	67.3	3.0	2.7
70	2	5	s-hv	56.4	2.2	4.1
70	2	5	hv	53.1	0.5	1.7
70	2	3	s-cv	32.2	0.1	0.0
70	3	5	hv	47.2	1.5	5.2
70	3	5	cv	55.8	1.9	0.0
70	3	4	s-cv	29.4	0.0	0.0
70	4	5	s-cv	60.2	2.4	3.0
70	4	5	cv	52.2	1.9	0.8
70	4	4	s-cv	46.8	2.5	3.8
70	5	5	s-hv	56.7	4.7	0.0
70	5	5	r	54.3	0.1	2.9
70	5	4	s-cv	31.7	3.8	1.7
70	6	5	s-cv	38.0	7.7	2.0
70	6	5	s-hv	56.9	3.8	4.0
70	6	6	s-cv	65.1	3.5	0.3
70	7	5	cv	57.6	0.3	3.7
70	7	6	s-cv	66.7	1.9	2.8
70	8	5	s-hv	55.0	4.1	0.8
70	8	5	cv	54.3	0.1	3.2
70	8	6	s-cv	39.5	0.0	1.2
70	9	5	s-hv	56.9	0.4	2.4
70	9	5	cv	55.8	3.9	1.9
70	9	4	s-cv	47.9	0.2	0.8
70	10	5	s-hv	43.0	2.2	5.6
70	10	5	s-cv	56.7	1.3	0.2
70	10	6	hv	56.5	1.5	4.2
70	11	5	cv	54.8	28.6	0.6
70	11	5	s-hv	57.0	0.8	4.9
70	11	3	s-cv	54.0	0.1	3.5
70	12	5	s-hv	49.2	0.2	5.9
70	12	5	cv	57.1	1.2	0.1
70	12	4	s-cv	46.6	0.0	0.7
70	13	5	s-hv	51.1	0.3	1.9
70	13	5	cv	57.2	4.3	3.1
70	13	6	hv	50.5	2.2	1.5
70	14	5	cv	51.9	2.1	0.7
70	14	5	r	54.1	2.6	1.1
70	15	5	hv	58.1	0.8	1.1
70	15	5	cv	54.0	0.8	2.8
70	15	5	s-cv	32.7	0.6	0.1
70	16	5	cv	56.7	1.4	2.3
70	16	5	hv	52.5	3.3	2.0
70	16	1	s-cv	21.4	0.0	9.1
70	17	5	cv	59.6	0.3	2.8
70	17	5	cv	52.8	1.6	3.3
70	17	4	s-cv	47.5	1.2	2.3
70	18	5	s-cv	59.9	4.2	1.0
70	18	5	cv	50.5	0.0	2.3
70	19	5	s-cv	33.6	8.3	8.6
70	19	5	hv	54.0	4.8	0.3
70	19	4	s-cv	53.0	2.7	0.4
70	20	5	s-cv	66.6	2.1	2.7
70	20	5	hv	53.8	0.7	4.8

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
70	20	4	s-cv	31.1	0.5	0.9
70	21	5	cv	61.4	4.8	1.3
70	21	5	hv	50.8	3.3	3.3
70	22	5	hv	54.1	1.7	3.2
70	22	5	s-cv	60.3	3.9	4.5
70	22	4	s-cv	44.4	2.6	2.3
70	23	5	cv	59.9	3.6	3.0
70	23	5	s-hv	49.6	1.3	3.0
70	23	4	s-cv	44.2	0.0	3.6
70	24	5	cv	60.4	2.8	2.9
70	24	4	s-cv	44.3	0.7	4.0
70	25	5	s-cv	63.5	1.4	3.1
70	25	6	s-cv	56.3	6.3	0.9
70	26	5	s-hv	46.4	6.8	0.6
70	26	5	s-cv	56.3	2.9	1.5
70	26	4	s-cv	61.0	3.1	2.1
70	27	5	s-cv	58.3	4.0	0.5
70	27	5	s-cv	49.8	3.6	3.2
70	27	3	s-cv	27.0	0.1	0.0
70	28	5	hv	59.0	3.3	1.1
70	28	5	s-cv	60.8	2.0	4.5
70	28	6	s-hv	46.1	0.0	2.6
70	29	5	s-hv	53.7	4.2	3.6
70	29	6	s-cv	49.1	0.0	2.2
80	0	5	s-cv	62.7	4.0	2.9
80	0	5	hv	51.9	5.3	2.8
80	0	3	s-cv	43.1	0.1	0.0
80	1	5	s-hv	64.2	1.9	0.3
80	1	5	s-hv	59.6	1.4	4.1
80	1	6	s-cv	62.8	2.0	0.1
80	2	5	cv	62.1	2.5	1.4
80	2	5	s-cv	51.5	7.9	4.3
80	2	3	s-cv	40.3	2.9	0.0
80	3	5	s-hv	57.9	7.3	2.4
80	3	5	hv	52.1	4.9	1.9
80	3	6	r	55.5	4.3	3.6
80	4	5	hv	61.9	0.9	0.3
80	4	5	cv	60.2	2.9	3.4
80	4	4	s-cv	39.7	0.0	1.0
80	5	5	s-hv	60.3	4.0	3.4
80	5	5	s-hv	60.0	1.6	0.7
80	5	5	cv	53.5	4.7	1.7
80	5	1	s-cv	24.3	0.0	9.1
80	6	5	s-hv	65.3	5.0	4.7
80	6	5	r	52.9	2.0	1.3
80	6	5	s-cv	36.6	1.2	0.7
80	7	5	s-hv	48.2	3.1	1.1
80	7	5	cv	34.7	19.0	0.1
80	7	5	s-hv	55.0	2.7	2.7
80	7	4	s-cv	42.4	0.0	1.4
80	8	5	cv	61.6	5.0	3.5
80	8	5	cv	56.1	2.5	3.2
80	8	4	s-cv	37.6	1.8	2.8
80	9	5	s-cv	58.3	0.6	4.0
80	9	5	s-cv	55.4	6.5	4.3
80	9	4	s-cv	39.4	0.0	0.4
80	10	5	cv	59.0	3.2	1.6
80	10	5	s-cv	57.5	2.8	3.2
80	10	4	s-cv	32.0	0.0	4.1
80	11	5	hv	55.7	1.0	0.8
80	11	5	s-cv	63.5	0.3	6.7
80	11	4	s-cv	59.0	2.4	0.6
80	12	5	hv	47.2	0.4	0.4
80	12	5	s-hv	56.4	3.8	1.1
80	12	4	s-cv	18.3	8.2	0.0
80	13	5	hv	67.8	5.0	3.1
80	13	5	s-cv	58.9	3.1	2.5
80	13	1	s-cv	24.3	0.0	15.1
80	14	5	hv	63.0	4.9	2.2
80	14	5	s-cv	51.9	3.5	1.7
80	15	5	hv	64.8	3.6	7.1
80	15	5	s-hv	62.4	0.3	0.6
80	15	4	s-cv	31.9	0.7	0.0
80	16	5	cv	36.7	1.1	9.7
80	16	5	cv	54.2	0.9	1.4
80	16	4	s-hv	66.6	0.0	7.4
80	17	5	s-hv	49.5	4.0	5.4
80	17	5	cv	55.3	4.4	4.0
80	17	4	s-cv	40.4	3.5	1.1
80	18	5	s-cv	59.8	7.4	4.0
80	18	5	s-cv	58.4	0.9	4.8
80	18	3	s-cv	64.6	0.1	2.3
80	19	5	s-cv	69.9	0.1	1.1
80	19	5	hv	60.0	4.5	1.4
80	19	4	s-cv	37.1	0.8	2.0
80	20	5	s-hv	64.6	0.7	1.4
80	20	5	hv	54.3	2.3	2.3
80	20	3	s-cv	12.3	0.0	0.0
80	21	5	cv	46.2	17.0	1.0
80	21	5	s-cv	62.3	1.3	0.4
80	21	5	r	46.3	2.9	0.4
80	21	1	s-cv	29.2	0.0	9.1
80	22	5	cv	45.0	3.6	4.7
80	22	5	s-cv	65.6	3.6	1.6

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
80	22	5	s-cv	42.9	1.7	1.6
80	23	5	s-cv	50.5	2.7	4.3
80	23	5	hv	60.5	0.3	4.0
80	23	5	s-hv	44.3	1.1	0.0
80	24	5	hv	52.2	2.5	3.2
80	24	5	s-hv	54.7	1.1	3.0
80	24	3	s-cv	45.1	0.1	0.0
80	25	5	s-cv	60.8	0.8	4.9
80	25	5	hv	52.6	1.8	5.7
80	25	4	s-cv	25.1	1.3	3.9
80	26	5	s-hv	63.6	3.2	1.1
80	26	5	hv	52.2	1.7	3.5
80	26	3	s-cv	33.4	0.1	2.5
80	27	5	s-hv	54.5	0.4	3.4
80	27	5	cv	56.4	2.4	3.2
80	27	4	s-cv	40.4	0.6	4.7
80	28	5	hv	52.1	2.1	1.8
80	28	5	hv	56.8	0.5	0.4
80	28	5	s-cv	45.4	2.6	0.3
80	29	5	cv	55.0	2.6	4.6
80	29	5	cv	59.5	2.8	0.8
80	29	5	cv	52.4	1.2	0.7
80	29	4	s-cv	23.5	0.0	1.5
90	0	5	cv	57.9	3.7	0.3
90	0	5	r	57.3	5.2	1.3
90	0	5	s-cv	31.8	4.1	0.1
90	1	5	cv	59.6	3.0	1.4
90	1	5	r	53.4	3.9	1.4
90	1	5	s-cv	36.5	4.3	4.1
90	2	5	cv	65.0	3.9	1.0
90	2	5	s-cv	56.7	1.9	4.5
90	2	5	s-cv	47.9	1.3	0.1
90	2	1	s-cv	22.5	0.0	9.1
90	3	5	hv	52.8	3.9	3.0
90	3	5	s-hv	55.8	0.8	1.9
90	3	4	s-hv	46.9	3.1	0.0
90	4	5	cv	41.0	25.2	4.7
90	4	5	s-cv	64.2	4.9	0.1
90	4	5	r	52.3	1.3	3.3
90	4	4	s-cv	29.8	0.0	0.3
90	5	5	s-hv	54.7	1.1	1.7
90	5	5	s-hv	58.8	1.1	2.4
90	5	4	s-cv	38.1	0.0	1.0
90	6	5	hv	56.3	6.5	1.6
90	6	5	s-cv	56.8	1.3	3.4
90	6	5	hv	50.1	4.4	1.2
90	6	1	s-cv	15.2	0.0	10.0
90	7	5	s-hv	61.2	3.5	0.0
90	7	5	s-cv	54.4	4.4	4.5
90	7	4	s-hv	63.7	8.3	3.0
90	8	5	s-hv	60.0	3.3	1.9
90	8	5	s-cv	56.0	3.2	3.7
90	8	4	s-cv	25.1	0.0	2.9
90	9	5	s-cv	56.1	2.4	1.7
90	9	5	s-cv	64.2	6.2	2.2
90	9	4	s-hv	59.6	0.0	1.6
90	10	5	hv	48.8	10.8	5.6
90	10	5	s-cv	58.5	0.1	3.3
90	10	5	s-cv	51.2	0.5	1.9
90	10	4	s-cv	44.6	2.4	0.3
90	11	5	s-cv	62.6	3.5	2.4
90	11	5	s-hv	56.1	4.5	4.3
90	11	4	s-cv	38.3	6.1	1.5
90	12	5	s-cv	70.5	3.4	2.0
90	12	5	s-hv	51.3	2.2	2.8
90	12	3	s-cv	28.5	0.0	0.0
90	13	5	hv	59.4	3.9	0.4
90	13	5	s-hv	57.2	8.0	1.0
90	13	3	s-cv	27.3	0.0	0.0
90	14	5	s-cv	57.5	3.6	3.6
90	14	5	r	56.0	3.3	1.6
90	14	5	r	53.5	1.7	3.7
90	14	1	s-cv	39.6	0.0	15.5
90	15	5	hv	65.8	4.3	3.7
90	15	5	cv	64.4	4.2	1.7
90	15	4	s-cv	31.0	0.0	4.4
90	16	5	hv	64.4	4.5	1.5
90	16	5	s-cv	57.2	0.9	0.2
90	16	4	s-cv	37.1	2.6	0.3
90	17	5	s-hv	61.0	0.8	3.3
90	17	5	s-cv	61.2	2.6	1.2
90	17	5	s-cv	50.7	0.7	4.9
90	18	5	s-cv	55.2	1.9	2.3
90	18	5	s-hv	60.3	0.9	1.0
90	18	5	r	46.6	4.4	4.8
90	19	5	hv	58.4	3.0	0.1
90	19	5	hv	59.4	0.0	4.6
90	19	4	r	53.5	2.1	2.9
90	20	5	hv	40.7	13.7	0.7
90	20	5	cv	60.4	0.9	4.8
90	20	5	hv	47.9	3.8	6.1
90	20	4	s-cv	38.0	1.8	2.1
90	21	5	cv	66.4	2.8	4.0
90	21	5	cv	58.8	2.9	4.8

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
90	21	5	r	50.9	2.7	3.3
90	21	4	s-cv	16.9	0.0	15.9
90	22	5	s-cv	60.2	3.0	3.9
90	22	5	hv	57.3	0.0	0.4
90	22	4	s-cv	47.8	0.5	1.3
90	23	5	s-hv	60.4	0.9	4.2
90	23	5	cv	57.9	1.2	4.6
90	23	4	s-cv	46.8	4.1	0.7
90	24	5	s-cv	52.1	1.6	1.8
90	24	5	hv	49.1	3.3	2.8
90	24	5	s-hv	53.5	1.7	2.7
90	24	3	s-cv	26.2	0.1	0.0
90	25	5	hv	48.3	1.6	2.7
90	25	5	s-cv	58.2	1.1	0.9
90	25	4	cv	45.2	2.1	1.4
90	26	5	cv	46.2	2.8	3.7
90	26	5	s-hv	53.3	1.5	2.3
90	26	5	s-cv	48.2	1.2	2.1
90	26	3	s-cv	12.3	0.0	0.0
90	27	5	s-hv	51.0	3.0	5.7
90	27	5	hv	60.4	2.8	2.5
90	27	5	s-hv	49.4	3.3	3.1
90	27	1	s-cv	18.6	0.0	10.0
90	28	5	s-hv	51.2	1.4	0.8
90	28	5	s-cv	56.1	3.4	1.2
90	28	5	s-cv	50.4	0.9	4.5
90	28	3	s-cv	35.5	0.1	0.0
90	29	5	cv	41.1	5.7	3.1
90	29	5	s-hv	59.0	0.9	0.4
90	29	5	s-cv	59.7	3.2	4.2
90	29	3	s-cv	44.9	0.1	3.3
100	0	5	hv	57.1	4.2	2.2
100	0	5	cv	56.4	0.0	4.0
100	0	3	s-cv	25.0	2.4	0.0
100	1	5	s-hv	63.4	1.2	0.4
100	1	5	s-hv	60.3	2.4	7.1
100	1	5	cv	53.0	6.0	2.4
100	1	3	s-cv	13.8	0.0	0.0
100	2	5	s-cv	58.7	4.5	4.4
100	2	5	hv	58.2	1.3	2.2
100	2	5	r	51.6	2.1	1.4
100	2	4	s-cv	25.3	12.7	0.0
100	3	5	s-hv	42.6	6.4	4.6
100	3	5	s-cv	64.2	3.0	1.5
100	3	5	hv	55.4	4.2	3.2
100	3	4	s-cv	47.1	0.5	3.7
100	4	5	cv	47.4	5.1	1.2
100	4	5	cv	60.7	2.7	2.6
100	4	5	cv	50.9	0.2	0.0
100	4	4	s-cv	22.0	0.0	0.0
100	5	5	hv	54.9	1.2	5.3
100	5	5	s-hv	55.6	1.0	1.0
100	5	5	hv	38.0	2.2	0.1
100	6	5	cv	55.9	1.0	3.1
100	6	5	cv	67.9	2.0	2.0
100	6	5	cv	55.6	10.3	2.9
100	6	4	s-hv	50.6	0.0	3.3
100	7	5	s-hv	66.2	5.5	2.5
100	7	5	hv	65.3	2.9	1.2
100	7	3	s-cv	32.1	0.1	3.6
100	8	5	s-hv	50.8	2.7	2.5
100	8	5	s-hv	60.4	4.4	2.7
100	8	5	s-cv	51.9	1.7	3.7
100	8	1	s-cv	27.0	0.0	9.1
100	9	5	hv	60.4	1.0	4.0
100	9	5	cv	57.9	5.6	0.1
100	9	5	s-hv	50.0	2.5	2.7
100	9	4	s-hv	23.9	4.6	0.0
100	10	5	r	57.5	0.5	3.0
100	10	5	hv	56.4	2.9	3.6
100	10	5	cv	52.2	1.9	0.1
100	10	4	s-cv	15.7	0.0	0.0
100	11	5	cv	54.5	6.1	3.9
100	11	5	hv	46.8	3.7	3.7
100	11	5	r	53.8	2.3	5.0
100	11	4	s-cv	25.7	0.0	0.0
100	12	5	cv	50.9	0.4	12.2
100	12	5	s-cv	64.9	1.3	2.2
100	12	5	hv	60.5	1.8	1.1
100	12	5	cv	43.9	0.6	3.6
100	13	5	s-hv	66.4	8.8	1.7
100	13	5	s-hv	59.9	4.7	3.8
100	13	4	s-cv	43.3	0.1	1.4
100	14	5	hv	61.2	5.0	3.4
100	14	5	s-hv	52.2	3.5	2.9
100	14	5	cv	48.6	1.9	1.5
100	14	3	s-cv	39.3	0.0	0.0
100	15	5	hv	48.5	1.4	1.8
100	15	5	s-cv	58.3	2.8	0.0
100	15	5	s-hv	57.7	1.7	4.3
100	15	4	s-cv	25.6	0.4	0.0
100	16	5	s-cv	49.5	3.0	4.0
100	16	5	cv	64.1	3.6	0.1
100	16	5	cv	50.9	0.6	3.9

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
100	16	4	s-cv	36.9	0.0	4.8
100	17	5	s-cv	69.4	0.6	5.7
100	17	5	s-hv	56.3	4.7	0.0
100	17	6	s-cv	48.9	4.4	1.2
100	18	5	cv	53.7	4.0	1.2
100	18	5	s-hv	63.2	6.4	1.2
100	18	4	s-cv	49.8	0.0	5.5
100	19	5	s-cv	46.2	6.0	4.0
100	19	5	hv	55.7	1.7	1.0
100	19	5	s-hv	51.0	9.6	3.0
100	19	3	s-cv	44.9	0.0	1.5
100	20	5	s-hv	62.4	4.0	0.7
100	20	5	s-cv	57.5	6.4	4.4
100	20	5	cv	48.4	0.1	1.5
100	21	5	cv	60.9	1.3	1.4
100	21	5	cv	39.9	2.7	3.5
100	21	5	r	60.8	1.8	3.4
100	21	6	s-cv	43.3	3.5	3.6
100	22	5	s-hv	36.2	6.4	4.7
100	22	5	s-hv	66.5	1.2	6.0
100	22	5	cv	52.3	4.4	4.2
100	22	5	s-cv	38.6	1.5	0.9
100	23	5	hv	67.6	1.9	2.3
100	23	5	s-hv	59.4	3.5	0.9
100	23	4	s-hv	70.9	0.0	4.1
100	24	5	s-hv	55.2	1.8	4.1
100	24	5	cv	58.1	1.4	0.7
100	24	5	hv	54.0	4.2	4.3
100	24	4	s-cv	26.0	1.6	0.1
100	25	5	cv	36.1	1.1	3.9
100	25	5	cv	56.2	0.6	2.3
100	25	5	s-hv	56.2	3.6	3.4
100	25	5	s-cv	48.8	3.1	3.0
100	26	5	cv	51.9	26.7	4.5
100	26	5	s-hv	63.4	2.5	1.9
100	26	5	hv	51.1	0.5	3.1
100	26	4	s-cv	36.9	3.5	4.5
100	27	5	s-hv	58.2	0.8	1.0
100	27	5	cv	58.3	3.5	1.1
100	27	5	s-hv	51.3	1.7	4.5
100	27	3	s-cv	19.0	0.1	0.0
100	28	5	cv	56.5	4.1	3.7
100	28	5	cv	61.7	4.3	2.7
100	28	5	hv	57.2	4.1	3.8
100	28	3	s-cv	28.6	0.0	0.0
100	29	5	cv	42.2	1.9	10.1
100	29	5	cv	48.7	2.2	0.2
100	29	5	s-cv	54.6	0.3	0.5
100	29	5	s-cv	42.3	5.7	4.4
100	29	4	s-cv	22.8	0.0	3.3

Table 10: Detailed results for adapted Paquay *et al.* (2018b) instances with an unlimited number of available ULDs.

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
10	0	41	10	1	24.9
10	1	98	10	1	27.9
10	2	82	10	1	41.6
10	3	92	10	1	27.4
10	4	57	10	1	56.1
10	5	22	10	1	13.1
10	6	44	10	1	48.5
10	7	60	10	1	58.5
10	8	82	10	1	38.3
10	9	78	10	1	48.2
10	10	55	10	1	26.4
10	11	107	10	1	26.0
10	12	58	10	1	46.6
10	13	40	10	1	55.0
10	14	62	10	1	62.6
10	15	63	10	1	48.1
10	16	97	10	1	35.9
10	17	93	10	1	25.8
10	18	93	10	1	29.7
10	19	58	10	1	42.5
10	20	43	10	1	37.1
10	21	89	10	1	38.4
10	22	42	10	1	15.8
10	23	94	10	1	27.9
10	24	96	10	1	26.1
10	25	43	10	1	23.3
10	26	102	10	1	16.1
10	27	62	10	1	51.8
10	28	61	10	1	50.5
10	29	43	10	1	41.9
20	0	264	20	1	58.9
20	1	60	20	1	39.4
20	2	118	20	1	42.4
20	3	258	20	1	47.2
20	4	180	20	1	48.1
20	5	62	20	1	33.3
20	6	106	20	1	27.2
20	7	284	20	1	61.5
20	8	173	20	1	43.4
20	9	219	20	1	67.3
20	10	74	20	2	55.3
20	11	93	20	1	47.2
20	12	61	20	1	40.0
20	13	293	20	1	62.6
20	14	119	20	1	40.5
20	15	96	20	1	30.5
20	16	260	20	1	44.5
20	17	265	20	1	55.9
20	18	113	20	1	35.7
20	19	59	20	2	41.6
20	20	66	20	1	46.8
20	21	128	20	1	46.0
20	22	67	20	2	49.9
20	23	227	20	1	38.2
20	24	276	20	1	51.8
20	25	232	20	1	57.9
20	26	62	20	1	30.5
20	27	240	20	1	53.4
20	28	235	20	1	56.2
20	29	62	20	1	46.6
30	0	124	30	2	49.2
30	1	627	30	1	57.6
30	2	556	30	1	61.0
30	3	132	30	2	39.5
30	4	113	30	2	50.9
30	5	128	30	2	50.7
30	6	113	30	2	49.5
30	7	571	30	1	52.7
30	8	566	30	1	50.3
30	9	114	30	2	50.2
30	10	122	30	1	38.3
30	11	556	30	1	47.8
30	12	118	30	2	47.1
30	13	577	30	1	49.4
30	14	117	30	2	48.6
30	15	479	30	1	67.0
30	16	119	30	2	51.7
30	17	115	30	1	52.7
30	18	123	30	2	47.8
30	19	143	30	2	39.8
30	20	110	30	2	53.3
30	21	131	30	2	46.5
30	22	346	30	1	45.5
30	23	122	30	1	45.1
30	24	574	30	1	45.1
30	25	122	30	2	46.8
30	26	126	30	2	51.9
30	27	481	30	1	54.3
30	28	580	30	1	64.6
30	29	119	30	1	43.0
40	0	217	40	2	51.5
40	1	183	40	2	50.3

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
40	2	217	40	2	46.7
40	3	184	40	2	49.5
40	4	197	40	2	46.9
40	5	184	40	2	50.8
40	6	199	40	2	48.1
40	7	860	40	1	57.6
40	8	205	40	2	60.9
40	9	212	40	2	60.5
40	10	536	40	1	51.8
40	11	209	40	2	45.1
40	12	199	40	2	54.1
40	13	201	40	2	47.9
40	14	220	40	2	52.3
40	15	186	40	2	46.0
40	16	197	40	2	50.1
40	17	210	40	2	52.4
40	18	176	40	2	58.8
40	19	378	40	1	51.6
40	20	205	40	2	49.7
40	21	191	40	2	48.2
40	22	211	40	2	56.2
40	23	195	40	2	49.9
40	24	195	40	2	47.9
40	25	204	40	2	53.4
40	26	216	40	2	52.5
40	27	160	40	2	51.6
40	28	207	40	2	58.8
40	29	181	40	2	53.6
50	0	330	50	2	62.8
50	1	293	50	2	57.5
50	2	286	50	2	50.3
50	3	318	50	2	50.0
50	4	265	50	3	52.4
50	5	249	50	3	52.5
50	6	268	50	2	64.1
50	7	304	50	2	50.3
50	8	325	50	2	50.4
50	9	280	50	1	59.3
50	10	295	50	2	57.7
50	11	297	50	2	52.1
50	12	280	50	2	56.1
50	13	309	50	2	56.4
50	14	303	50	2	45.5
50	15	287	50	2	47.0
50	16	288	50	2	53.2
50	17	268	50	2	65.3
50	18	310	50	2	45.1
50	19	323	50	2	57.8
50	20	548	50	1	54.3
50	21	289	50	2	51.9
50	22	343	50	2	61.7
50	23	308	50	2	51.2
50	24	305	50	2	51.0
50	25	327	50	2	50.4
50	26	259	50	2	51.3
50	27	320	50	2	50.8
50	28	264	50	2	62.5
50	29	274	50	2	47.7
60	0	421	60	3	53.5
60	1	386	60	3	52.0
60	2	465	60	2	59.3
60	3	412	60	2	55.9
60	4	417	60	3	49.8
60	5	458	60	2	63.8
60	6	310	60	3	50.6
60	7	389	60	3	53.5
60	8	391	60	2	48.9
60	9	457	60	2	62.4
60	10	358	60	3	53.0
60	11	389	60	2	56.9
60	12	370	60	2	64.2
60	13	370	60	3	49.8
60	14	353	60	2	50.4
60	15	328	60	3	58.6
60	16	325	60	3	57.2
60	17	431	60	2	60.4
60	18	388	60	3	53.5
60	19	394	60	2	58.9
60	20	411	60	2	52.3
60	21	427	60	2	57.2
60	22	403	60	2	49.6
60	23	437	60	2	60.2
60	24	424	60	2	51.3
60	25	375	60	2	57.8
60	26	368	60	2	58.4
60	27	384	60	3	50.3
60	28	388	60	2	56.6
60	29	339	60	3	54.1
70	0	455	70	3	57.2
70	1	522	70	2	67.3
70	2	581	70	2	58.8
70	3	560	70	3	54.0

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
70	4	450	70	3	54.0
70	5	473	70	3	58.2
70	6	530	70	3	54.2
70	7	686	70	2	53.1
70	8	442	70	3	52.5
70	9	458	70	3	54.3
70	10	476	70	3	53.6
70	11	544	70	2	62.7
70	12	412	70	3	60.3
70	13	469	70	3	55.2
70	14	565	70	2	61.3
70	15	431	70	3	55.1
70	16	604	70	2	65.2
70	17	452	70	3	54.1
70	18	496	70	2	63.8
70	19	490	70	3	55.8
70	20	403	70	3	62.2
70	21	532	70	2	56.1
70	22	505	70	3	54.1
70	23	463	70	3	61.1
70	24	445	70	2	54.2
70	25	509	70	2	64.2
70	26	423	70	3	53.7
70	27	568	70	2	57.4
70	28	394	70	3	58.3
70	29	531	70	2	54.9
80	0	705	80	2	62.7
80	1	534	80	3	64.5
80	2	572	80	3	57.0
80	3	670	80	3	57.2
80	4	560	80	3	55.9
80	5	565	80	3	65.1
80	6	664	80	3	58.8
80	7	558	80	4	50.7
80	8	559	80	3	62.9
80	9	676	80	3	52.7
80	10	628	80	3	52.0
80	11	605	80	3	59.5
80	12	801	80	2	57.6
80	13	539	80	3	60.3
80	14	663	80	2	66.5
80	15	567	80	3	56.0
80	16	585	80	3	50.5
80	17	601	80	3	57.9
80	18	515	80	3	62.0
80	19	615	80	3	58.3
80	20	850	80	2	61.0
80	21	630	80	3	53.3
80	22	627	80	3	58.4
80	23	546	80	3	59.0
80	24	762	80	2	59.0
80	25	663	80	3	57.4
80	26	699	80	2	62.1
80	27	552	80	3	51.9
80	28	572	80	3	58.7
80	29	517	80	4	57.3
90	0	823	90	3	55.9
90	1	744	90	3	56.9
90	2	836	90	3	65.9
90	3	835	90	3	52.5
90	4	706	90	4	55.6
90	5	828	90	3	61.1
90	6	883	90	3	60.7
90	7	715	90	3	59.3
90	8	820	90	3	60.8
90	9	613	90	3	60.0
90	10	772	90	4	51.4
90	11	724	90	3	54.3
90	12	698	90	3	59.4
90	13	960	90	2	61.7
90	14	816	90	3	57.9
90	15	680	90	3	56.9
90	16	735	90	3	55.1
90	17	736	90	3	63.4
90	18	709	90	3	59.4
90	19	879	90	3	57.6
90	20	732	90	3	65.8
90	21	739	90	3	62.3
90	22	780	90	3	56.1
90	23	706	90	3	56.2
90	24	695	90	3	61.3
90	25	784	90	3	60.0
90	26	854	90	3	57.3
90	27	747	90	3	62.3
90	28	655	90	3	63.3
90	29	688	90	4	54.0
100	0	1239	100	2	59.9
100	1	935	100	3	66.0
100	2	866	100	3	61.5
100	3	843	100	4	59.1
100	4	910	100	3	63.3
100	5	1045	100	3	56.5

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
100	6	743	100	4	65.0
100	7	873	100	3	62.0
100	8	1020	100	3	63.8
100	9	891	100	3	61.1
100	10	1005	100	3	67.0
100	11	1070	100	4	52.7
100	12	830	100	4	60.7
100	13	1014	100	3	58.4
100	14	1022	100	3	57.2
100	15	897	100	4	57.0
100	16	925	100	4	59.3
100	17	823	100	3	61.4
100	18	973	100	3	65.9
100	19	970	100	3	60.1
100	20	903	100	3	64.0
100	21	871	100	4	59.5
100	22	859	100	4	59.6
100	23	834	100	3	65.3
100	24	910	100	3	67.3
100	25	782	100	4	60.7
100	26	884	100	4	52.3
100	27	946	100	3	63.2
100	28	904	100	3	66.9
100	29	907	100	4	55.7

Table 11: Detailed ULD results for adapted Paquay *et al.* (2018b) instances with an unlimited number of available ULDs.

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
10	0	5	s-cv	24.9	3.0	2.6
10	1	4	s-cv	27.9	6.5	6.6
10	2	4	s-cv	41.6	13.9	8.9
10	3	4	s-cv	27.4	1.8	6.8
10	4	3	s-cv	56.1	0.2	2.9
10	5	5	s-cv	13.1	1.8	0.0
10	6	1	cv	48.5	9.3	24.7
10	7	3	s-cv	58.5	1.5	6.6
10	8	4	s-cv	38.3	2.7	5.1
10	9	4	s-cv	48.2	1.9	1.3
10	10	5	s-cv	26.4	0.4	7.2
10	11	4	s-cv	26.0	1.9	5.7
10	12	3	s-cv	46.6	9.2	6.6
10	13	3	s-cv	55.0	4.1	0.6
10	14	3	s-hv	62.6	1.2	2.7
10	15	3	s-cv	48.1	5.6	3.0
10	16	4	s-cv	35.9	1.5	1.5
10	17	4	s-cv	25.8	5.5	1.5
10	18	4	s-cv	29.7	9.9	1.3
10	19	3	s-cv	42.5	5.7	0.0
10	20	1	s-cv	37.1	2.1	20.6
10	21	4	s-cv	38.4	6.6	0.4
10	22	5	s-cv	15.8	0.8	0.0
10	23	4	s-cv	27.9	2.2	0.3
10	24	4	s-cv	26.1	5.2	0.0
10	25	4	s-cv	23.3	2.3	3.2
10	26	4	s-cv	16.1	7.6	8.4
10	27	3	s-cv	51.8	6.3	2.1
10	28	3	s-cv	50.5	6.0	3.2
10	29	3	s-cv	41.9	2.5	2.0
20	0	4	s-cv	58.9	8.6	5.8
20	1	5	s-cv	39.4	3.5	0.1
20	2	4	s-hv	42.4	1.2	7.4
20	3	4	s-cv	47.2	3.8	0.2
20	4	5	s-hv	48.1	3.8	3.7
20	5	5	s-cv	33.3	0.5	9.8
20	6	5	s-cv	27.2	6.4	0.1
20	7	6	s-cv	61.5	6.0	1.1
20	8	3	s-cv	43.4	6.1	2.3
20	9	4	s-cv	67.3	4.7	3.6
20	10	5	hv	56.4	0.3	8.3
20	10	3	s-cv	50.7	0.1	2.2
20	11	5	s-cv	47.2	4.4	3.2
20	12	5	s-cv	40.0	4.1	9.4
20	13	6	s-cv	62.6	0.0	0.9
20	14	4	s-cv	40.5	7.7	1.4
20	15	5	s-cv	30.5	7.9	5.8
20	16	4	s-cv	44.5	4.5	10.0
20	17	4	s-cv	55.9	6.1	5.3
20	18	5	s-cv	35.7	6.5	7.0
20	19	5	hv	48.6	7.8	8.9
20	19	3	s-cv	13.8	0.0	0.0
20	20	5	s-cv	46.8	1.5	2.6
20	21	4	s-cv	46.0	7.8	1.1
20	22	5	hv	52.2	5.6	5.6
20	22	3	s-cv	40.9	0.0	0.0
20	23	4	s-cv	38.2	7.5	0.3
20	24	6	s-cv	51.8	5.2	4.1
20	25	4	s-hv	57.9	3.4	7.9
20	26	5	s-cv	30.5	8.5	7.7
20	27	4	s-cv	53.4	5.3	12.8
20	28	4	s-cv	56.2	9.6	4.2
20	29	5	s-hv	46.6	8.0	9.8
30	0	5	s-cv	54.3	3.0	5.0
30	0	1	s-cv	18.6	0.0	10.8
30	1	6	s-cv	57.6	8.2	2.8
30	2	6	s-cv	61.0	1.7	5.0
30	3	5	s-cv	48.3	4.8	6.6
30	3	4	s-cv	25.6	0.0	7.6
30	4	5	cv	62.5	9.2	9.4
30	4	3	s-cv	4.2	0.0	0.0
30	5	5	s-cv	55.9	4.5	2.4
30	5	3	s-cv	30.0	0.0	0.0
30	6	5	hv	55.8	8.3	8.0
30	6	3	s-cv	24.2	0.1	0.0
30	7	5	cv	52.7	0.8	5.2
30	8	5	s-cv	50.3	3.3	8.9
30	9	5	s-cv	55.0	1.3	2.3
30	9	1	s-cv	21.6	0.0	1.4
30	10	5	s-cv	38.3	2.7	3.2
30	11	6	cv	47.8	10.1	2.3
30	12	5	s-hv	52.2	3.7	8.6
30	12	3	s-cv	26.9	0.0	0.0
30	13	6	s-cv	49.4	4.3	4.2
30	14	5	hv	53.5	3.6	9.3
30	14	3	s-cv	29.1	0.1	0.0
30	15	4	s-hv	67.0	4.5	1.0
30	16	5	r	52.2	2.4	3.4
30	16	3	s-cv	50.0	4.0	1.6
30	17	5	hv	52.7	9.3	4.4
30	18	5	cv	52.9	9.2	1.3
30	18	3	s-cv	27.2	0.1	8.8

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
30	19	5	r	51.3	2.1	5.3
30	19	4	s-cv	21.6	0.0	1.4
30	20	5	hv	58.6	5.9	1.3
30	20	1	s-cv	21.8	0.0	1.4
30	21	5	cv	46.1	0.3	5.2
30	21	3	s-cv	48.2	8.9	3.6
30	22	5	s-hv	45.5	6.4	7.5
30	23	5	s-cv	45.1	4.4	5.7
30	24	5	s-cv	45.1	8.3	6.2
30	25	5	r	51.5	0.5	2.1
30	25	3	s-cv	28.1	7.4	0.0
30	26	5	cv	56.4	2.1	5.0
30	26	3	s-cv	34.2	2.6	8.1
30	27	5	hv	54.3	1.6	7.3
30	28	6	s-cv	64.6	0.6	6.0
30	29	5	s-cv	43.0	4.2	2.0
40	0	5	hv	58.3	4.4	0.1
40	0	4	s-cv	40.6	2.9	0.0
40	1	5	s-cv	55.8	2.9	3.6
40	1	3	s-cv	28.2	0.0	0.0
40	2	5	s-hv	62.2	9.0	3.7
40	2	4	s-cv	21.9	0.4	9.1
40	3	5	hv	53.8	10.6	0.6
40	3	3	s-cv	32.1	0.1	0.0
40	4	5	cv	51.3	0.7	1.5
40	4	1	s-cv	20.5	0.0	1.4
40	5	5	hv	56.9	7.8	5.0
40	5	3	s-cv	26.2	0.0	9.9
40	6	5	s-hv	55.0	3.0	6.9
40	6	4	s-cv	37.1	0.0	0.0
40	7	5	s-hv	57.6	7.0	0.2
40	8	5	hv	61.1	9.2	1.8
40	8	4	s-cv	60.7	0.2	2.7
40	9	5	s-hv	63.4	5.4	5.3
40	9	4	s-cv	55.9	8.8	5.3
40	10	5	s-hv	51.8	8.7	0.8
40	11	5	cv	59.0	0.1	5.7
40	11	4	s-cv	22.9	0.0	0.0
40	12	5	hv	59.9	1.9	3.2
40	12	3	s-cv	30.9	9.2	7.6
40	13	5	s-cv	54.2	3.5	3.8
40	13	1	s-cv	10.8	0.0	1.4
40	14	5	hv	54.3	1.9	8.7
40	14	4	s-cv	49.0	8.6	2.8
40	15	5	s-hv	56.9	7.0	8.8
40	15	4	s-cv	28.6	2.2	6.7
40	16	5	s-cv	52.0	1.6	1.0
40	16	3	s-cv	42.1	0.1	0.0
40	17	5	cv	56.0	10.0	5.7
40	17	3	s-cv	38.0	0.1	7.9
40	18	5	s-cv	63.0	2.3	1.9
40	18	3	s-cv	41.8	0.1	0.0
40	19	5	s-hv	51.6	8.9	2.5
40	20	5	s-hv	60.3	7.3	1.7
40	20	4	s-cv	33.0	0.3	7.0
40	21	5	hv	62.9	9.7	5.7
40	21	4	s-cv	24.9	4.6	1.8
40	22	5	s-cv	61.0	11.8	5.2
40	22	4	s-cv	48.7	5.4	9.3
40	23	5	hv	60.5	0.6	6.3
40	23	3	s-cv	7.1	0.0	0.0
40	24	5	s-cv	55.1	3.2	0.0
40	24	4	s-cv	36.4	0.0	8.6
40	25	5	hv	55.2	0.7	0.3
40	25	4	s-cv	50.6	9.8	9.4
40	26	5	hv	53.6	8.5	6.6
40	26	4	s-cv	50.7	6.8	0.2
40	27	5	s-cv	55.2	8.7	9.5
40	27	1	s-cv	29.8	0.0	10.8
40	28	5	hv	59.8	7.7	6.7
40	28	4	s-cv	57.2	7.0	7.3
40	29	5	s-hv	52.7	9.2	9.7
40	29	3	s-cv	57.2	0.1	15.7
50	0	5	s-hv	63.4	7.7	0.9
50	0	6	s-cv	62.0	2.0	1.1
50	1	5	cv	64.9	6.6	6.0
50	1	1	s-cv	13.5	0.1	23.9
50	2	5	hv	58.0	7.9	2.9
50	2	5	s-hv	42.6	2.0	6.4
50	3	5	cv	66.8	4.0	1.0
50	3	4	s-cv	23.4	2.0	0.0
50	4	5	s-hv	50.3	3.7	0.7
50	4	5	hv	58.9	0.9	5.8
50	4	1	s-cv	26.5	0.0	10.0
50	5	5	hv	55.0	13.8	3.6
50	5	5	cv	55.3	6.7	6.0
50	5	1	s-cv	21.0	3.5	1.4
50	6	5	cv	70.5	7.4	1.0
50	6	2	s-cv	43.7	0.8	1.8
50	7	5	r	56.4	10.1	1.1
50	7	5	s-cv	44.3	0.3	2.1
50	8	5	cv	52.4	4.8	7.6

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
50	8	4	s-cv	47.3	7.6	3.6
50	9	5	hv	59.3	9.0	0.6
50	10	5	s-cv	56.4	8.8	0.4
50	10	4	s-cv	59.9	8.4	4.9
50	11	5	hv	51.4	9.0	3.4
50	11	6	s-cv	53.1	0.6	6.7
50	12	5	s-cv	61.6	4.1	0.8
50	12	1	s-cv	23.2	4.2	16.0
50	13	5	s-hv	54.3	9.0	7.3
50	13	6	s-cv	59.3	5.1	1.7
50	14	5	hv	58.8	0.2	3.3
50	14	4	s-cv	24.3	3.1	1.2
50	15	5	s-cv	60.0	1.9	5.7
50	15	4	s-cv	26.3	0.0	6.6
50	16	5	s-cv	58.9	8.3	6.7
50	16	4	s-cv	44.1	0.4	5.5
50	17	5	s-hv	68.4	2.9	4.5
50	17	4	cv	60.4	2.3	8.5
50	18	5	hv	55.4	2.1	0.5
50	18	4	s-cv	28.8	0.0	0.5
50	19	5	s-hv	52.0	2.5	5.9
50	19	4	r	67.1	9.9	2.8
50	20	5	s-cv	54.3	6.0	5.6
50	21	5	hv	60.4	4.8	9.9
50	21	4	s-cv	38.3	6.4	6.5
50	22	5	s-cv	53.1	1.7	4.9
50	22	6	hv	73.4	9.9	7.8
50	23	5	cv	53.8	2.8	7.0
50	23	4	s-cv	47.2	2.1	7.4
50	24	5	s-hv	63.0	2.1	2.1
50	24	4	s-cv	31.9	0.1	5.0
50	25	5	hv	53.0	14.0	2.9
50	25	5	s-cv	47.7	7.9	6.8
50	26	5	cv	62.1	0.8	8.4
50	26	4	s-cv	34.1	0.8	8.3
50	27	5	cv	57.8	4.9	11.8
50	27	4	s-cv	39.6	4.1	0.7
50	28	5	cv	61.8	4.2	7.3
50	28	4	s-cv	63.7	7.4	5.8
50	29	5	cv	61.9	9.8	6.6
50	29	4	s-cv	25.1	5.9	1.8
60	0	5	hv	59.5	40.3	5.1
60	0	5	s-hv	52.6	5.5	1.6
60	0	1	s-cv	23.2	4.2	16.0
60	1	5	s-hv	54.3	3.6	1.0
60	1	5	hv	55.5	1.3	8.5
60	1	3	s-cv	29.1	0.0	0.0
60	2	5	s-hv	57.8	11.5	9.1
60	2	6	s-cv	61.4	2.7	1.7
60	3	5	cv	57.7	9.2	3.7
60	3	5	s-cv	54.0	0.7	9.6
60	4	5	hv	54.2	2.1	10.0
60	4	5	s-hv	50.2	9.9	7.8
60	4	1	s-cv	21.2	0.0	10.8
60	5	5	cv	60.1	6.6	1.0
60	5	6	s-hv	69.0	4.4	0.8
60	6	5	hv	49.8	14.1	3.6
60	6	5	cv	65.3	8.2	0.1
60	6	4	s-cv	28.6	8.1	8.8
60	7	5	cv	53.9	10.1	3.7
60	7	5	s-hv	59.2	5.1	4.5
60	7	1	s-cv	17.3	0.0	1.4
60	8	5	s-cv	60.8	6.0	5.4
60	8	4	s-cv	30.0	8.0	4.5
60	9	5	s-cv	60.0	3.9	0.6
60	9	6	hv	65.7	4.8	8.2
60	10	5	cv	57.2	3.9	3.1
60	10	5	s-cv	54.4	5.2	1.2
60	10	3	s-cv	30.8	0.0	8.0
60	11	5	hv	64.1	0.9	1.2
60	11	4	s-cv	45.5	6.2	7.5
60	12	5	s-hv	63.1	6.3	6.0
60	12	6	s-cv	65.6	1.8	2.1
60	13	5	hv	52.4	4.6	3.4
60	13	5	hv	61.1	11.8	6.7
60	13	4	s-cv	28.0	6.1	6.9
60	14	5	s-hv	63.8	9.0	0.8
60	14	4	s-cv	29.2	7.6	7.0
60	15	5	cv	64.9	3.1	5.4
60	15	5	s-hv	56.8	5.5	8.1
60	15	1	s-cv	31.1	0.0	10.0
60	16	5	cv	62.0	1.2	0.4
60	16	5	cv	60.3	1.9	1.9
60	16	3	s-cv	24.7	0.0	0.0
60	17	5	hv	63.2	9.5	3.6
60	17	6	s-cv	56.7	1.7	6.5
60	18	5	hv	50.4	12.0	4.2
60	18	5	s-hv	59.2	4.0	8.6
60	18	3	s-cv	43.2	0.1	0.0
60	19	5	s-cv	62.5	6.5	0.4
60	19	4	s-cv	53.1	0.0	7.6
60	20	5	s-cv	59.4	8.9	1.9

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
60	20	5	s-cv	45.2	3.7	6.2
60	21	5	hv	56.2	8.3	8.5
60	21	4	s-cv	59.0	8.3	9.3
60	22	5	cv	59.7	0.7	6.0
60	22	4	s-cv	33.6	0.0	2.5
60	23	5	s-hv	62.1	5.7	5.2
60	23	6	cv	57.7	7.9	7.9
60	24	5	hv	58.9	8.1	6.4
60	24	4	s-cv	39.2	6.2	8.8
60	25	5	hv	64.0	5.0	2.1
60	25	5	s-cv	51.6	7.4	6.6
60	26	5	s-cv	66.1	3.9	5.7
60	26	4	s-cv	46.2	0.5	6.7
60	27	5	cv	42.0	1.8	4.1
60	27	5	s-cv	58.7	9.2	2.3
60	27	3	s-cv	50.0	8.1	0.0
60	28	5	s-cv	59.1	1.4	5.3
60	28	5	s-cv	54.1	5.0	8.7
60	29	5	cv	57.1	3.0	8.4
60	29	5	s-hv	56.6	6.2	6.6
60	29	1	s-cv	21.4	0.0	10.0
70	0	5	hv	59.6	12.4	2.3
70	0	5	cv	57.6	6.5	2.6
70	0	3	s-cv	46.3	0.0	12.8
70	1	5	hv	67.9	7.3	0.2
70	1	4	s-cv	66.3	4.3	0.8
70	2	5	s-hv	63.7	8.2	7.0
70	2	5	cv	53.8	6.8	7.8
70	3	5	cv	52.7	0.9	1.5
70	3	5	hv	56.4	10.9	5.8
70	3	3	s-cv	49.5	0.0	1.0
70	4	5	s-cv	64.8	7.0	1.9
70	4	5	hv	59.8	3.5	1.9
70	4	4	s-cv	27.5	0.1	0.0
70	5	5	cv	61.6	6.3	6.0
70	5	5	s-cv	57.1	0.3	1.2
70	5	3	s-cv	48.9	0.1	0.0
70	6	5	s-hv	51.3	6.9	5.2
70	6	5	s-hv	58.2	3.0	9.3
70	6	4	hv	52.2	9.7	4.9
70	7	5	cv	60.0	2.5	4.0
70	7	5	s-hv	46.2	4.9	5.5
70	8	5	cv	59.4	5.0	8.6
70	8	5	s-cv	56.5	5.2	9.0
70	8	4	s-cv	35.2	0.0	3.4
70	9	5	s-hv	61.9	9.0	9.0
70	9	5	s-cv	58.9	9.7	1.1
70	9	4	s-cv	35.2	3.7	0.0
70	10	5	s-cv	58.6	1.9	8.2
70	10	5	s-cv	58.6	4.6	7.4
70	10	4	s-cv	37.6	3.5	0.3
70	11	5	hv	61.8	14.9	1.6
70	11	5	cv	63.6	3.6	1.2
70	12	5	hv	62.7	0.7	6.7
70	12	5	hv	62.1	8.7	0.6
70	12	3	s-cv	43.0	0.0	0.0
70	13	5	hv	56.1	8.5	4.2
70	13	5	s-cv	63.5	4.8	0.4
70	13	4	s-cv	40.6	4.8	0.0
70	14	5	cv	57.8	1.7	4.2
70	14	6	s-cv	66.2	6.7	8.8
70	15	5	s-cv	66.1	1.5	4.1
70	15	5	hv	59.6	8.2	0.8
70	15	4	s-cv	30.2	2.4	5.9
70	16	5	hv	64.0	1.7	8.2
70	16	6	s-cv	66.9	7.5	5.1
70	17	5	cv	60.3	0.3	1.7
70	17	5	hv	60.7	9.0	9.0
70	17	4	s-cv	34.0	0.1	4.3
70	18	5	s-hv	65.5	0.8	0.1
70	18	6	s-cv	61.5	3.8	3.4
70	19	5	cv	55.9	1.0	8.9
70	19	5	s-cv	60.0	8.6	4.6
70	19	1	s-cv	30.3	0.0	10.0
70	20	5	s-cv	72.5	6.4	3.7
70	20	5	hv	60.7	5.9	5.0
70	20	3	s-cv	27.0	0.0	3.7
70	21	5	s-hv	65.7	1.2	4.5
70	21	5	s-cv	46.6	2.4	1.9
70	22	5	cv	56.9	6.6	8.7
70	22	5	s-hv	61.5	6.5	5.8
70	22	4	s-cv	38.0	0.0	0.0
70	23	5	hv	66.6	7.8	2.5
70	23	5	cv	60.0	0.8	0.1
70	23	3	s-cv	43.4	0.1	2.1
70	24	5	hv	67.7	7.2	4.1
70	24	4	s-cv	32.8	1.1	0.0
70	25	5	s-cv	67.3	2.2	5.5
70	25	4	s-cv	59.1	8.2	4.0
70	26	5	s-cv	50.4	7.8	4.7
70	26	5	s-hv	62.4	9.9	7.3
70	26	4	s-cv	45.0	0.0	0.0

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
70	27	5	s-hv	62.1	5.2	6.3
70	27	5	s-cv	52.6	1.4	8.4
70	28	5	s-hv	65.6	4.5	7.5
70	28	5	hv	69.5	0.7	8.8
70	28	4	s-cv	29.0	0.0	3.7
70	29	5	hv	60.2	8.6	2.0
70	29	4	s-cv	46.6	2.2	1.6
80	0	5	s-cv	70.7	2.3	2.8
80	0	5	s-cv	54.6	5.9	9.2
80	1	5	s-hv	69.8	7.6	2.3
80	1	5	cv	61.7	6.0	3.4
80	1	4	s-cv	60.5	0.0	0.8
80	2	5	s-cv	66.1	1.6	1.1
80	2	5	s-hv	54.4	2.8	0.2
80	2	1	s-cv	18.3	0.0	16.4
80	3	5	s-cv	62.4	0.8	3.3
80	3	5	s-cv	60.6	2.8	6.7
80	3	4	s-cv	43.7	9.0	6.1
80	4	5	cv	66.1	8.2	6.7
80	4	5	s-hv	62.3	2.4	8.6
80	4	4	s-cv	29.6	0.4	5.3
80	5	5	cv	63.1	1.0	2.7
80	5	5	s-hv	61.8	2.3	3.4
80	5	6	cv	72.4	9.1	4.7
80	6	5	s-hv	68.1	4.3	7.1
80	6	5	cv	59.9	7.9	1.1
80	6	4	s-cv	42.3	0.0	7.0
80	7	5	s-cv	61.8	9.9	2.8
80	7	5	cv	40.9	7.3	8.3
80	7	5	hv	53.3	6.4	2.7
80	7	3	s-cv	34.3	0.1	0.0
80	8	5	s-hv	69.6	4.6	10.4
80	8	5	cv	60.2	9.2	4.4
80	8	3	s-cv	46.5	0.1	0.1
80	9	5	hv	64.2	9.5	5.3
80	9	5	cv	60.3	6.3	2.3
80	9	4	s-cv	22.2	3.0	0.0
80	10	5	s-cv	61.2	7.0	4.4
80	10	5	cv	61.0	0.1	1.2
80	10	4	s-cv	22.9	1.4	9.6
80	11	5	s-hv	68.5	8.2	3.0
80	11	5	cv	59.3	3.9	7.7
80	11	4	s-cv	45.4	6.9	0.2
80	12	5	hv	61.3	6.7	2.2
80	12	5	s-hv	53.9	7.3	0.6
80	13	5	cv	65.4	1.4	4.7
80	13	5	s-hv	61.7	7.7	1.3
80	13	1	s-cv	21.6	0.0	1.4
80	14	5	s-hv	70.2	6.4	5.5
80	14	6	s-hv	61.3	6.4	0.0
80	15	5	hv	65.7	1.8	1.7
80	15	5	r	64.0	1.9	2.2
80	15	4	s-cv	28.0	0.0	7.7
80	16	5	cv	48.3	4.4	4.3
80	16	5	cv	62.9	4.1	9.2
80	16	4	s-cv	34.3	5.4	5.3
80	17	5	hv	51.7	1.9	3.6
80	17	5	cv	63.6	0.5	1.4
80	17	3	s-cv	59.8	7.6	2.4
80	18	5	s-hv	67.4	1.0	0.7
80	18	5	r	59.2	9.0	9.2
80	18	1	s-cv	46.4	5.5	16.0
80	19	5	s-cv	75.2	0.9	4.6
80	19	5	hv	58.3	3.6	1.3
80	19	4	s-cv	31.3	1.5	7.5
80	20	5	s-hv	65.0	3.2	1.6
80	20	5	s-cv	56.9	0.8	5.9
80	21	5	cv	48.7	9.5	2.8
80	21	5	cv	63.0	6.6	1.5
80	21	5	s-cv	48.1	0.5	6.9
80	22	5	cv	58.3	8.8	9.6
80	22	5	s-cv	61.4	8.1	8.5
80	22	4	s-cv	53.6	2.8	8.0
80	23	5	s-cv	62.4	11.1	2.3
80	23	5	s-cv	56.7	11.7	7.6
80	23	4	s-hv	57.3	2.8	8.7
80	24	5	hv	60.2	0.2	0.3
80	24	5	cv	57.9	9.0	3.1
80	25	5	s-cv	62.6	7.5	5.3
80	25	5	cv	57.6	1.3	4.5
80	25	3	s-cv	36.2	0.1	0.0
80	26	5	s-cv	70.9	3.3	2.9
80	26	5	s-cv	53.2	4.5	1.4
80	27	5	s-cv	58.9	5.7	8.7
80	27	5	s-hv	59.0	4.2	4.8
80	27	4	s-cv	29.5	6.4	9.2
80	28	5	hv	63.9	7.1	1.4
80	28	5	cv	67.0	0.7	5.5
80	28	4	s-cv	37.1	7.5	1.5
80	29	5	cv	56.2	3.0	1.1
80	29	5	cv	68.0	0.6	2.7
80	29	5	cv	53.9	0.2	2.5

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
80	29	1	s-cv	21.8	7.4	11.7
90	0	5	cv	63.4	2.9	3.0
90	0	5	cv	61.9	1.8	7.9
90	0	4	s-cv	34.4	2.9	7.7
90	1	5	hv	66.6	8.7	4.3
90	1	5	s-hv	64.6	5.2	4.1
90	1	4	s-cv	29.1	7.9	0.0
90	2	5	cv	75.6	5.7	3.8
90	2	5	s-hv	61.4	1.4	8.7
90	2	4	s-cv	57.8	6.5	4.4
90	3	5	hv	57.9	4.2	1.7
90	3	5	cv	60.3	4.7	1.9
90	3	4	s-cv	31.6	0.0	9.7
90	4	5	hv	54.2	19.3	0.1
90	4	5	s-cv	66.5	5.5	3.2
90	4	5	s-hv	51.8	0.3	8.6
90	4	1	s-cv	22.5	0.0	10.0
90	5	5	cv	61.4	4.4	0.8
90	5	5	s-cv	59.1	3.1	8.5
90	5	3	s-hv	68.0	11.1	6.3
90	6	5	cv	54.2	6.3	3.7
90	6	5	s-hv	65.2	4.7	2.4
90	6	6	s-cv	63.5	5.1	12.3
90	7	5	cv	70.8	3.0	7.9
90	7	5	cv	60.8	7.3	8.7
90	7	4	s-cv	38.5	3.3	9.4
90	8	5	hv	68.4	3.2	6.3
90	8	5	s-cv	59.9	5.4	7.1
90	8	1	s-cv	21.2	0.0	10.8
90	9	5	s-cv	60.5	1.8	8.6
90	9	5	hv	61.6	8.2	0.9
90	9	4	s-cv	56.7	2.2	1.3
90	10	5	hv	53.6	8.3	10.6
90	10	5	s-cv	58.1	3.7	9.4
90	10	5	hv	58.9	10.2	0.3
90	10	4	s-cv	25.6	2.0	0.0
90	11	5	s-cv	66.4	0.3	0.2
90	11	5	hv	58.8	5.0	5.5
90	11	4	s-cv	27.8	1.8	0.5
90	12	5	s-cv	74.3	9.1	2.8
90	12	5	s-hv	52.6	7.6	2.6
90	12	1	s-cv	11.2	0.0	10.0
90	13	5	s-hv	66.1	5.0	1.3
90	13	5	s-hv	57.3	0.3	0.5
90	14	5	cv	65.6	8.5	6.2
90	14	5	s-cv	58.9	6.6	5.1
90	14	5	s-cv	49.2	2.9	8.0
90	15	5	s-hv	72.1	3.7	2.8
90	15	5	hv	59.7	3.8	3.7
90	15	4	s-cv	28.5	0.0	7.2
90	16	5	s-hv	68.9	6.7	2.8
90	16	5	s-hv	58.1	6.9	8.9
90	16	4	s-cv	28.4	0.0	2.1
90	17	5	cv	69.1	8.1	2.9
90	17	5	s-hv	64.5	5.3	3.4
90	17	6	s-cv	53.9	0.0	2.6
90	18	5	hv	61.3	0.5	0.4
90	18	5	r	64.0	1.8	2.4
90	18	6	s-cv	50.5	6.8	8.7
90	19	5	s-hv	67.1	9.2	8.0
90	19	5	s-hv	62.8	3.7	6.8
90	19	4	s-cv	34.3	3.4	1.9
90	20	5	s-cv	64.6	1.9	2.5
90	20	5	s-cv	67.8	6.3	8.5
90	20	4	s-hv	64.5	3.0	4.2
90	21	5	s-cv	67.4	6.9	2.4
90	21	5	s-cv	65.2	9.0	4.8
90	21	5	s-cv	54.2	0.1	8.7
90	22	5	s-cv	66.5	4.7	4.1
90	22	5	s-cv	58.1	6.3	2.1
90	22	4	s-cv	36.5	5.0	1.1
90	23	5	cv	64.1	8.8	4.6
90	23	5	cv	59.6	9.5	0.7
90	23	4	s-cv	38.2	0.2	0.0
90	24	5	s-cv	55.7	12.5	7.3
90	24	5	s-cv	65.3	7.9	1.8
90	24	4	s-cv	63.9	2.0	9.5
90	25	5	s-cv	64.7	4.8	7.6
90	25	5	cv	58.6	2.4	2.0
90	25	3	s-cv	46.4	0.0	4.9
90	26	5	s-hv	62.5	2.6	3.0
90	26	5	cv	62.5	3.3	2.6
90	26	4	s-cv	40.9	7.0	6.7
90	27	5	hv	68.0	6.8	6.8
90	27	5	r	63.1	4.3	4.7
90	27	4	s-cv	52.1	7.0	5.5
90	28	5	s-cv	63.9	8.7	7.0
90	28	5	hv	59.3	1.2	7.8
90	28	4	s-cv	68.8	1.7	2.4
90	29	5	s-cv	49.0	9.8	4.8
90	29	5	s-cv	60.1	1.7	1.4
90	29	5	s-cv	56.5	8.6	4.7

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
90	29	1	s-cv	32.6	0.0	10.0
100	0	5	hv	66.2	4.6	4.7
100	0	5	s-hv	53.6	2.3	0.1
100	1	5	s-cv	69.3	3.4	7.9
100	1	5	cv	66.7	5.0	7.7
100	1	6	cv	60.5	9.0	1.0
100	2	5	hv	64.3	6.3	2.0
100	2	5	cv	68.0	1.4	9.2
100	2	5	s-cv	52.2	6.2	5.3
100	3	5	s-cv	59.3	0.4	6.0
100	3	5	s-hv	65.2	6.9	2.3
100	3	5	s-hv	54.2	0.2	0.1
100	3	3	s-cv	52.7	0.1	5.9
100	4	5	s-hv	61.6	7.2	8.3
100	4	5	s-cv	63.3	6.1	5.8
100	4	6	s-cv	65.7	3.3	2.8
100	5	5	s-hv	60.0	5.5	0.7
100	5	5	s-hv	63.7	5.9	4.3
100	5	4	s-cv	39.2	0.5	9.7
100	6	5	hv	76.6	0.2	1.7
100	6	5	s-cv	67.0	1.2	6.8
100	6	5	s-hv	56.6	7.6	7.4
100	6	3	s-cv	44.3	0.0	0.0
100	7	5	s-cv	70.0	6.3	6.3
100	7	5	s-cv	60.6	3.8	2.4
100	7	3	s-cv	35.5	0.1	0.0
100	8	5	s-cv	57.4	1.5	1.8
100	8	5	r	65.9	5.7	7.2
100	8	4	s-cv	70.3	1.6	8.2
100	9	5	cv	70.4	7.0	3.8
100	9	5	s-cv	61.7	5.3	2.3
100	9	5	s-hv	51.3	6.0	0.6
100	10	5	hv	70.9	1.7	7.8
100	10	5	cv	64.6	6.1	4.0
100	10	4	s-cv	64.5	2.6	11.5
100	11	5	s-hv	52.9	9.8	2.4
100	11	5	cv	56.2	7.2	4.8
100	11	5	s-hv	55.5	0.8	8.8
100	11	3	s-cv	26.6	0.0	0.0
100	12	5	hv	58.2	9.0	7.9
100	12	5	s-cv	74.7	6.9	6.6
100	12	5	cv	59.5	5.3	5.2
100	12	4	s-cv	44.4	0.0	8.6
100	13	5	s-hv	68.8	2.5	4.4
100	13	5	s-hv	62.8	7.1	1.9
100	13	4	s-cv	34.8	0.0	2.9
100	14	5	hv	60.6	6.8	3.2
100	14	5	hv	59.9	6.6	2.9
100	14	5	s-cv	51.3	10.9	4.6
100	15	5	s-hv	60.0	4.5	1.3
100	15	5	cv	62.3	1.8	3.4
100	15	5	s-cv	52.4	0.6	7.0
100	15	1	s-cv	34.6	0.0	10.0
100	16	5	cv	64.2	9.3	7.3
100	16	5	cv	65.8	8.2	2.7
100	16	5	hv	54.2	6.2	0.5
100	16	1	s-cv	21.4	0.0	10.0
100	17	5	s-hv	73.8	6.2	7.8
100	17	5	r	59.8	6.5	3.4
100	17	4	s-cv	44.4	0.1	2.9
100	18	5	cv	66.7	8.1	0.1
100	18	5	s-hv	63.9	2.5	2.9
100	18	3	s-cv	71.0	0.0	9.7
100	19	5	s-hv	61.2	2.0	3.8
100	19	5	s-hv	62.9	5.5	7.0
100	19	6	s-cv	54.9	7.7	2.0
100	20	5	s-cv	63.2	7.8	0.2
100	20	5	hv	67.0	5.2	9.2
100	20	4	s-cv	60.6	0.7	2.1
100	21	5	cv	57.6	27.5	14.9
100	21	5	cv	73.0	0.7	0.2
100	21	5	cv	55.4	4.2	9.6
100	21	3	s-cv	28.9	0.0	8.0
100	22	5	hv	59.4	9.2	2.3
100	22	5	cv	68.0	6.3	7.5
100	22	5	s-hv	58.2	5.4	8.7
100	22	3	s-cv	32.2	0.1	0.0
100	23	5	s-cv	71.6	4.1	6.7
100	23	5	s-cv	67.8	9.3	1.4
100	23	4	s-cv	51.2	3.0	6.4
100	24	5	s-hv	68.8	7.8	8.4
100	24	5	cv	63.1	4.7	2.3
100	24	6	s-hv	71.1	0.8	8.0
100	25	5	cv	64.9	7.6	0.0
100	25	5	hv	66.0	7.8	4.2
100	25	5	hv	55.1	0.1	4.0
100	25	3	s-cv	45.6	0.1	4.5
100	26	5	s-cv	51.2	3.4	6.2
100	26	5	s-cv	63.2	5.6	1.0
100	26	5	r	58.1	4.8	7.1
100	26	4	s-cv	27.3	2.0	3.8
100	27	5	hv	67.6	2.4	0.3

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
100	27	5	cv	63.2	8.0	6.6
100	27	6	s-cv	57.2	0.5	7.8
100	28	5	s-cv	68.4	2.8	1.7
100	28	5	s-hv	62.4	9.2	3.9
100	28	6	r	70.9	5.5	8.4
100	29	5	hv	51.9	22.7	5.3
100	29	5	r	58.8	2.7	7.6
100	29	5	s-cv	63.6	6.3	5.3
100	29	4	s-cv	44.1	3.7	6.0

Table 12: Detailed results for adapted Paquay *et al.* (2018b) instances where each ULD type is available only once.

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
10	0	41	10	1	24.9
10	1	101	10	1	27.9
10	2	81	10	1	41.6
10	3	94	10	1	27.4
10	4	57	10	1	56.1
10	5	22	10	1	13.1
10	6	44	10	1	48.5
10	7	60	10	1	58.5
10	8	94	10	1	38.3
10	9	80	10	1	48.2
10	10	56	10	1	26.4
10	11	110	10	1	26.0
10	12	59	10	1	46.6
10	13	40	10	1	55.0
10	14	61	10	1	62.6
10	15	64	10	1	48.1
10	16	98	10	1	35.9
10	17	96	10	1	25.8
10	18	93	10	1	29.7
10	19	58	10	1	42.5
10	20	44	10	1	37.1
10	21	90	10	1	38.4
10	22	43	10	1	15.8
10	23	96	10	1	27.9
10	24	97	10	1	26.1
10	25	44	10	1	23.3
10	26	103	10	1	16.1
10	27	63	10	1	51.8
10	28	61	10	1	50.5
10	29	43	10	1	41.9
20	0	265	20	1	58.9
20	1	60	20	1	39.4
20	2	121	20	1	42.4
20	3	259	20	1	47.2
20	4	183	20	1	48.1
20	5	64	20	1	33.3
20	6	104	20	1	27.2
20	7	287	20	1	61.5
20	8	175	20	1	43.4
20	9	222	20	1	67.3
20	10	73	20	2	55.3
20	11	96	20	1	47.2
20	12	62	20	1	40.0
20	13	292	20	1	62.6
20	14	121	20	1	40.5
20	15	98	20	1	30.5
20	16	267	20	1	44.5
20	17	268	20	1	55.9
20	18	116	20	1	35.7
20	19	59	20	2	41.6
20	20	67	20	1	46.8
20	21	130	20	1	46.0
20	22	69	20	2	49.9
20	23	230	20	1	38.2
20	24	280	20	1	51.8
20	25	238	20	1	57.9
20	26	62	20	1	30.5
20	27	241	20	1	53.4
20	28	244	20	1	56.2
20	29	64	20	1	46.6
30	0	129	30	2	49.2
30	1	669	30	1	57.6
30	2	570	30	1	61.0
30	3	135	30	2	39.5
30	4	112	30	2	50.9
30	5	130	30	2	50.7
30	6	115	30	2	49.5
30	7	579	30	1	52.7
30	8	570	30	1	50.3
30	9	115	30	2	50.2
30	10	124	30	1	38.3
30	11	573	30	1	47.8
30	12	120	30	2	47.1
30	13	579	30	1	49.4
30	14	117	30	2	48.6
30	15	485	30	1	67.0
30	16	121	30	2	51.7
30	17	116	30	1	52.7
30	18	124	30	2	47.8
30	19	144	30	2	39.8
30	20	110	30	2	53.3
30	21	132	30	2	46.5
30	22	348	30	1	45.5
30	23	123	30	1	45.1
30	24	583	30	1	45.1
30	25	123	30	2	46.8
30	26	127	30	2	51.9
30	27	488	30	1	54.3
30	28	595	30	1	64.6
30	29	119	30	1	43.0
40	0	221	40	2	51.5
40	1	183	40	2	50.3

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
40	2	214	40	2	46.7
40	3	186	40	2	49.5
40	4	195	40	2	46.9
40	5	185	40	2	50.8
40	6	200	40	2	48.1
40	7	854	40	1	57.6
40	8	211	40	2	60.9
40	9	215	40	2	60.5
40	10	543	40	1	51.8
40	11	206	40	2	45.1
40	12	205	40	2	54.1
40	13	203	40	2	47.9
40	14	225	40	2	52.3
40	15	183	40	2	46.0
40	16	198	40	2	50.1
40	17	211	40	2	52.4
40	18	178	40	2	58.8
40	19	384	40	1	51.6
40	20	208	40	2	49.7
40	21	190	40	2	48.2
40	22	215	40	2	56.2
40	23	197	40	2	49.9
40	24	196	40	2	47.9
40	25	204	40	2	53.4
40	26	221	40	2	52.5
40	27	161	40	2	51.6
40	28	211	40	2	58.8
40	29	183	40	2	53.6
50	0	310	50	2	62.8
50	1	295	50	2	57.5
50	2	265	50	3	50.8
50	3	324	50	2	50.0
50	4	269	50	3	57.4
50	5	253	50	3	57.6
50	6	273	50	2	64.1
50	7	292	50	2	58.2
50	8	335	50	2	50.4
50	9	285	50	1	59.3
50	10	297	50	2	57.7
50	11	281	50	2	52.1
50	12	284	50	2	56.1
50	13	296	50	2	56.4
50	14	309	50	2	45.5
50	15	291	50	2	47.0
50	16	293	50	2	53.2
50	17	273	50	2	65.3
50	18	318	50	2	45.1
50	19	329	50	2	54.5
50	20	548	50	1	54.3
50	21	295	50	2	51.9
50	22	318	50	2	61.7
50	23	316	50	2	51.2
50	24	307	50	2	51.0
50	25	265	50	3	50.9
50	26	259	50	2	51.3
50	27	324	50	2	50.8
50	28	263	50	2	62.5
50	29	280	50	2	47.7
60	0	434	60	3	58.6
60	1	398	60	3	49.6
60	2	439	60	2	59.3
60	3	335	60	3	56.5
60	4	450	60	3	45.8
60	5	394	60	3	58.2
60	6	307	60	3	56.4
60	7	396	60	3	58.6
60	8	394	60	2	48.9
60	9	433	60	2	62.4
60	10	361	60	3	60.3
60	11	396	60	2	56.9
60	12	349	60	2	64.2
60	13	363	60	3	55.6
60	14	355	60	2	50.4
60	15	336	60	3	62.3
60	16	327	60	3	65.0
60	17	412	60	2	60.4
60	18	387	60	3	51.0
60	19	405	60	2	58.9
60	20	355	60	3	52.9
60	21	439	60	2	57.2
60	22	414	60	2	49.6
60	23	420	60	2	60.2
60	24	431	60	2	51.3
60	25	326	60	3	60.9
60	26	378	60	2	58.4
60	27	387	60	3	48.0
60	28	336	60	3	59.7
60	29	344	60	3	59.3
70	0	461	70	3	65.1
70	1	523	70	2	67.3
70	2	507	70	3	59.4
70	3	556	70	3	61.4

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
70	4	442	70	3	71.7
70	5	478	70	3	55.5
70	6	500	70	4	56.3
70	7	573	70	3	56.0
70	8	435	70	3	58.5
70	9	453	70	3	60.6
70	10	459	70	3	59.7
70	11	455	70	3	63.3
70	12	414	70	3	57.5
70	13	473	70	3	61.5
70	14	545	70	2	61.3
70	15	425	70	3	61.4
70	16	580	70	2	65.2
70	17	434	70	3	60.3
70	18	483	70	2	63.8
70	19	503	70	3	61.1
70	20	403	70	3	59.3
70	21	462	70	3	59.2
70	22	488	70	3	60.3
70	23	478	70	3	58.2
70	24	456	70	2	54.2
70	25	518	70	2	64.2
70	26	394	70	4	54.1
70	27	504	70	3	58.0
70	28	385	70	3	65.0
70	29	536	70	2	54.9
80	0	594	80	3	63.3
80	1	506	80	3	71.9
80	2	582	80	3	62.5
80	3	664	80	3	63.8
80	4	539	80	3	62.3
80	5	513	80	4	68.2
80	6	643	80	3	65.6
80	7	555	80	5	56.4
80	8	564	80	3	59.9
80	9	671	80	3	58.7
80	10	628	80	3	57.9
80	11	602	80	3	66.3
80	12	731	80	3	58.2
80	13	555	80	3	66.1
80	14	644	80	2	66.5
80	15	568	80	3	62.4
80	16	579	80	3	56.3
80	17	601	80	3	55.2
80	18	515	80	3	67.9
80	19	603	80	3	64.9
80	20	700	80	3	64.3
80	21	577	80	4	61.3
80	22	598	80	4	60.8
80	23	541	80	3	65.8
80	24	689	80	3	62.2
80	25	665	80	3	54.8
80	26	600	80	3	62.7
80	27	548	80	3	57.8
80	28	570	80	4	59.1
80	29	509	80	5	62.2
90	0	829	90	3	62.3
90	1	743	90	3	63.4
90	2	816	90	3	73.5
90	3	815	90	3	58.5
90	4	699	90	5	60.4
90	5	842	90	3	58.3
90	6	806	90	4	63.5
90	7	694	90	3	66.0
90	8	837	90	3	66.6
90	9	584	90	4	62.5
90	10	765	89	5	62.2
90	11	727	90	3	60.5
90	12	710	90	3	65.1
90	13	866	90	3	65.0
90	14	803	90	5	61.2
90	15	726	90	3	63.5
90	16	744	90	3	61.4
90	17	692	90	4	68.4
90	18	667	90	4	62.2
90	19	868	90	3	64.2
90	20	674	90	4	66.3
90	21	645	90	5	64.0
90	22	767	90	3	62.6
90	23	693	90	3	62.6
90	24	657	90	4	61.8
90	25	779	90	3	57.2
90	26	840	90	4	57.8
90	27	734	90	4	64.9
90	28	653	90	4	65.9
90	29	721	90	5	58.6
100	0	1039	100	3	60.5
100	1	883	100	5	61.7
100	2	817	100	5	63.2
100	3	806	98	5	63.9
100	4	847	100	4	66.3
100	5	1041	100	3	62.9

Continued on next column

$\|I\|$	No.	t	$\|L\|$	$\|C\|$	u^{total}
100	6	740	100	6	68.4
100	7	909	100	3	59.1
100	8	956	100	4	64.3
100	9	816	100	4	70.3
100	10	981	100	4	67.5
100	11	1049	99	5	58.0
100	12	787	95	6	64.9
100	13	997	100	3	65.1
100	14	947	99	5	59.8
100	15	868	100	5	61.9
100	16	924	97	5	62.2
100	17	813	100	3	68.4
100	18	1018	100	3	62.8
100	19	947	100	4	62.9
100	20	867	100	4	64.6
100	21	864	100	6	62.6
100	22	829	100	5	66.4
100	23	808	100	3	72.8
100	24	832	100	5	64.8
100	25	780	98	5	63.9
100	26	862	99	6	60.4
100	27	914	100	4	68.2
100	28	828	100	5	64.4
100	29	883	92	6	57.0

Table 13: Detailed ULD results for adapted Paquay *et al.* (2018b) instances where each ULD type is available only once.

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
10	0	5	s-cv	24.9	3.0	2.6
10	1	4	s-cv	27.9	6.5	6.6
10	2	4	s-cv	41.6	13.9	8.9
10	3	4	s-cv	27.4	1.8	6.8
10	4	3	s-cv	56.1	0.2	2.9
10	5	5	s-cv	13.1	1.8	0.0
10	6	1	cv	48.5	9.3	24.7
10	7	3	s-cv	58.5	1.5	6.6
10	8	4	s-cv	38.3	2.7	5.1
10	9	4	s-cv	48.2	1.9	1.3
10	10	5	s-cv	26.4	0.4	7.2
10	11	4	s-cv	26.0	1.9	5.7
10	12	3	s-cv	46.6	9.2	6.6
10	13	3	s-cv	55.0	4.1	0.6
10	14	3	s-hv	62.6	1.2	2.7
10	15	3	s-cv	48.1	5.6	3.0
10	16	4	s-cv	35.9	1.5	1.5
10	17	4	s-cv	25.8	5.5	1.5
10	18	4	s-cv	29.7	9.9	1.3
10	19	3	s-cv	42.5	5.7	0.0
10	20	1	s-cv	37.1	2.1	20.6
10	21	4	s-cv	38.4	6.6	0.4
10	22	5	s-cv	15.8	0.8	0.0
10	23	4	s-cv	27.9	2.2	0.3
10	24	4	s-cv	26.1	5.2	0.0
10	25	4	s-cv	23.3	2.3	3.2
10	26	4	s-cv	16.1	7.6	8.4
10	27	3	s-cv	51.8	6.3	2.1
10	28	3	s-cv	50.5	6.0	3.2
10	29	3	s-cv	41.9	2.5	2.0
20	0	4	s-cv	58.9	8.6	5.8
20	1	5	s-cv	39.4	3.5	0.1
20	2	4	s-hv	42.4	1.2	7.4
20	3	4	s-cv	47.2	3.8	0.2
20	4	5	s-hv	48.1	3.8	3.7
20	5	5	s-cv	33.3	0.5	9.8
20	6	5	s-cv	27.2	6.4	0.1
20	7	6	s-cv	61.5	6.0	1.1
20	8	3	s-cv	43.4	6.1	2.3
20	9	4	s-cv	67.3	4.7	3.6
20	10	5	hv	56.4	0.3	8.3
20	10	3	s-cv	50.7	0.1	2.1
20	11	5	s-cv	47.2	4.4	3.2
20	12	5	s-cv	40.0	4.1	9.4
20	13	6	s-cv	62.6	0.0	0.9
20	14	4	s-cv	40.5	7.7	1.4
20	15	5	s-cv	30.5	7.9	5.8
20	16	4	s-cv	44.5	4.5	10.0
20	17	4	s-cv	55.9	6.1	5.3
20	18	5	s-cv	35.7	6.5	7.0
20	19	5	hv	48.6	7.8	8.9
20	19	3	s-cv	13.8	0.0	0.0
20	20	5	s-cv	46.8	1.5	2.6
20	21	4	s-cv	46.0	7.8	1.1
20	22	5	hv	52.2	5.6	5.6
20	22	3	s-cv	40.9	0.0	0.0
20	23	4	s-cv	38.2	7.5	0.3
20	24	6	s-cv	51.8	5.2	4.1
20	25	4	s-hv	57.9	3.4	7.9
20	26	5	s-cv	30.5	8.5	7.7
20	27	4	s-cv	53.4	5.3	12.8
20	28	4	s-cv	56.2	9.6	4.2
20	29	5	s-hv	46.6	8.0	9.8
30	0	5	s-cv	54.3	3.0	5.0
30	0	1	s-cv	18.6	0.0	10.8
30	1	6	s-cv	57.6	8.2	2.8
30	2	6	s-cv	61.0	1.7	5.0
30	3	5	s-cv	48.3	4.8	6.6
30	3	4	s-cv	25.6	0.0	7.6
30	4	5	cv	62.5	9.2	9.4
30	4	3	s-cv	4.2	0.0	0.0
30	5	5	s-cv	55.9	4.5	2.4
30	5	3	s-cv	30.0	0.0	0.0
30	6	5	hv	55.8	8.3	8.0
30	6	3	s-cv	24.2	0.1	0.0
30	7	5	cv	52.7	0.8	5.2
30	8	5	s-cv	50.3	3.3	8.9
30	9	5	s-cv	55.0	1.3	2.3
30	9	1	s-cv	21.6	0.0	1.4
30	10	5	s-cv	38.3	2.7	3.2
30	11	6	cv	47.8	10.1	2.3
30	12	5	s-hv	52.2	3.7	8.6
30	12	3	s-cv	26.9	0.0	0.0
30	13	6	s-cv	49.4	4.3	4.2
30	14	5	hv	53.5	3.6	9.3
30	14	3	s-cv	29.1	0.1	0.0
30	15	4	s-hv	67.0	4.5	1.0
30	16	5	r	52.2	2.4	3.4
30	16	3	s-cv	50.0	4.0	1.6
30	17	5	hv	52.7	9.3	4.4
30	18	5	cv	52.9	9.2	1.3
30	18	3	s-cv	27.2	0.1	8.8

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
30	19	5	r	51.3	2.1	5.3
30	19	4	s-cv	21.6	0.0	1.4
30	20	5	hv	58.6	5.9	1.3
30	20	1	s-cv	21.8	0.0	1.4
30	21	5	cv	46.1	0.3	5.2
30	21	3	s-cv	48.2	8.9	3.6
30	22	5	s-hv	45.5	6.4	7.5
30	23	5	s-cv	45.1	4.4	5.7
30	24	5	s-cv	45.1	8.3	6.2
30	25	5	r	51.5	0.5	2.1
30	25	3	s-cv	28.1	7.4	0.0
30	26	5	cv	56.4	2.1	5.0
30	26	3	s-cv	34.2	2.6	8.1
30	27	5	hv	54.3	1.6	7.3
30	28	6	s-cv	64.6	0.6	6.0
30	29	5	s-cv	43.0	4.2	2.0
40	0	5	hv	58.3	4.4	0.1
40	0	4	s-cv	40.6	2.9	0.0
40	1	5	s-cv	55.8	2.9	3.6
40	1	3	s-cv	28.2	0.0	0.0
40	2	5	s-hv	62.2	9.0	3.7
40	2	4	s-cv	21.9	0.4	9.1
40	3	5	hv	53.8	10.6	0.6
40	3	3	s-cv	32.1	0.1	0.0
40	4	5	cv	51.3	0.7	1.5
40	4	1	s-cv	20.5	0.0	1.4
40	5	5	hv	56.9	7.8	5.0
40	5	3	s-cv	26.2	0.0	9.9
40	6	5	s-hv	55.0	3.0	6.9
40	6	4	s-cv	37.1	0.0	0.0
40	7	5	s-hv	57.6	7.0	0.2
40	8	5	hv	61.1	9.2	1.8
40	8	4	cv	60.7	3.0	2.7
40	9	5	s-hv	63.4	5.4	5.3
40	9	4	s-cv	55.9	8.8	5.3
40	10	5	s-hv	51.8	8.7	0.8
40	11	5	cv	59.0	0.1	5.7
40	11	4	s-cv	22.9	0.0	0.0
40	12	5	hv	59.9	1.9	3.2
40	12	3	s-cv	30.9	9.2	7.6
40	13	5	s-cv	54.2	3.5	3.8
40	13	1	s-cv	10.8	0.0	1.4
40	14	5	hv	54.3	1.9	8.7
40	14	4	s-cv	49.0	8.6	2.8
40	15	5	s-hv	56.9	7.0	8.8
40	15	4	s-cv	28.6	2.2	6.7
40	16	5	s-cv	52.0	1.6	1.0
40	16	3	s-cv	42.1	0.1	0.0
40	17	5	cv	56.0	10.0	5.7
40	17	3	s-cv	38.0	0.1	7.9
40	18	5	s-cv	63.0	2.3	1.9
40	18	3	s-cv	41.8	0.1	0.0
40	19	5	s-hv	51.6	8.9	2.5
40	20	5	s-hv	60.3	7.3	1.7
40	20	4	s-cv	33.0	0.3	7.0
40	21	5	hv	62.9	9.7	5.7
40	21	4	s-cv	24.9	4.6	1.8
40	22	5	s-cv	61.0	11.8	5.2
40	22	4	s-cv	48.7	5.4	9.3
40	23	5	hv	60.5	0.6	6.3
40	23	3	s-cv	7.1	0.0	0.0
40	24	5	s-cv	55.1	3.2	0.0
40	24	4	s-cv	36.4	0.0	8.6
40	25	5	hv	55.2	0.7	0.3
40	25	4	s-cv	50.6	9.8	9.4
40	26	5	hv	53.6	8.5	6.6
40	26	4	s-cv	50.7	6.7	0.2
40	27	5	s-cv	55.2	8.7	9.5
40	27	1	s-cv	29.8	0.0	10.8
40	28	5	hv	59.8	7.7	6.7
40	28	4	s-cv	57.2	0.3	7.6
40	29	5	s-hv	52.7	9.2	9.7
40	29	3	s-cv	57.2	0.1	15.7
50	0	5	s-hv	63.4	7.7	0.9
50	0	6	s-cv	62.0	7.7	0.8
50	1	5	cv	64.9	6.6	6.0
50	1	1	s-cv	13.5	0.1	23.9
50	2	5	hv	58.0	7.9	2.9
50	2	6	s-hv	48.7	3.6	8.7
50	2	3	s-cv	28.1	0.1	13.7
50	3	5	cv	66.8	4.0	1.0
50	3	4	s-cv	23.4	2.0	0.0
50	4	5	s-hv	50.3	3.7	0.7
50	4	6	s-cv	72.2	0.9	3.3
50	4	3	s-cv	42.8	8.3	6.9
50	5	5	hv	55.0	13.8	3.6
50	5	6	s-hv	64.7	6.8	7.4
50	5	3	s-cv	46.7	11.5	0.0
50	6	5	cv	70.5	7.4	1.0
50	6	2	s-cv	43.7	0.8	1.8
50	7	5	r	56.4	10.1	1.1
50	7	6	s-hv	60.7	3.2	2.9

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
50	8	5	cv	52.4	4.8	7.6
50	8	4	s-cv	47.3	7.6	3.6
50	9	5	hv	59.3	9.0	0.6
50	10	5	s-cv	56.4	8.8	0.4
50	10	4	s-cv	59.9	8.4	4.9
50	11	5	hv	51.4	9.0	3.4
50	11	6	s-cv	53.1	2.2	0.5
50	12	5	s-cv	61.6	4.1	0.8
50	12	1	s-cv	23.2	4.2	16.0
50	13	5	s-hv	54.3	9.0	7.3
50	13	6	s-hv	59.3	4.9	6.1
50	14	5	hv	58.8	0.2	3.3
50	14	4	s-cv	24.3	3.1	1.2
50	15	5	s-cv	60.0	1.9	5.7
50	15	4	s-cv	26.3	0.0	6.6
50	16	5	s-cv	58.9	8.3	6.7
50	16	4	s-cv	44.1	0.4	5.5
50	17	5	s-hv	68.4	2.9	4.5
50	17	4	s-cv	60.4	2.4	5.5
50	18	5	hv	55.4	2.1	0.5
50	18	4	s-cv	28.8	0.0	0.5
50	19	5	s-hv	52.0	2.5	5.9
50	19	6	s-cv	57.9	6.6	9.3
50	20	5	s-cv	54.3	6.0	5.6
50	21	5	hv	60.4	4.8	9.9
50	21	4	s-cv	38.3	6.4	6.5
50	22	5	s-cv	53.1	1.7	4.9
50	22	6	s-cv	73.4	0.2	0.6
50	23	5	cv	53.8	2.8	7.0
50	23	4	s-cv	47.2	0.6	2.0
50	24	5	s-hv	63.0	2.1	2.1
50	24	4	s-cv	31.9	0.1	2.6
50	25	5	hv	53.0	14.0	2.9
50	25	6	s-hv	60.7	1.3	1.3
50	25	3	s-cv	13.8	0.0	0.0
50	26	5	cv	62.1	0.8	8.4
50	26	4	s-cv	34.1	0.8	8.3
50	27	5	cv	57.8	4.9	11.8
50	27	4	s-cv	39.6	4.1	0.7
50	28	5	cv	61.8	4.2	7.3
50	28	4	s-cv	63.7	7.9	6.7
50	29	5	cv	61.9	9.8	6.6
50	29	4	s-cv	25.1	5.9	1.8
60	0	5	hv	59.5	40.3	5.1
60	0	6	hv	76.0	3.1	8.8
60	0	3	s-cv	4.2	0.0	0.0
60	1	5	s-hv	54.3	3.6	1.0
60	1	6	s-cv	72.2	6.5	8.9
60	1	4	s-cv	16.0	0.0	0.0
60	2	5	s-hv	57.8	11.5	9.1
60	2	6	s-cv	61.4	4.2	1.3
60	3	5	cv	57.7	9.2	3.7
60	3	6	s-cv	67.0	0.8	7.6
60	3	3	s-cv	20.7	0.0	0.0
60	4	5	hv	54.2	2.1	10.0
60	4	6	s-hv	66.9	7.5	4.1
60	4	4	s-cv	7.9	0.0	0.0
60	5	5	cv	60.1	6.6	1.0
60	5	6	s-hv	64.9	5.5	0.1
60	5	1	s-cv	17.6	0.0	2.7
60	6	5	hv	49.8	14.1	3.6
60	6	6	r	72.9	0.5	9.8
60	6	4	s-cv	47.8	3.7	2.9
60	7	5	cv	53.9	10.1	3.7
60	7	6	hv	76.2	9.0	4.6
60	7	3	s-cv	25.9	0.0	0.0
60	8	5	s-cv	60.8	6.0	5.4
60	8	4	s-cv	30.0	8.0	4.5
60	9	5	s-cv	60.0	3.9	0.6
60	9	6	s-cv	65.7	8.5	4.8
60	10	5	cv	57.2	3.9	3.1
60	10	6	cv	74.3	5.8	7.1
60	10	3	s-cv	31.5	0.0	0.0
60	11	5	hv	64.1	0.9	1.2
60	11	4	s-cv	45.5	6.2	7.5
60	12	5	s-hv	63.1	6.3	6.0
60	12	6	s-cv	65.6	4.1	5.5
60	13	5	hv	52.4	4.6	3.4
60	13	6	hv	71.6	1.6	1.8
60	13	4	s-cv	42.1	3.9	7.3
60	14	5	s-hv	63.8	9.0	0.8
60	14	4	s-cv	29.2	5.4	7.0
60	15	5	cv	64.9	3.1	5.4
60	15	6	s-hv	70.2	10.2	1.7
60	15	2	s-cv	34.9	8.8	3.1
60	16	5	cv	62.0	1.2	0.4
60	16	6	hv	71.2	5.9	3.0
60	16	3	s-cv	58.4	3.7	0.0
60	17	5	hv	63.2	9.5	3.6
60	17	6	s-cv	56.7	1.1	2.5
60	18	5	hv	50.4	12.0	4.2
60	18	6	cv	67.5	0.9	5.2

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
60	18	4	s-cv	32.9	7.0	8.2
60	19	5	s-cv	62.5	6.5	0.4
60	19	4	s-cv	53.1	0.0	7.6
60	20	5	s-cv	59.4	8.9	1.9
60	20	6	cv	60.6	7.3	7.4
60	20	3	s-cv	4.2	0.0	0.0
60	21	5	hv	56.2	8.3	8.5
60	21	4	cv	59.0	5.1	0.6
60	22	5	cv	59.7	0.7	6.0
60	22	4	s-cv	33.6	0.0	2.5
60	23	5	s-hv	62.1	5.7	5.2
60	23	6	s-cv	57.7	2.7	4.6
60	24	5	hv	58.9	8.1	6.4
60	24	4	s-cv	39.2	10.6	0.8
60	25	5	hv	64.0	5.0	2.1
60	25	6	cv	67.1	4.9	9.3
60	25	1	s-cv	15.9	0.0	10.0
60	26	5	s-cv	66.1	3.9	5.7
60	26	4	s-cv	46.2	0.5	6.7
60	27	5	cv	42.0	1.8	4.1
60	27	6	cv	73.2	8.8	8.9
60	27	4	s-cv	28.2	0.0	0.0
60	28	5	s-cv	59.1	1.4	5.3
60	28	6	s-cv	69.2	2.9	0.7
60	28	1	s-cv	21.6	0.0	1.4
60	29	5	cv	57.1	3.0	8.4
60	29	6	cv	71.9	6.6	13.4
60	29	3	s-cv	31.1	0.1	0.0
70	0	5	hv	59.6	12.4	2.3
70	0	6	cv	74.3	2.3	1.9
70	0	3	s-hv	59.9	1.6	7.2
70	1	5	hv	67.9	7.3	0.2
70	1	4	hv	66.3	3.8	4.5
70	2	5	s-hv	63.7	8.2	7.0
70	2	6	cv	65.2	4.8	0.7
70	2	3	s-cv	24.9	0.0	0.0
70	3	5	cv	52.7	0.9	1.5
70	3	6	hv	74.4	6.4	5.5
70	3	3	s-cv	58.1	8.9	3.4
70	4	5	s-cv	64.8	7.0	1.9
70	4	6	s-cv	78.1	2.4	0.4
70	4	3	s-cv	80.7	3.8	17.0
70	5	5	cv	61.6	6.3	6.0
70	5	6	s-hv	78.2	5.3	4.2
70	5	4	s-cv	19.4	8.6	7.5
70	6	5	s-hv	51.3	6.9	5.2
70	6	6	hv	77.0	1.1	7.6
70	6	4	s-hv	50.6	3.1	1.0
70	6	1	s-cv	18.3	0.0	16.4
70	7	5	cv	60.0	2.5	4.0
70	7	6	s-cv	58.0	0.7	10.4
70	7	1	s-cv	23.2	4.2	16.0
70	8	5	cv	59.4	5.0	8.6
70	8	6	s-hv	75.0	7.8	7.4
70	8	4	s-cv	38.0	2.0	4.6
70	9	5	s-hv	61.9	9.0	9.0
70	9	6	cv	68.2	0.6	0.0
70	9	4	s-cv	49.7	8.0	7.7
70	10	5	s-cv	58.6	1.9	8.2
70	10	6	r	74.2	2.2	6.1
70	10	4	s-cv	44.7	1.1	0.0
70	11	5	hv	61.8	14.9	1.6
70	11	6	hv	72.7	6.9	2.8
70	11	3	s-cv	42.3	0.1	0.0
70	12	5	hv	62.7	0.7	6.7
70	12	6	hv	72.2	7.5	6.0
70	12	4	s-cv	32.0	3.7	0.0
70	13	5	hv	56.1	8.5	4.2
70	13	6	s-hv	71.4	6.1	4.2
70	13	4	s-cv	58.7	7.2	0.6
70	14	5	cv	57.8	1.7	4.2
70	14	6	hv	66.2	6.8	9.7
70	15	5	s-cv	66.1	1.5	4.1
70	15	6	hv	73.2	7.7	5.1
70	15	4	s-cv	40.1	4.2	0.2
70	16	5	hv	64.0	1.7	8.2
70	16	6	s-cv	66.9	2.0	2.2
70	17	5	cv	60.3	0.3	1.7
70	17	6	s-hv	78.4	8.5	6.8
70	17	4	s-cv	39.5	0.0	0.6
70	18	5	s-hv	65.5	0.8	0.1
70	18	6	s-cv	61.5	3.5	3.0
70	19	5	cv	55.9	1.0	8.9
70	19	6	s-hv	75.5	7.6	8.6
70	19	3	s-cv	40.1	0.1	17.4
70	20	5	s-cv	72.5	6.4	3.7
70	20	6	s-cv	76.1	1.1	6.6
70	20	4	s-cv	18.9	5.1	0.0
70	21	5	s-hv	65.7	1.2	4.5
70	21	6	s-hv	59.1	0.9	6.8
70	21	1	s-cv	20.7	0.0	1.4
70	22	5	cv	56.9	6.6	8.7

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
70	22	6	s-hv	77.0	7.0	2.7
70	22	4	s-cv	46.5	0.6	9.3
70	23	5	hv	66.6	7.8	2.5
70	23	6	hv	75.3	7.3	7.3
70	23	4	s-cv	25.3	14.9	7.9
70	24	5	hv	67.7	7.2	4.1
70	24	4	s-cv	32.8	1.1	0.0
70	25	5	s-cv	67.3	2.2	5.5
70	25	4	s-cv	59.1	8.2	4.0
70	26	5	s-cv	50.4	7.8	4.7
70	26	6	s-cv	74.8	2.8	6.0
70	26	4	s-cv	46.0	0.0	9.0
70	26	3	s-cv	29.1	0.0	0.0
70	27	5	s-hv	62.1	5.2	6.3
70	27	6	s-hv	66.0	6.5	6.3
70	27	3	s-cv	18.0	3.0	0.0
70	28	5	s-hv	65.6	4.5	7.5
70	28	6	hv	78.2	0.9	2.9
70	28	4	s-cv	48.8	0.0	0.0
70	29	5	hv	60.2	8.6	2.0
70	29	4	s-cv	46.6	2.2	1.6
80	0	5	s-cv	70.7	2.3	2.8
80	0	6	hv	72.5	5.0	2.8
80	0	3	s-cv	7.1	0.0	0.0
80	1	5	s-hv	69.8	7.6	2.3
80	1	6	hv	83.4	2.1	7.2
80	1	4	s-hv	61.9	7.7	4.1
80	2	5	s-cv	66.1	1.6	1.1
80	2	6	hv	69.7	4.6	6.3
80	2	3	s-cv	26.6	0.1	0.0
80	3	5	s-cv	62.4	0.8	3.3
80	3	6	s-hv	79.6	4.0	3.4
80	3	4	s-cv	47.7	8.6	8.9
80	4	5	cv	66.1	8.2	6.7
80	4	6	s-hv	76.6	8.2	0.7
80	4	4	s-cv	39.9	4.5	6.8
80	5	5	cv	63.1	1.0	2.7
80	5	6	s-hv	82.1	6.6	5.1
80	5	4	r	62.2	3.2	8.6
80	5	3	s-cv	62.4	7.3	5.0
80	6	5	s-hv	68.1	4.3	7.1
80	6	6	s-cv	73.4	9.7	0.7
80	6	4	s-cv	52.5	9.5	8.9
80	7	5	s-cv	61.8	9.9	2.8
80	7	6	cv	53.2	2.2	6.0
80	7	4	r	61.7	7.4	7.8
80	7	2	s-cv	56.5	0.0	5.0
80	7	3	s-cv	30.6	0.0	0.0
80	8	5	s-hv	69.6	4.6	10.4
80	8	6	r	76.6	1.6	1.0
80	8	4	s-cv	25.2	0.0	8.3
80	9	5	hv	64.2	9.5	5.3
80	9	6	hv	69.8	2.0	6.1
80	9	4	s-cv	37.1	0.0	0.8
80	10	5	s-cv	61.2	7.0	4.4
80	10	6	s-cv	75.7	3.5	5.9
80	10	4	s-cv	32.0	7.5	2.9
80	11	5	s-hv	68.5	8.2	3.0
80	11	6	s-cv	77.5	2.4	0.1
80	11	4	cv	49.8	7.1	7.4
80	12	5	hv	61.3	6.7	2.2
80	12	6	hv	71.4	2.4	6.0
80	12	3	s-cv	7.1	0.0	0.0
80	13	5	cv	65.4	1.4	4.7
80	13	6	s-hv	74.5	10.0	0.7
80	13	3	s-cv	44.0	0.0	0.0
80	14	5	s-hv	70.2	6.4	5.5
80	14	6	s-cv	61.3	4.6	8.9
80	15	5	hv	65.7	1.8	1.7
80	15	6	cv	73.5	3.4	4.4
80	15	4	s-cv	44.5	0.0	1.9
80	16	5	cv	48.3	4.4	4.3
80	16	6	s-hv	80.2	9.1	6.2
80	16	4	s-cv	41.2	0.9	2.8
80	17	5	hv	51.7	1.9	3.6
80	17	6	s-cv	81.0	3.4	5.0
80	17	4	s-cv	30.8	4.3	0.0
80	18	5	s-hv	67.4	1.0	0.7
80	18	6	s-cv	74.9	0.6	6.6
80	18	3	s-cv	49.4	0.0	0.0
80	19	5	s-cv	75.2	0.9	4.6
80	19	6	cv	78.1	7.9	0.1
80	19	4	s-cv	33.5	0.1	6.3
80	20	5	s-hv	65.0	3.2	1.6
80	20	6	s-hv	75.4	9.9	8.6
80	20	1	s-cv	11.1	0.0	1.4
80	21	5	cv	48.7	9.5	2.8
80	21	6	s-cv	79.1	2.5	5.0
80	21	4	hv	67.4	7.6	3.5
80	21	3	s-cv	43.8	0.0	0.0
80	22	5	cv	58.3	8.8	9.6
80	22	6	hv	74.5	6.2	0.3

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
80	22	4	s-cv	60.0	3.6	4.1
80	22	1	s-cv	18.3	0.0	1.4
80	23	5	s-cv	62.4	11.1	2.3
80	23	6	s-hv	79.1	5.8	4.7
80	23	4	s-cv	55.7	0.9	2.8
80	24	5	hv	60.2	0.2	0.3
80	24	6	r	74.3	1.3	1.8
80	24	1	s-cv	21.8	0.0	1.4
80	25	5	s-cv	62.6	7.5	5.3
80	25	6	s-cv	68.7	4.3	2.7
80	25	4	s-cv	26.1	2.5	0.0
80	26	5	s-cv	70.9	3.3	2.9
80	26	6	s-hv	64.5	1.3	0.8
80	26	3	s-cv	24.6	0.1	0.0
80	27	5	s-cv	58.9	5.7	8.7
80	27	6	s-cv	73.5	9.4	5.2
80	27	4	s-cv	38.0	25.5	0.0
80	28	5	hv	63.9	7.1	1.4
80	28	6	s-hv	76.8	1.8	0.7
80	28	4	s-cv	49.1	5.3	7.0
80	28	3	s-cv	13.8	0.0	0.0
80	29	5	cv	56.2	3.0	1.1
80	29	6	s-hv	78.1	1.1	10.2
80	29	4	cv	64.7	3.9	3.3
80	29	2	s-cv	55.4	7.5	19.6
80	29	3	s-cv	42.8	0.0	5.7
90	0	5	cv	63.4	2.9	3.0
90	0	6	s-hv	76.8	2.6	2.5
90	0	4	s-cv	43.8	8.5	1.6
90	1	5	hv	66.6	8.7	4.3
90	1	6	cv	75.3	7.1	8.8
90	1	4	s-cv	44.5	0.5	5.5
90	2	5	cv	75.6	5.7	3.8
90	2	6	s-hv	82.8	2.9	9.2
90	2	4	s-hv	59.3	3.7	1.6
90	3	5	hv	57.9	4.2	1.7
90	3	6	r	77.8	6.7	8.7
90	3	4	s-cv	37.2	0.0	6.6
90	4	5	hv	54.2	19.3	0.1
90	4	6	s-hv	79.7	0.9	3.0
90	4	4	s-cv	65.7	0.0	1.5
90	4	2	cv	38.4	0.1	3.2
90	4	3	s-cv	42.5	0.0	0.0
90	5	5	cv	61.4	4.4	0.8
90	5	6	hv	78.2	6.6	6.7
90	5	4	s-cv	30.2	0.0	0.0
90	6	5	cv	54.2	6.3	3.7
90	6	6	s-hv	79.5	9.4	6.3
90	6	4	hv	63.1	6.9	3.2
90	6	3	s-cv	55.4	3.3	5.8
90	7	5	cv	70.8	3.0	7.9
90	7	6	s-hv	81.5	2.5	9.1
90	7	4	s-cv	40.6	7.2	4.6
90	8	5	hv	68.4	3.2	6.3
90	8	6	cv	74.7	5.7	8.7
90	8	3	s-cv	36.0	16.5	5.6
90	9	5	s-cv	60.5	1.8	8.6
90	9	6	cv	72.8	2.4	7.4
90	9	4	cv	64.8	6.8	1.0
90	9	1	s-cv	20.5	0.0	1.4
90	10	5	hv	53.6	8.3	10.6
90	10	6	cv	79.2	9.3	7.1
90	10	4	cv	71.2	8.3	4.7
90	10	2	s-cv	45.1	0.0	3.6
90	10	3	s-cv	45.8	8.7	18.6
90	11	5	s-cv	66.4	0.3	0.2
90	11	6	s-cv	76.8	5.8	6.2
90	11	4	s-cv	32.2	0.0	5.0
90	12	5	s-cv	74.3	9.1	2.8
90	12	6	s-cv	67.4	0.1	3.2
90	12	3	s-cv	21.3	0.0	2.9
90	13	5	s-hv	66.1	5.0	1.3
90	13	6	s-cv	72.0	5.1	2.0
90	13	1	s-cv	28.5	0.9	10.0
90	14	5	cv	65.6	8.5	6.2
90	14	6	cv	66.6	1.7	6.6
90	14	4	s-cv	66.4	1.5	0.1
90	14	2	s-cv	45.6	19.3	3.6
90	14	1	s-cv	21.2	0.0	10.8
90	15	5	s-hv	72.1	3.7	2.8
90	15	6	s-hv	75.6	5.5	9.5
90	15	4	s-cv	35.7	0.2	3.6
90	16	5	s-hv	68.9	6.7	2.8
90	16	6	s-hv	76.2	7.2	9.5
90	16	4	s-cv	32.5	14.0	7.7
90	17	5	cv	69.1	8.1	2.9
90	17	6	s-cv	82.2	6.0	4.1
90	17	4	s-cv	63.5	5.3	8.9
90	17	1	s-cv	23.2	4.2	16.0
90	18	5	hv	61.3	0.5	0.4
90	18	6	r	71.5	0.8	4.3
90	18	4	s-cv	54.9	9.1	7.6

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
90	18	3	s-cv	56.5	0.1	0.6
90	19	5	s-hv	67.1	9.2	8.0
90	19	6	s-cv	74.4	8.8	0.0
90	19	4	s-cv	47.9	1.0	1.1
90	20	5	s-cv	64.6	1.9	2.5
90	20	6	hv	77.1	8.4	3.0
90	20	4	s-hv	61.5	8.1	3.5
90	20	3	s-cv	53.6	0.0	4.9
90	21	5	s-cv	67.4	6.9	2.4
90	21	6	hv	78.4	1.2	8.2
90	21	4	s-hv	70.9	7.8	5.7
90	21	2	s-cv	34.1	0.0	0.0
90	21	3	s-cv	27.8	0.0	0.0
90	22	5	s-cv	66.5	4.7	4.1
90	22	6	s-cv	81.3	1.8	8.7
90	22	4	s-cv	34.6	0.0	2.0
90	23	5	cv	64.1	8.8	4.6
90	23	6	s-cv	79.3	5.7	7.6
90	23	4	s-cv	40.9	5.1	0.0
90	24	5	s-cv	55.7	12.5	7.3
90	24	6	hv	75.5	0.0	7.3
90	24	4	r	66.7	3.4	4.9
90	24	3	s-cv	33.9	0.1	0.0
90	25	5	s-cv	64.7	4.8	7.6
90	25	6	cv	78.1	1.7	4.4
90	25	4	s-cv	20.9	0.0	0.0
90	26	5	s-hv	62.5	2.6	3.0
90	26	6	cv	74.4	5.2	3.6
90	26	4	s-hv	51.0	8.8	0.1
90	26	3	s-cv	7.1	0.0	0.0
90	27	5	hv	68.0	6.8	6.8
90	27	6	s-hv	73.6	2.9	0.3
90	27	4	s-hv	61.1	9.3	2.0
90	27	1	s-cv	22.5	0.0	10.0
90	28	5	s-cv	63.9	8.7	7.0
90	28	6	s-cv	80.1	9.4	8.3
90	28	4	hv	64.5	0.5	5.3
90	28	1	s-cv	21.2	0.0	10.8
90	29	5	s-cv	49.0	9.8	4.8
90	29	6	s-cv	83.4	5.5	3.4
90	29	4	hv	67.0	0.7	8.2
90	29	2	s-cv	36.3	9.1	2.3
90	29	3	s-cv	31.1	0.1	0.0
100	0	5	hv	66.2	4.6	4.7
100	0	6	s-hv	65.3	5.8	9.1
100	0	3	s-cv	24.0	0.1	13.7
100	1	5	s-cv	69.3	3.4	7.9
100	1	6	s-cv	75.9	7.0	3.9
100	1	4	cv	65.7	3.8	10.7
100	1	2	s-cv	36.3	9.1	2.3
100	1	3	s-cv	11.5	0.1	0.0
100	2	5	hv	64.3	6.3	2.0
100	2	6	s-cv	81.2	7.4	4.9
100	2	4	s-cv	63.2	3.5	0.1
100	2	2	s-cv	45.4	13.7	7.3
100	2	3	s-cv	28.5	0.0	0.0
100	3	5	s-cv	59.3	0.4	6.0
100	3	2	cv	58.6	0.0	5.6
100	3	6	hv	82.2	9.5	3.9
100	3	4	s-cv	61.4	7.9	4.9
100	3	3	s-cv	41.9	0.1	0.0
100	4	5	s-hv	61.6	7.2	8.3
100	4	6	s-cv	80.6	0.1	7.4
100	4	4	hv	65.9	3.5	3.7
100	4	3	s-cv	43.7	3.4	0.0
100	5	5	s-hv	60.0	5.5	0.7
100	5	6	cv	79.8	4.8	4.2
100	5	4	s-cv	47.9	2.5	3.0
100	6	5	hv	76.6	0.2	1.7
100	6	6	s-cv	74.0	0.0	1.6
100	6	4	s-cv	76.3	3.6	7.2
100	6	2	hv	52.8	0.0	2.2
100	6	3	s-cv	48.7	9.3	6.8
100	6	1	s-cv	24.3	0.0	10.0
100	7	5	s-cv	70.0	6.3	6.3
100	7	6	hv	71.4	2.4	6.9
100	7	4	s-cv	27.6	1.5	1.1
100	8	5	s-cv	57.4	1.5	1.8
100	8	6	cv	81.6	1.2	7.7
100	8	4	s-cv	69.0	4.3	9.1
100	8	3	s-cv	28.9	3.5	8.0
100	9	5	cv	70.4	7.0	3.8
100	9	6	s-cv	78.5	4.1	1.2
100	9	4	s-hv	66.7	13.8	5.1
100	9	3	s-cv	54.9	19.6	5.4
100	10	5	hv	70.9	1.7	7.8
100	10	6	hv	80.9	6.4	14.2
100	10	4	s-cv	62.5	7.1	9.2
100	10	3	s-cv	27.0	9.9	0.0
100	11	5	s-hv	52.9	9.8	2.4
100	11	6	cv	64.0	10.4	8.3
100	11	4	cv	67.3	1.5	5.4

Continued on next column

$\|I\|$	No.	ID	$\mathcal{C}$	u	cog_1^{dev}	cog_2^{dev}
100	11	2	s-cv	50.9	2.1	0.0
100	11	3	s-cv	45.7	0.1	0.0
100	12	5	hv	58.2	9.0	7.9
100	12	6	s-cv	76.7	2.3	6.9
100	12	4	s-hv	70.7	1.4	9.1
100	12	2	s-cv	62.4	8.5	4.8
100	12	3	s-hv	71.2	4.1	2.2
100	12	1	s-cv	26.5	0.0	10.0
100	13	5	s-hv	68.8	2.5	4.4
100	13	6	hv	76.0	7.8	5.0
100	13	4	s-cv	46.6	1.9	2.9
100	14	5	hv	60.6	6.8	3.2
100	14	6	s-hv	62.6	0.0	2.2
100	14	4	s-hv	71.6	7.3	9.5
100	14	2	s-cv	48.9	8.8	13.4
100	14	1	s-cv	18.6	0.0	10.8
100	15	5	s-hv	60.0	4.5	1.3
100	15	6	hv	78.8	5.0	5.9
100	15	4	hv	67.1	0.8	6.4
100	15	2	s-cv	44.8	13.3	4.6
100	15	3	s-cv	27.9	0.0	0.0
100	16	5	cv	64.2	9.3	7.3
100	16	6	s-hv	75.3	3.5	10.8
100	16	4	cv	68.5	9.7	0.7
100	16	2	s-cv	38.7	0.1	1.7
100	16	3	s-cv	29.5	0.0	0.0
100	17	5	s-hv	73.8	6.2	7.8
100	17	6	r	76.8	8.4	1.0
100	17	4	s-cv	50.3	0.0	7.6
100	18	5	cv	66.7	8.1	0.1
100	18	6	hv	84.5	0.6	4.0
100	18	4	s-cv	31.6	7.4	3.9
100	19	5	s-hv	61.2	2.0	3.8
100	19	6	s-hv	74.4	5.7	2.4
100	19	4	cv	60.0	4.0	5.5
100	19	3	s-cv	43.6	0.0	0.0
100	20	5	s-cv	63.2	7.8	0.2
100	20	6	hv	78.4	6.2	0.6
100	20	4	s-hv	65.2	5.1	11.0
100	20	3	s-cv	27.9	0.0	0.0
100	21	5	cv	57.6	27.5	14.9
100	21	6	s-cv	78.8	2.6	8.5
100	21	4	cv	72.1	1.5	2.6
100	21	2	s-cv	54.7	0.0	8.4
100	21	3	cv	50.7	0.1	8.8
100	21	1	s-cv	18.6	0.0	10.8
100	22	5	hv	59.4	9.2	2.3
100	22	6	s-cv	79.2	6.8	7.5
100	22	4	cv	73.3	4.0	7.1
100	22	2	s-cv	59.7	4.9	4.3
100	22	3	s-cv	47.8	0.1	3.6
100	23	5	s-cv	71.6	4.1	6.7
100	23	6	cv	79.0	7.2	1.1
100	23	4	hv	67.4	8.0	4.4
100	24	5	s-hv	68.8	7.8	8.4
100	24	6	hv	78.3	6.8	1.2
100	24	4	s-cv	70.7	7.8	4.8
100	24	2	s-cv	33.1	0.1	7.7
100	24	1	s-cv	18.3	0.0	1.4
100	25	5	cv	64.9	7.6	0.0
100	25	6	hv	77.3	5.0	5.3
100	25	4	hv	66.4	4.8	8.5
100	25	2	s-cv	40.3	0.0	9.9
100	25	3	s-cv	43.9	0.0	2.1
100	26	5	s-cv	51.2	3.4	6.2
100	26	6	s-hv	77.4	5.9	0.8
100	26	4	s-cv	71.5	4.4	6.9
100	26	2	s-cv	59.2	5.1	4.7
100	26	3	cv	44.7	0.1	0.5
100	26	1	s-cv	24.3	0.0	16.0
100	27	5	hv	67.6	2.4	0.3
100	27	6	hv	81.1	0.2	4.7
100	27	4	hv	66.8	6.1	1.8
100	27	1	s-cv	21.8	0.0	1.4
100	28	5	s-cv	68.4	2.8	1.7
100	28	6	hv	82.6	2.9	0.7
100	28	4	s-cv	64.9	8.2	9.1
100	28	2	s-cv	28.7	4.1	0.0
100	28	1	s-cv	24.3	0.0	16.0
100	29	5	hv	51.9	22.7	5.3
100	29	6	hv	61.4	3.7	0.0
100	29	4	r	74.7	5.1	0.6
100	29	2	r	50.2	0.1	5.5
100	29	3	s-hv	51.6	5.8	8.2
100	29	1	s-cv	21.8	0.0	1.4