1038 A10s Fit into One A0

Noel Friedrich
Department of Mathematics, ETH Zürich, Switzerland

Abstract

The A-series paper sizes are specified in integer millimetres in ISO 216. Since the sizes are based on an irrational aspect ratio, rounding errors introduce small but cumulative discrepancies from their nominal areas. In this work, we prove that exactly 1038 sheets of ISO A10 can be packed without overlap into a single ISO A0 sheet, allowing only orthogonal (axis-aligned) placements. A lower bound is given by an explicit construction, while the upper bound is proved by a simple-to-verify computer-assisted certificate. Along the way, the dual certificates produce unexpectedly pretty weightmaps.[1]

2012 ACM Subject Classification Theory of computation → Packing and covering problems

Keywords and phrases rectangle packing, pallet loading problem, rounding effects, ISO 216

Digital Object Identifier 10.4230/LIPIcs.FUN.2026.18

Supplementary Material *Software (Verification scripts, rounded weightmap certificate, and packing coordinates)*: `https://doi.org/10.5281/zenodo.18813570`

1 Introduction

There are many reasons why the A-series has become the dominant international system for paper sizes [17]: it is metrically grounded, self-similar and built around the elegant aspect ratio $1 : \sqrt{2}$ [12]. Yet, ISO 216 [6] (and its predecessor DIN 476 [11]) specify the actual paper sizes in whole millimetres. Since the nominal side ratio is irrational, this discretization introduces a rounding error. In the standard construction, A0 is approximated first, and each subsequent size is obtained by halving the previous one and rounding down so that two sheets of $A(n)$ can always be cut from one sheet of $A(n-1)$ [14]. In this work we quantify the cumulative effect of these roundings by studying a concrete packing problem: how many A10 sheets can be placed, without overlap and with axis-aligned orientation, inside a single sheet of A0?

To formalise this, let us fix integers $(L, W) := (1189, 841)$ and $(a, b) := (37, 26)$ to be the sizes of the A0 and A10 rectangles in millimetre units. We further define an integer coordinate system where the top-left corner represents $(0, 0)$ and the bottom-right corner of the larger A0-sized rectangle represents (L, W).

▶ **Definition 1** (Orthogonal placement and packing). *An* orthogonal placement *is a rectangle* $R = [x, x+w] \times [y, y+h] \subseteq [0, L] \times [0, W]$ *where* $(w, h) \in \{(a, b), (b, a)\}$ *and* $x, y \in \mathbb{R}$ *(or* $x, y \in \mathbb{Z}$ *in the integer-mm model).*

A set $\mathcal{P}$ *of orthogonal placements is a* non-overlapping orthogonal packing *if for any two distinct rectangles* $R, R' \in \mathcal{P}$*, their interiors are disjoint, i.e.,* $\operatorname{int}(R) \cap \operatorname{int}(R') = \emptyset$ *(Equivalently, rectangles may touch at boundaries but may not overlap in area).*

▶ **Definition 2** (Optimal packing number). *Let* $\mathrm{OPT}(L, W, a, b)$ *be the maximum size of a non-overlapping orthogonal packing of* $(a \times b)$*-rectangles into an* $(L \times W)$*-rectangle. In this article, we study* $\mathrm{OPT} := \mathrm{OPT}(1189, 841, 37, 26)$.

[1] Companion video (expository): `https://youtu.be/zDKBCIMkDbw` [10]

Without much work, we can show some lower and upper bounds for OPT.

▶ **Proposition 3** (Trivial bounds). $\mathrm{OPT} \in \{1024, \ldots, 1039\} \subseteq \mathbb{N}$.

Proof. The lower bound of $1024 = 2^{10}$ holds because the paper sizes are designed such that every sheet of $A(n)$ can fit (at least) two sheets of $A(n+1)$ for $0 \leq n \leq 9$ [6]. The upper bound holds by area: $\mathrm{OPT} \leq \left\lfloor \frac{\text{Area of A0}}{\text{Area of A10}} \right\rfloor = \left\lfloor \frac{1189 \times 841}{37 \times 26} \right\rfloor = 1039$. ◀

Our primary goal in this article is proving the exact value of OPT, namely:

▶ **Theorem 4** (Main Theorem, computer-assisted). $\mathrm{OPT} = 1038$.

The proof of Theorem 4 will be done in three steps. We will first show that we can assume the integer-mm model without loss of generality. Then we will provide an explicit packing of size 1038, hence proving a lower bound. Finally, we will show an upper bound of 1038 using a computer-assisted argument, concluding the proof.

▶ **Lemma 5** (Integer-coordinates without loss of generality). *Consider an* orthogonal packing *of axis-aligned rectangles of sizes* $(a \times b)$ *and* $(b \times a)$ *inside the container* $[0, L] \times [0, W]$*, where rectangle positions may use arbitrary real coordinates.*

Then there exists an orthogonal packing of the same cardinality *in which every rectangle has integer coordinates, i.e., each rectangle is of the form* $[x, x+w] \times [y, y+h]$ *with* $x, y \in \mathbb{Z}$ *and* $(w, h) \in \{(a, b), (b, a)\}$.

Proof. Consider an arbitrary *orthogonal packing* $\mathcal{P} = \{R_1, \ldots, R_n\}$ of size n. We first make this packing *left-stable* by repeatedly selecting any rectangle $R \in \mathcal{P}$ and translating it leftwards, i.e., decreasing its x-coordinate $x(R)$, as far as possible while keeping the packing *non-overlapping* and $x(R) \geq 0$. This process terminates at a packing in which every rectangle is either touching the boundary ($x(R) = 0$) or its left side touches the right side of another rectangle.

The resulting *left-stable* packing is still *non-overlapping* and has the same size n. Furthermore, every rectangle must either have an x-coordinate of $x(R) = 0$ (on the boundary) or $x(R) = x(R') + w(R')$ where R' is a rectangle blocking leftwards movements and $w(R')$ is the width of said rectangle. Since we can form a chain of blocking rectangles from every rectangle to the line $x = 0$ and since $w(R) \in \mathbb{Z}$ for all rectangles $R \in \mathcal{P}$, all $x(R)$ must be integer.

We can apply an analogous argument for producing a *top-stable* packing that will yield integer y-coordinates. Therefore, there exists a feasible packing of the same size whose rectangle coordinates are integers. ◀

2 Lower Bound

▶ **Theorem 6** (Lower bound). $\mathrm{OPT} \geq 1038$.

We will prove this by splitting the $L \times W$ area into two parts. **Part A** is 962×841, **Part B** is 227×841. Combined, these two parts exactly cover the entire $(962 + 227) \times 841 = L \times W$ area. An explicit construction for both parts is depicted in Figure 1.

▶ **Lemma 7** (**Part A**). *A* (962×841) *rectangle can be packed gaplessly with a* non-overlapping orthogonal packing *of size* 841*, i.e.,* $\mathrm{OPT}(962, W, a, b) = 841$.

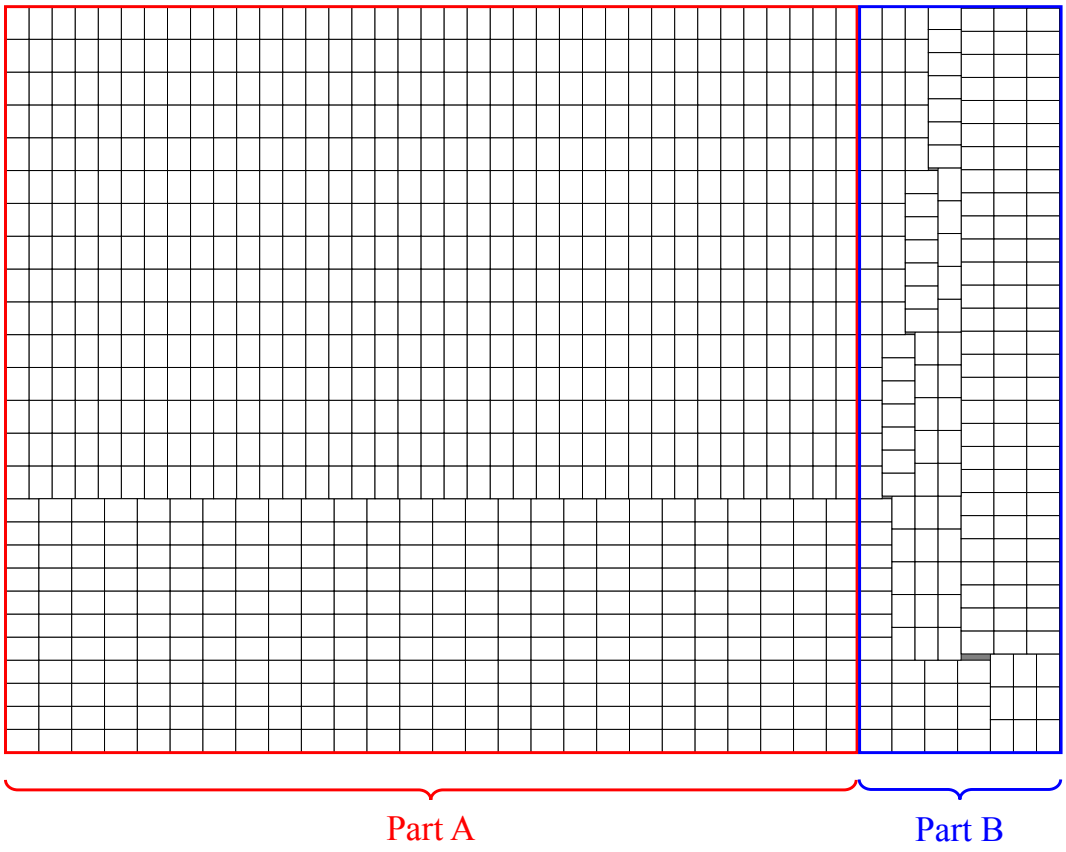

Figure 1 An arrangement for 1038 A10 in one A0, constructed on a custom website (FORP [7]).

Proof. Since $15 \cdot 37 + 11 \cdot 26 = 841$, we tile the height by 15 strips of height 37 and 11 strips of height 26. Each 37-high strip is tiled by 37 upright (26×37) rectangles as $37 \cdot 26 = 962$. Each 26-high strip is tiled by 26 horizontal (37×26) rectangles as $26 \cdot 37 = 962$.

Thus the (962×841) rectangle is tiled gaplessly by $15 \cdot 37 + 11 \cdot 26 = 841$ rectangles, implying $\mathrm{OPT}(962, 841, a, b) \geq 841$. Equality follows directly from the lack of gaps. ◀

▶ **Lemma 8 (Part B).** *A* (227×841) *rectangle can be packed with a* non-overlapping orthogonal packing *of size* 197*, i.e.,* $\mathrm{OPT}(227, W, a, b) \geq 197$.

Proof. Appendix A lists 197 rectangles by integer coordinates (x_0, y_0, x_1, y_1), representing $R = [x_0, x_1] \times [y_0, y_1]$. It suffices to check:

- **(i) Correct dimensions:** for every listed rectangle, $(x_1 - x_0, y_1 - y_0) \in \{(26, 37), (37, 26)\}$.
- **(ii) Containment:** for every listed rectangle, $0 \leq x_0 < x_1 \leq 227$ and $0 \leq y_0 < y_1 \leq 841$.
- **(iii) Non-overlap:** the rectangles are pairwise interior-disjoint. A *fun* way to verify this (using integrality of the coordinates) is to count covered unit squares: if we mark every covered 1×1 cell in the 227×841 grid, then disjointness is equivalent to the number of covered cells being exactly the sum of areas, i.e., $197 \cdot 37 \cdot 26$.

Therefore the listed rectangles form a non-overlapping orthogonal packing of size 197 in the (227×841) container, proving $\mathrm{OPT}(227, 841, a, b) \geq 197$. The appendix also contains a short script for verifying these claims mechanically. Combined with Lemma 7, this proves Theorem 6. ◀

This specific packing was constructed and exported using a custom website featuring an interactive visualization and playground we developed for this problem appropriately called **F**un **O**rthogonal **R**ectangle **P**acking [7]. The presented arrangements were constructed by hand and by using an existing algorithm developed by Birgin, Lobato and Morabito [4] for smaller subproblems.

3 Upper Bound

Existing techniques for finding upper bounds use a structural approach [15] on a Minimum Size Instance (MSI) [16]. The MSI is a minimal instance preserving the same set of optimal packing arrangements. In this setting, our packing problem can be formulated as the $(L, W, a, b) = (1189, 841, 37, 26)$ instance of the Pallet Loading Problem (PLP).

Our instance is already the MSI, since A0 can be tiled gaplessly by A10s in both width and height: $23 \cdot 37 + 13 \cdot 26 = 1189$ and $15 \cdot 37 + 11 \cdot 26 = 841$. The Barnes bound [3] for this instance also yields 1039. Together, both of these facts indicate that this instance does not permit an easy simplification. Therefore, we need to employ more complex computational bounds.

3.1 Structural LP Bounds (that almost work)

Isermann [13] and Nelißen [18] show how we can compute upper bounds by considering $a \times 1$ and $b \times 1$ strips combined with structural information about all possible feasible partitions of the side lengths, expressed by the following linear program:

$$F(L, a, b) := \{(i, j) \in \mathbb{Z}^2_{\geq 0} \mid ia + jb \leq L,\ (i, j) \neq (0, 0)\},$$

$$F(W, a, b) := \{(f, g) \in \mathbb{Z}^2_{\geq 0} \mid fa + gb \leq W,\ (f, g) \neq (0, 0)\}.$$

$$\begin{aligned}
\max \quad & \sum_{(i,j)\in F(L,a,b)} \frac{i}{b}\, x_{ij} + \sum_{(f,g)\in F(W,a,b)} \frac{f}{b}\, y_{fg} \\
\text{s.t.} \quad & \sum_{(i,j)\in F(L,a,b)} x_{ij} \leq W, \qquad \sum_{(f,g)\in F(W,a,b)} y_{fg} \leq L, \\
& a \sum_{(i,j)\in F(L,a,b)} i\, x_{ij} - b \sum_{(f,g)\in F(W,a,b)} g\, y_{fg} = 0, \\
& b \sum_{(i,j)\in F(L,a,b)} j\, x_{ij} - a \sum_{(f,g)\in F(W,a,b)} f\, y_{fg} = 0, \\
& x_{ij} \geq 0 \quad \forall (i, j) \in F(L, a, b), \qquad y_{fg} \geq 0 \quad \forall (f, g) \in F(W, a, b).
\end{aligned}$$

We formulated and ran this linear program using the Gurobi LP solver, which computed an optimum of `1039.0254329004`. This structural bound does not disprove 1039, but is close enough to yield hope that some clever tweaks might drop this bound below 1039. Unfortunately, we did not manage to improve this bound, even when employing more advanced constraints suggested by Nelißen [18] and Letchford [15]. This supports the earlier claim that our specific instance of the PLP does not permit a *simple* proof of optimality.

3.2 Weighing Bounds (a.k.a. dual witnesses)

The aforementioned structural bounds can be expressed in a simpler way by reformulating them as weightmap-bounds. The idea is to find a weightmap $w(x, y)$ which maps every 1×1 square on the A0-sized lattice to a non-negative real value. If we construct a weightmap such that for every possible rectangle placement the sum over that rectangle is ≥ 1, the total sum over all weights is then an upper bound for the optimal packing size.

We formalize this by defining the set of all integer placements:

$$\begin{aligned} \mathcal{R}_{horizontal} &= \{(x, y, a, b) \mid 0 \leq x \leq L - a,\ 0 \leq y \leq W - b\}, \\ \mathcal{R}_{vertical} &= \{(x, y, b, a) \mid 0 \leq x \leq L - b,\ 0 \leq y \leq W - a\}, \\ \mathcal{R} &= \mathcal{R}_{horizontal} \cup \mathcal{R}_{vertical}. \end{aligned}$$

▶ **Lemma 9** (Weight-map upper bound). *Let* $w : \{0, \ldots, L-1\} \times \{0, \ldots, W-1\} \to \mathbb{R}_{\geq 0}$. *Assume that every placement* $(x_0, y_0, w, h) \in \mathcal{R}$ *satisfies*

$$\sum_{x=x_0}^{x_0+w-1} \sum_{y=y_0}^{y_0+h-1} w(x, y) \ \geq \ 1.$$

Then the size of any non-overlapping orthogonal packing is at most

$$\mathrm{OPT}(L, W, a, b) \leq T := \sum_{x=0}^{L-1} \sum_{y=0}^{W-1} w(x, y).$$

Proof. Let $\mathcal{P}$ be any non-overlapping orthogonal packing. By assumption, every rectangle in $\mathcal{P}$ covers a weight of at least 1. Since rectangles in $\mathcal{P}$ are interior-disjoint, the total weight covered by all rectangles is at most the total weight T of the container. Hence $|\mathcal{P}| \leq T$. ◀

▶ **Corollary 10** (Scaled integer certificate). *Let* $S \in \mathbb{Z}_{\geq 1}$ *and let* $w : \{0, \ldots, L-1\} \times \{0, \ldots, W-1\} \to \mathbb{Z}_{\geq 0}$. *If every placement in* $\mathcal{R}$ *has weight at least* S *and* $\sum_{x,y} w(x, y) < 1039 \cdot S$, *then* $\mathrm{OPT} \leq 1038$.

Proof. Apply Lemma 9 to the scaled weightmap w/S. ◀

The space of all weightmaps is quite large (there are approximately 1 million variable weights and approximately 2 million possible rectangle placements) and hence computationally expensive to optimize for directly. If we restrict the weightmaps to specific (easier to compute) forms, we can recover some structural bounds.

For example, restricting to weightmaps of the form $w(x, y) = a_x + b_y$ severely limits the space of free variables to only $L + W = 2030$. Optimizing over this restricted family yields the familiar value `1039.0254329004` (Figure 2), i.e., the same bound as the structural LP. We tried several more restricted weightmaps (periodic tilings, diagonal striping patterns, extra border variables and combinations), but all either produced a worse upper bound or collapsed to the striping pattern. This raised the fear that the optimal value of any weightmap approach might not lie below 1039, which would invalidate this proof approach.

We therefore solve the general weightmap LP. To do so efficiently, we use a prefix-sum reformulation that dramatically reduces the number of nonzero constraint coefficients. We introduce variables $F_{x,y} \geq 0$ for $x \in \{1, \ldots, L\}$ and $y \in \{1, \ldots, W\}$ with boundary convention $F_{0,y} = F_{x,0} = 0$, and solve:

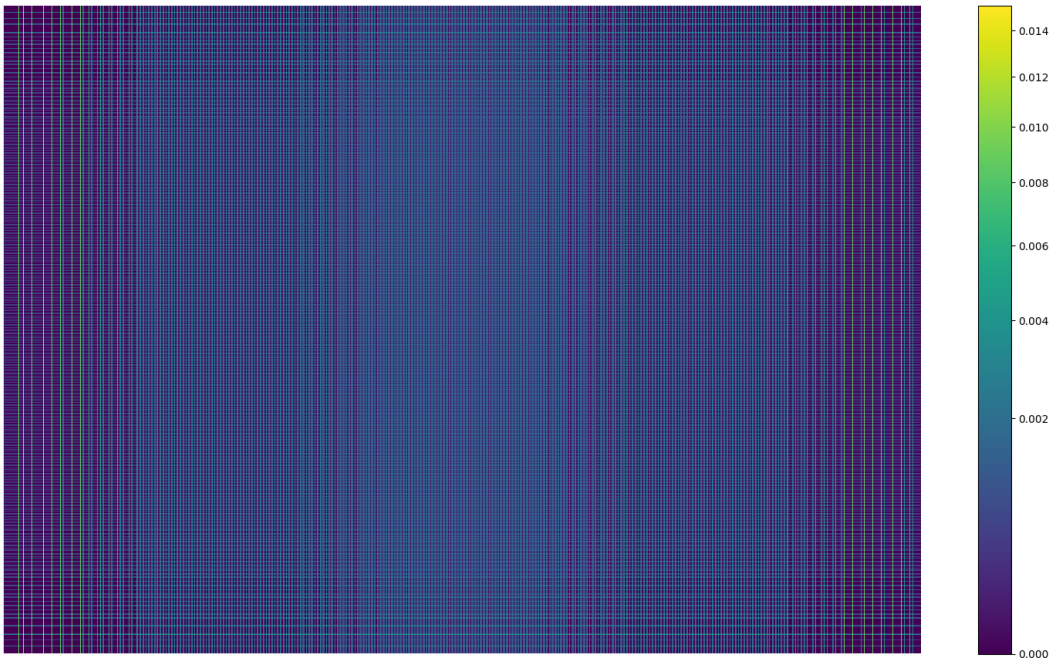

Figure 2 The optimal weightmap under restriction $w(x, y) = a_x + b_y$.

$$
\begin{aligned}
\min \quad & F_{L,W} \\
\text{s.t.} \quad & F_{x,y} - F_{x-1,y} - F_{x,y-1} + F_{x-1,y-1} \;\geq\; 0 \qquad \forall\, x \in \{1,\dots,L\}\ \forall\, y \in \{1,\dots,W\}, \\
& F_{x+w,\,y+h} - F_{x,\,y+h} - F_{x+w,\,y} + F_{x,\,y} \;\geq\; 1 \qquad \forall\, (x,y,w,h) \in \mathcal{R}, \\
& F_{x,y} \geq 0 \qquad \forall\, x \in \{1,\dots,L\}\ \forall\, y \in \{1,\dots,W\}.
\end{aligned}
$$

This reformulation uses prefix sums over the weights instead of variables for each weight; the individual weights can be reconstructed trivially.

We computed the optimum of this LP using Gurobi (PDHG method). We ran the instance on the Euler Cluster [5]. The computed optimum value is `1038.7291940779483`. Since this is an upper bound, this indicates that an orthogonal packing of size 1038 is optimal, but needs to be verified in exact arithmetic to avoid floating-point inaccuracies.

▶ **Theorem 11** (Upper bound, computer-assisted)**.** *There exists an integer weightmap* $w : \{0,\dots,L-1\} \times \{0,\dots,W-1\} \to \mathbb{Z}_{\geq 0}$ *with scale* $S = 10^6$ *such that:*

(i) *every placement in* $\mathcal{R}$ *has total weight at least* S*,*

(ii) $\sum_{x,y} w(x,y) < 1039 \cdot S$*.*

Consequently, OPT ≤ 1038*.*

Proof. We take an optimal floating-point-valued weightmap from the LP above, multiply it by 10^6 and round to integers. Using numerically exact integer addition, we computed a minimum rectangle-sum of $\Sigma_{\min} = 1{,}000{,}000$ and a total sum of $\Sigma_{\text{total}} = 1{,}038{,}768{,}234$. Thus condition (i) holds (since $\Sigma_{\min} \geq S$) and condition (ii) holds (since $\Sigma_{\text{total}} < 1039S$). The claim OPT ≤ 1038 follows from Corollary 10. ◀

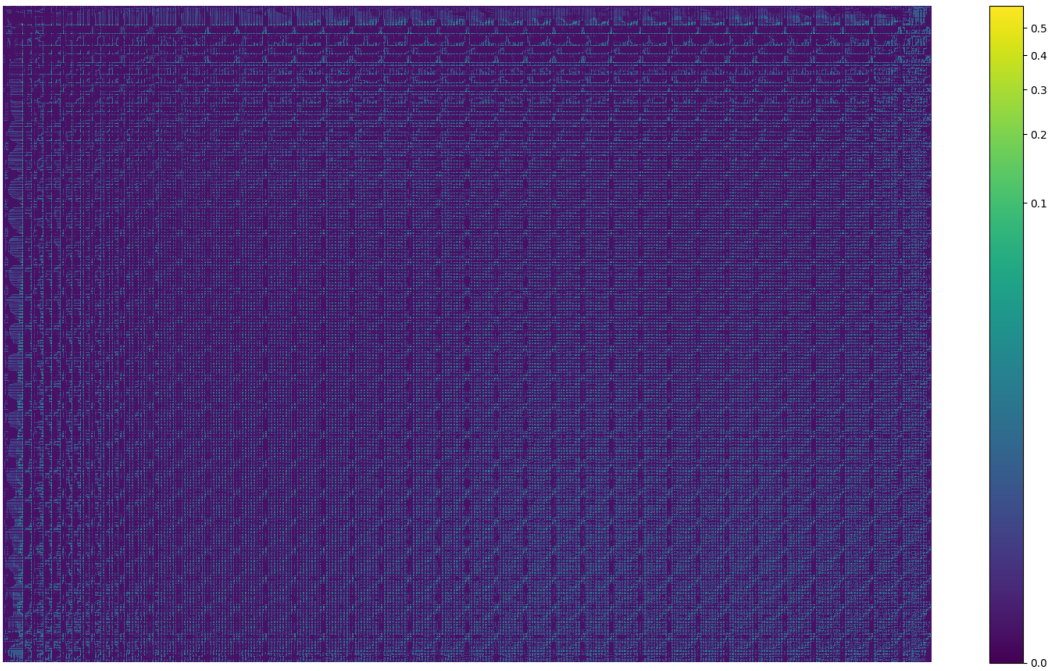

Figure 3 The (unrounded) optimal weightmap found through the prefix-LP.

The proof argument used might be considered useful for applications of the PLP, which has many real-world applications, especially in logistics [20], but the general approach of using a weightmap as an upper bound is not entirely novel [15].

3.3 Verifier

The rounded weights are available on the GitHub page [9]. The following script verifies the conditions of Theorem 11 using exact integer arithmetic:

```
import numpy as np

weights = np.load("upper-bound/weights/rounded-weights.npy")
assert weights.shape == (1189, 841) and weights.dtype == np.int64
assert weights.min() >= 0
assert weights.sum() < 1039_000_000

# prefix-sum matrix over large rectangle
S = np.zeros((weights.shape[0] + 1, weights.shape[1] + 1), dtype=np.int64)
S[1:, 1:] = weights.cumsum(axis=0).cumsum(axis=1)

def all_rect_sums_at_least(w, h, threshold):
    # returns all sums for (w, h) windows as a 2d array
    sums = S[w:, h:] - S[:-w, h:] - S[w:, :-h] + S[:-w, :-h]
    return sums.min() >= threshold

assert all_rect_sums_at_least(26, 37, 1_000_000)  # horizontal
assert all_rect_sums_at_least(37, 26, 1_000_000)  # vertical
```

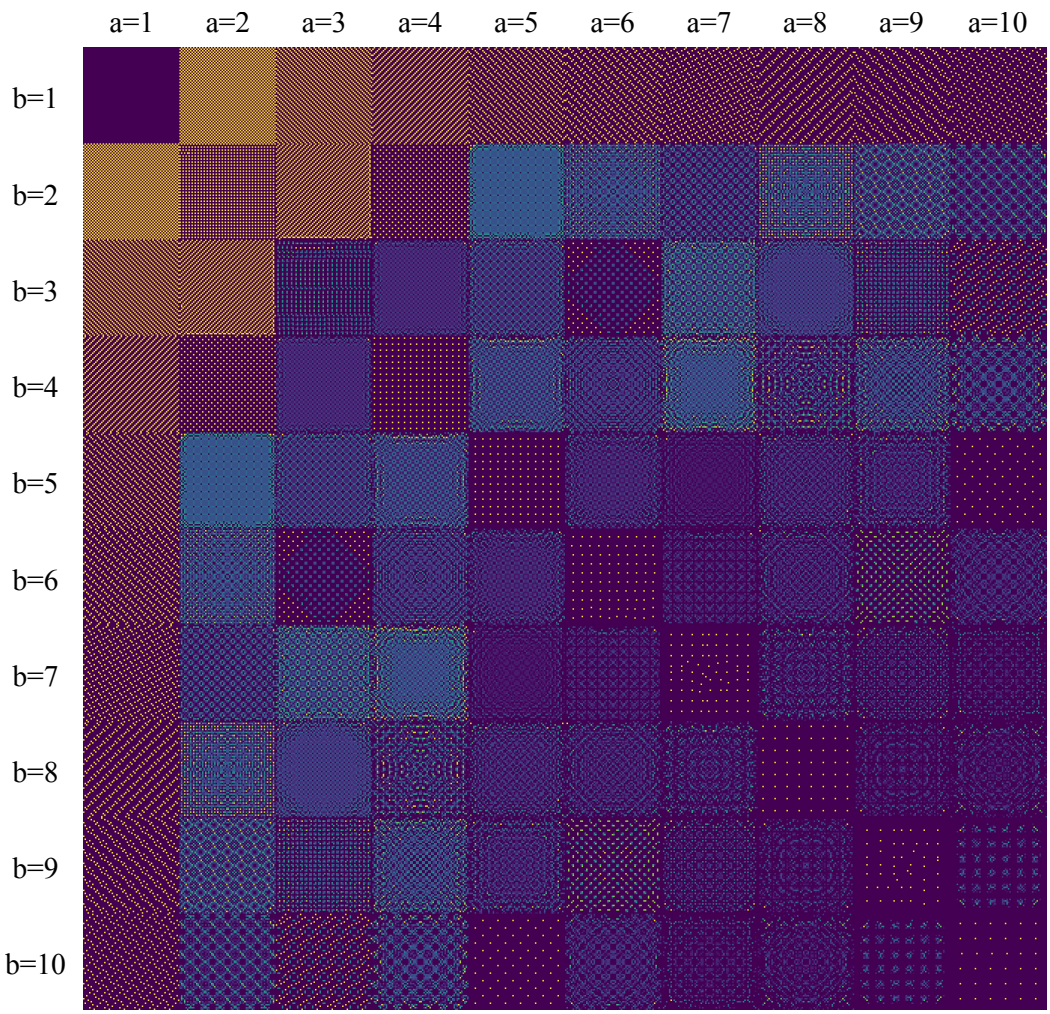

Figure 4 Grid of optimal weightmaps of family of instances $(64, 64, a, b)$.

4 Good-looking weightmaps

The proof technique of generating minimal weightmaps can be applied to other instances of the PLP. By colouring these weightmaps according to their values, we get some good-looking pictures. Figure 4 shows a grid of all computed optimal weightmaps for instances $(64, 64, a, b)$ with $1 \leq a, b \leq 10$. The colourmap between weightmaps is not synchronized; a tone of yellow (indicating the highest weight value across a given weightmap) must not always indicate the same weight value as the same tone in another weightmap.

These weights were computed via Gurobi within a numerical precision of 10^{-9}. The code used to generate these and other weightmaps can be found in the GitHub repository [9]. Because these pictures look nice and display a surprising number of patterns and complexity, we also present these in a digital art gallery, appropriately called the *Gallery of Papel* [8]. The Gallery showcases all optimal weightmaps of instances of the form (S, S, a, b) with $64 \geq S \geq a \geq b \geq 1$, hence containing a total number of $\sum_{S=1}^{64} \sum_{a=1}^{S} a = \sum_{S=1}^{64} \frac{S(S+1)}{2} = 45760$ weightmaps.

Table 1 Remaining A series optimal orthogonal packing results.

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A0	$= 1$	$= 2$	$= 4$	$= 8$	$= 16$	$= 32$	$= 64$	$= 128$	$= 258$	$516 \leq x_1 \leq 518$	$= 1038$
A1		$= 1$	$= 2$	$= 4$	$= 8$	$= 16$	$= 32$	$= 64$	$= 129$	$258 \leq x_2 \leq 259$	$518 \leq x_3 \leq 519$
A2			$= 1$	$= 2$	$= 4$	$= 8$	$= 16$	$= 32$	$= 64$	$= 128$	$= 258$
A3				$= 1$	$= 2$	$= 4$	$= 8$	$= 16$	$= 32$	$= 64$	$= 129$
A4					$= 1$	$= 2$	$= 4$	$= 8$	$= 16$	$= 32$	$= 64$
A5						$= 1$	$= 2$	$= 4$	$= 8$	$= 16$	$= 32$
A6							$= 1$	$= 2$	$= 4$	$= 8$	$= 16$
A7								$= 1$	$= 2$	$= 4$	$= 8$
A8									$= 1$	$= 2$	$= 4$
A9										$= 1$	$= 2$
A10											$= 1$

5 Related Work

The PLP, also known as the *manufacturer's pallet loading problem*, has been studied extensively; see, e.g., the survey by Silva–Oliveira–Wäscher [19]. Widely used computational benchmarks are given by the Cover I and Cover II instance sets, which contain representatives of equivalence classes with area ratio $(L \cdot W)/(a \cdot b) < 51$ and $51 \leq (L \cdot W)/(a \cdot b) < 101$ respectively [2, 4]. For these datasets, Birgin–Lobato–Morabito report that their combined recursive partitioning approach finds optimal solutions for all Cover I and II instances [4]. Heuristic methods have also been evaluated on these sets and achieve good performance in practice [2].

Our A0/A10 instance has a much larger area ratio, $(1189 \cdot 841)/(37 \cdot 26) \approx 1039.4$ and is therefore outside the classical benchmark regime. The natural 0–1 formulations for PLP scale with the number of feasible placements (and with the number of unit cells in a discretized model), which can become extremely large for instances with small items [1]. In our case, even in the integer-millimetre model there are on the order of millions of candidate placements, making direct exact optimization approaches impractical.

Direct upper bounds for PLP have also received significant attention. In particular, Nelißen [18] and Letchford–Amaral [15] study structural LP bounds and related dominance relationships between bounds. We implemented these structural bounds for our instance. Unfortunately, they did not yield an upper bound below 1039. To our knowledge, the specific ISO 216-motivated instance $(L, W, a, b) = (1189, 841, 37, 26)$ has not previously been studied.

6 Open Questions

While A0 and A10 arguably pose the most interesting packing problem inside the A series of paper sizes (as they have the largest difference), one can ask the same question for $A(n)$ and $A(m)$ for $10 \geq n \geq m \geq 0$.

We computed solutions to many of these instances and proved optimality in most of these cases using the algorithm presented by Birgin, Lobato and Morabito [4]. This solved all instances except A10 in A0, A9 in A0, A10 in A1 and A9 in A1. For these instances, we computed a simple upper bound using the area bound of the Minimum Size Instance [16] and found good arrangements by hand to prove the lower bounds. Details for these bounds can be found on the GitHub [9]. Naturally, these bounds could be tightened by computing bounds like we presented for A0 and A10. The computed bounds can be seen in Table 1, blue entries correspond to results where the optimum is not a power of two and are thus, in some sense, surprising.

Another question of interest could be the limits of the weightmap proof method. Does there exist an instance (L, W, a, b) of the PLP where the optimal weight does not correspond to the size of an optimal orthogonal packing? We have searched through many smaller instances by computing the optimal weightmaps but have not found a counterexample. The largest difference between the optimal packing size and total weight we were able to find was at instance $(50, 50, 5, 4)$, where the optimal packing has size 124, but the computed optimal weightmap has a weight-sum of `124.8`. Since the large weight-finding LP is dual to the relaxed direct packing-LP [15], they share a common optimal value under strong duality. Hence, this is equivalent to asking whether the LP-relaxation of the packing-LP ever causes a difference ≥ 1 in optimal value compared to the packing-ILP.

Lastly, Theorem 4 is a statement about *orthogonal* packings. It seems unlikely that allowing non-90° rotations would make a packing of size 1039 possible, but isn't ruled out in this article. The grid-based argument used in our upper bound doesn't seem to align with a non-orthogonal setting.

References

1 Ramón Alvarez-Valdés, Francisco Parreño, and José Manuel Tamarit. A branch-and-cut algorithm for the pallet loading problem. *Computers & Operations Research*, 32(11):3007–3029, 2005. `doi:10.1016/j.cor.2004.04.010`.

2 Ramón Alvarez-Valdés, Francisco Parreño, and José Manuel Tamarit. A tabu search algorithm for the pallet loading problem. *Or Spectrum*, 27(1):43–61, 2005. `doi:10.1007/s00291-004-0183-5`.

3 Frank W. Barnes. Packing the maximum number of m × n tiles in a large p × q rectangle. *Discret. Math.*, 26(2):93–100, 1979. `doi:10.1016/0012-365X(79)90115-8`.

4 E. G. Birgin, R. D. Lobato, and R. Morabito. An effective recursive partitioning approach for the packing of identical rectangles in a rectangle. *The Journal of the Operational Research Society*, 61(2):306–320, 2010. `doi:10.1057/jors.2008.141`.

5 ETH Zurich. Euler: High-performance computing cluster, 2024. Scientific IT Services, ETH Zurich. URL: `https://scicomp.ethz.ch/wiki/Euler`.

6 International Organization for Standardization. ISO 216:2007 – writing paper and certain classes of printed matter – trimmed sizes – a and b series, and indication of machine direction, 2007.

7 Noel Friedrich. Fun Orthonogal Rectangle Packing. `https://noel-friedrich.de/forp`, 2025. [Accessed 16-12-2025].

8 Noel Friedrich. Gallery of papel. `https://noel-friedrich.de/papel`, 2025. [Accessed 03-01-2026].

9 Noel Friedrich. Github repository containing material and code relating to the project. `https://github.com/noel-friedrich/paper-packing`, 2025. [Accessed 03-01-2026].

10 Noel Friedrich. A4 paper is annoying (I have proof), 2026. YouTube video explaining core concepts from this paper. URL: `https://www.youtube.com/watch?v=zDKBCIMkDbw`.

11 Deutsches Institut für Normung. DIN 476: Papierformate, 1976.

12 M. Helbig and W. Hennig. *DIN-Format A 4: e. Erfolgssystem in Gefahr*. Beuth-Kommentare. Beuth, 1988. URL: `https://books.google.ch/books?id=Eo9tAAAACAAJ`.

13 Heinz Isermann. *Obere Schranken für die Lösung des zweidimensionalen Packproblems auf der Basis struktureller Identitäten*, pages 341–348. Springer Berlin Heidelberg, Berlin, Heidelberg, 1991. `doi:10.1007/978-3-642-76537-7_22`.

14 Markus Kuhn. International standard paper sizes, 2025. Created 1996-10-29; last modified 2025-12-08. URL: `https://www.cl.cam.ac.uk/~mgk25/iso-paper.html`.

15 Adam N. Letchford and André Renato Sales Amaral. Analysis of upper bounds for the pallet loading problem. *Eur. J. Oper. Res.*, 132(3):582–593, 2001. doi:10.1016/S0377-2217(00)00163-6.

16 Gustavo H. A. Martins and Robert F. Dell. The minimum size instance of a pallet loading problem equivalence class. *Eur. J. Oper. Res.*, 179(1):17–26, 2007. doi:10.1016/j.ejor.2006.03.009.

17 Neenah Paper. International paper sizes and dimensions. Paper 101 (states that A4 is the most common international letterhead size). URL: https://www.neenahpaper.com/resources/paper-101/international-sizes.

18 Josef Nelißen. How to use structural constraints to compute an upper bound for the pallet loading problem. *European Journal of Operational Research*, 84(3):662–680, 1995. Cutting and Packing. doi:10.1016/0377-2217(95)00030-T.

19 Elsa Silva, José F Oliveira, and Gerhard Wäscher. The pallet loading problem: a review of solution methods and computational experiments. *International Transactions in Operational Research*, 23(1-2):147–172, 2016. doi:10.1111/itor.12099.

20 Saúl Vargas-Osorio and Catya Zúñiga. A literature review on the pallet loading problem. *Lámpsakos*, pages 69–80, 2016. doi:10.21501/21454086.1790.

A Appendix

Lower Bound Part B Placement Certificate

Each line contains integers (x_0, y_0, x_1, y_1) encoding the rectangle $[x_0, x_1] \times [y_0, y_1]$. All rectangles have $(x_1 - x_0, y_1 - y_0) \in \{(26, 37), (37, 26)\}$.

```
% x0 y0 x1 y1
78 0 115 26
115 0 152 26
152 0 189 26
189 0 226 26
78 26 115 52
115 26 152 52
152 26 189 52
189 26 226 52
78 52 115 78
115 52 152 78
152 52 189 78
189 52 226 78
78 78 115 104
115 78 152 104
152 78 189 104
189 78 226 104
189 104 226 130
189 130 226 156
189 156 226 182
189 182 226 208
189 208 226 234
189 234 226 260
189 260 226 286
200 286 226 323
200 323 226 360
200 360 226 397
200 397 226 434
200 434 226 471
200 471 226 508
200 508 226 545
200 545 226 582
200 582 226 619
200 619 226 656
200 656 226 693
200 693 226 730
200 730 226 767
200 767 226 804
200 804 226 841
111 104 137 141
137 104 163 141
163 104 189 141
111 141 137 178
137 141 163 178
163 141 189 178
111 178 137 215
137 178 163 215
163 178 189 215
111 215 137 252
137 215 163 252
163 215 189 252
111 252 137 289
137 252 163 289
163 252 189 289
163 289 200 315
163 315 200 341
163 341 200 367
163 367 200 393
163 393 200 419
163 419 200 445
163 445 200 471
174 471 200 508
174 508 200 545
174 545 200 582
174 582 200 619
174 619 200 656
174 656 200 693
174 693 200 730
174 730 200 767
174 767 200 804
174 804 200 841
111 289 137 326
137 289 163 326
111 326 137 363
137 326 163 363
111 363 137 400
137 363 163 400
111 400 137 437
137 400 163 437
111 437 137 474
137 437 163 474
137 474 174 500
137 500 174 526
137 526 174 552
137 552 174 578
137 578 174 604
137 604 174 630
137 630 174 656
148 656 174 693
148 693 174 730
148 730 174 767
148 767 174 804
148 804 174 841
111 474 137 511
111 511 137 548
111 548 137 585
111 585 137 622
111 622 137 659
111 659 148 685
111 685 148 711
111 711 148 737
111 737 148 763
111 763 148 789
111 789 148 815
111 815 148 841
0 0 26 37
26 0 52 37
52 0 78 37
0 37 26 74
26 37 52 74
52 37 78 74
0 74 26 111
26 74 52 111
52 74 78 111
0 111 37 137
37 111 74 137
74 111 111 137
0 137 37 163
37 137 74 163
74 137 111 163
0 163 37 189
37 163 74 189
74 163 111 189
0 189 37 215
37 189 74 215
74 189 111 215
0 215 37 241
37 215 74 241
74 215 111 241
0 241 37 267
37 241 74 267
74 241 111 267
0 267 37 293
37 267 74 293
74 267 111 293
0 293 37 319
37 293 74 319
74 293 111 319
0 319 37 345
37 319 74 345
74 319 111 345
0 345 37 371
37 345 74 371
74 345 111 371
0 371 37 397
37 371 74 397
74 371 111 397
0 397 37 423
37 397 74 423
74 397 111 423
0 423 37 449
37 423 74 449
74 423 111 449
0 449 37 475
37 449 74 475
74 449 111 475
0 475 37 501
37 475 74 501
74 475 111 501
0 501 37 527
37 501 74 527
74 501 111 527
0 527 37 553
37 527 74 553
74 527 111 553
0 553 37 579
37 553 74 579
74 553 111 579
0 579 37 605
37 579 74 605
74 579 111 605
0 605 37 631
37 605 74 631
74 605 111 631
0 631 37 657
37 631 74 657
74 631 111 657
0 657 37 683
37 657 74 683
74 657 111 683
0 683 37 709
37 683 74 709
74 683 111 709
0 709 37 735
37 709 74 735
74 709 111 735
0 735 37 761
37 735 74 761
74 735 111 761
0 761 37 787
37 761 74 787
74 761 111 787
0 787 37 813
37 787 74 813
74 787 111 813
0 813 37 839
37 813 74 839
74 813 111 839
```

Table 2 ISO 216 A-series paper sizes in millimetres.

Name	Width (mm)	Height (mm)	Nominal Width (mm)	Nominal Height (mm)
A0	841	1189	840.89641525...	1189.2071150...
A1	594	841	594.60355750...	840.89641525...
A2	420	594	420.44820762...	594.60355750...
A3	297	420	297.30177875...	420.44820762...
A4	210	297	210.22410381...	297.30177875...
A5	148	210	148.65088937...	210.22410381...
A6	105	148	105.11205190...	148.65088937...
A7	74	105	74.325444687...	105.11205190...
A8	52	74	52.556025953...	74.325444687...
A9	37	52	37.162722343...	52.556025953...
A10	26	37	26.278012976...	37.162722343...

These coordinates can be verified using the following concise python script:

```
import numpy as np

rects = np.loadtxt("rect_positions.txt", dtype=int)
assert rects.shape == (197, 4)

# check correct dimensions
for x0, y0, x1, y1 in rects:
    assert (x1 - x0, y1 - y0) in {(26, 37), (37, 26)}

# check containment
assert rects[:, 0].min() >= 0 and rects[:, 2].max() <= 227
assert rects[:, 1].min() >= 0 and rects[:, 3].max() <= 841

# check disjointness by checking covering map
covered_map = np.zeros((227, 841))
for x0, y0, x1, y1 in rects:
    covered_map[x0:x1, y0:y1] += 1
assert covered_map.max() <= 1
```