Annals of Operations Research (2025) 350:911–926
https://doi.org/10.1007/s10479-025-06633-5

ORIGINAL RESEARCH

Covering a square with consecutive squares

Janos Balogh[1] · Gyorgy Dosa[2] · Lars Magnus Hvattum[3] · Tomas Attila Olaj[2] · Istvan Szalkai[2] · Zsolt Tuza[2,4]

Received: 11 January 2023 / Accepted: 24 April 2025 / Published online: 20 May 2025

Abstract
In this article we address the following problem. Given are a 1×1 square, a 2×2 square, and so on, finally a $n \times n$ square. What is the biggest square that can be *covered* completely by this given set of "small" squares? It is assumed that the small squares must stand parallel to the sides of the big square, and overlap is allowed. In contrast to the *packing* version of the problem (asking for the smallest square that can accommodate all small squares without overlap) which has been studied in several papers since the 1960's, the covering version of the problem seems new. We construct optimal coverings for small values of n. For moderately bigger n values we solve the problem optimally by a commercial mathematical programming solver, and for even bigger n values we give a heuristic algorithm that can find near optimal solutions. We also provide an expansion-algorithm, that from a given good cover using consecutive squares up to size n, can generate a cover for a larger square using small squares up to size $n+1$. Finally we prove that a simple covering policy can generate an asymptotically optimal covering.

Keywords Square packing · Square covering · Asymptotic behavior · Heuristic method consecutive squares · Board packing

✉ Gyorgy Dosa
dosa.gyorgy@mik.uni-pannon.hu

Janos Balogh
baloghj@inf.u-szeged.hu

Lars Magnus Hvattum
lars.m.hvattum@himolde.no

Tomas Attila Olaj
tomas@olaj.no

Istvan Szalkai
szalkai.istvan@mik.uni-pannon.hu

Zsolt Tuza
tuza.zsolt@mik.uni-pannon.hu

1 University of Szeged, 6720 Szeged, Hungary

2 University of Pannonia, 8200 Veszprém, Hungary

3 Faculty of Logistics, Molde University College, N-6402 Molde, Norway

4 HUN-REN Alfréd Rényi Institute of Mathematics, 1053 Budapest, Hungary

1 Introduction

We address the following question.

Question *Can a square of size $K \times K$ be covered completely by using the consecutive squares of sizes 1×1, 2×2, …, $n \times n$? In other words, what is the largest K, so that the consecutive squares 1×1, 2×2, …, $n \times n$ — called "small squares", denoted as $\mathcal{Q}[n]$ — completely cover $K \times K$, allocating the small squares carefully?*

We allow that the small squares are overlapping. Up to the best knowledge of the authors, this question seems new. On the other hand, the related question, what the smallest accommodation square is where the small squares can be packed without overlap, is not new. This co-problem is originated by Gardner (1966, 1975), where he popularized an unpublished question due to R. B. Britton. We assume that the small squares must have sides parallel to the axes, but they are allowed to touch each other on their sides. For this packing problem, Gardner (1975) listed the best possible (tight) results up to $n = 17$, based on earlier works.

The packing of consecutive squares serves also as a benchmark problem for more general rectangle packing problems, see (Korf, 2003; Moffitt & Pollack, 2006; Simonis & O'Sullivan, 2008). Rectangle packing has many important applications in VLSI design, scheduling, cutting stock and pallet loading problems. In some of these applications, rotation of the rectangles is allowed.

In our recent paper (Balogh et al., 2022) we review the current best upper bounds for the packing problem, and it can also be found on the webpage (The on-line encyclopedia of integer sequences, 2021), searching for the integer sequence A005842. There the best found $UB(n)$ values are listed till $n = 56$ (most of them are tight), where $UB(n)$ is the upper bound for the packing problem with a given parameter n. One of the best lower bounds for large values of n comes by rounding up the square root of the total area of the squares, i.e. $LB_0 = \left\lceil \sqrt{\sum_{k=1}^{n} k^2} \right\rceil = \left\lceil \sqrt{n(n+1)(2n+1)/6} \right\rceil$. Tightness has not been proved for the cases $n = 38$, 40, 42, 48, 52, 53 and 55, while the other values of $UB(n)$ given in The on-line encyclopedia of integer sequences (2021) are proved to be tight for $1 \leq n \leq 56$.

An interesting special case of the problem is $n = 24$, which was unsolved until 2004. As proved already in Watson (1918), $n = 24$ is the unique non-trivial case where $A_n := \frac{n(n+1)(2n+1)}{6}$ is the square of an integer. However, while $LB_0 = 70$, the optimum is 71 as proved by Korf (204). Staying at the very special $n = 24$ case, since the small squares of $\mathcal{Q}[24]$ cannot be packed without overlap into the 70×70 accommodating big square, and their total size is just 70×70, it follows also that if we turn to the *covering* problem, those small squares cannot cover completely the 70×70 big square. Hence the K value that we look for in case of the covering problem, must be smaller than 70 for $n = 24$.

In Balogh et al. (2022) we also review some related packing problems, where an arbitrary set of squares with a total area of at most 1 should be packed into a smallest possible square or into a rectangle of minimum area. In Buchwald and Scheithauer (2016) the authors prove that any set of squares can be packed into the unit square if (i) the total are of the squares is at most $5/9$ and (ii) none of the squares has side length larger than $1/2$. They also proved that $5/9$ is the best possible such value. It is worth mentioning here that in their packing, the patterns fulfill the guillotine pattern properties. Their packing is similar to the First Fit Decreasing Height (FFDH) heuristic of Coffman et al. (1980) designed for the two-dimensional strip packing problem, but they used it both horizontally and vertically.

In Balogh et al. (2022), we give the first asymptotic results for the packing problem. Namely, we show that there exists a constant $c < 1$ such that the square of size $N + cn$ admits a guillotine-type packing of the squares of sizes $1, 2, \ldots, n$, where $N := \sqrt{A_n}$.

Our results for covering. To the best of our knowledge, the *covering* problem was never considered before. Our results are summarized below.

- For small values of n the covering problem is easy, and we give the optimal solutions. These can be calculated by hand or by a mathematical programming solver. There are several ways to formulate the problem using mathematical programming, and we present and discuss two alternative models.
- For moderately bigger values of n, such as $n = 23$, the problem is already hard for mathematical programming solvers, so we create a heuristic algorithm that can find near optimal solutions also for big values of n.
- We provide an expansion-algorithm, that from a given good cover of $K \times K$ for some n, can generate a cover for some $K' > K$ using small squares up to and including $n + 1$.
- We also prove that a simple covering policy can generate an asymptotically optimal covering.

The structure of the paper is as follows. In Section 2 we discuss some small cases in detail, up to $n \leq 7$. In Section 3 we first introduce a binary integer programming formulation that can be used to determine optimal covers, and then present an alternative mixed integer programming model; then, in section 3.2 we present the results of solving the models for up to $n \leq 25$. Section 4.1 gives a heuristic and Section 4.2 introduces an expansion-algorithm, to obtain a lower bound for the case with $n + 1$ based on a known cover using n. In Section 5 we prove that a square of size $N - O(n)$ can be covered with the squares of consecutive sizes $1, 2, \ldots, n$, giving a simple, asymptotically optimal covering algorithm. Some remarks are given in the concluding section.

2 Small cases

Here we discuss the small cases, up to $n \leq 7$. Let $\mathcal{Q}[n]$ denote the squares with sizes $1, 2, \ldots, n$, exactly one copy from each. Note that

$$UB_n := \left\lfloor \sqrt{\sum_{k=1}^{n} k^2} \right\rfloor = \left\lfloor \sqrt{n(n+1)(2n+1)/6} \right\rfloor = \left\lfloor \sqrt{A_n} \right\rfloor \tag{1}$$

is a trivial upper bound for the largest size K_n of a square that can be completely covered by $\mathcal{Q}[n]$. We will denote by Δ_K the increment of K from $n-1$ to n, i.e. $\Delta_K = K_n - K_{n-1}$. If no confusion arises, we will omit the index n and write simply K instead of K_n. Moreover, let L denote a lower bound for K for any fixed value of n.

The values for the cases $n \leq 7$ are shown in Table 1. Their validity can be checked by hand.

The cases $n \leq 3$ are trivial. For $n = 4$ the square 5×5 cannot be covered by $1 \times 1, \ldots, 4 \times 4$, since 4×4 leaves in 5×5 an empty L-shape of $5 + 4 = 9$ unit squares (or cells), but 3×3, 2×2, 1×1 can cover at most $5 + 2 + 1 = 8$ of those empty cells. The bigger cases are less immediate; they are discussed in the following two subsections.

Table 1 The cases $n \leq 7$

n	1	2	3	4	5	6	7
UB_n	1	2	3	5	7	9	11
K	1	2	3	4	6	9	11
Δ_K	1	1	1	1	2	3	2

2.1 The special case of $n = 5$, excluding $K = 7$

For $n = 5$, it is easy to make a cover for $L = 6$. To see that $K = 6$, it remains to exclude $K = 7$. One way to do this is as follows. Suppose there exists a complete cover for $K = 7$. Since it is impossible to cover a 7-long stripe with just the three smallest squares 1×1, 2×2, and 3×3, the union of 5×5 and 4×4 necessarily is a polygon of width 7 and height 7. At this point two rectangles R', R'' are uncovered, of sizes 2×3 and 3×2, in a position where the 3×3 square can cover an area of at most 6 from $R' \cup R''$, and the two smallest squares have a total area of 5. Hence the cover cannot be completed.

Another way to argue is that at least one horizontal side H and one vertical side V of the $K \times K$ square are disjoint from the largest square 5×5. The corner $C = V \cap H$ is contained in just one of the other four squares, because a second square at C would be superfluous (namely the smaller one). If C is covered with the square of size $x \leq 4$, then the side lengths must satisfy $1 + 2 + 3 + 4 + x \geq |V| + |H| = 14$, i.e. $x = 4$ holds and the remaining 3-length parts of V and H should be covered by the 3-square and by the union of the 1-square and 2-square, respectively. But then there is a 1×6 stripe inside $K \times K$ that touches the 1-square and is internally disjoint from all the four. It is impossible to cover this stripe with the 5-square.

2.2 The small cases $n = 6$ and $n = 7$

The largest possible covers for $n = 6$ and $n = 7$ are shown in Figure 1. Consider e.g. $n = 6$. Since we can realize that there is a good cover for $n = 6$ and $L = 9$, moreover for $n = 6$ the upper bound meets this L ($UB_6 = L$), we conclude that $K_6 = 9$, and no more work is required regarding this n. This specific cover is constructed so that the two biggest squares are allocated in two opposite corners (red and green squares of sizes 6 and 5, Fig. 1, left), the next two biggest squares are placed in the other two corners, and the remained few cells are covered by the two smallest squares. The same technique works also for $n = 7$ since $UB_7 = 11$ (cf. Fig. 1, right); but let us note that this simple way of covering will not work after some value of n.

3 Medium cases

For medium values of n, the problem cannot be solved easily by hand. We mean that the problem is easy if we calculate the trivial upper bound UB_n for an n and find a complete cover somehow for $L = UB_n$; in this case we can be sure that an optimal solution has been obtained, i.e. $K = L = UB_n$. But in some other cases, K is smaller than UB_n, or a packing of size UB_n cannot be found easily. For such cases we follow the next idea: we define a mathematical programming model that takes the sizes of the squares as inputs, and solve the

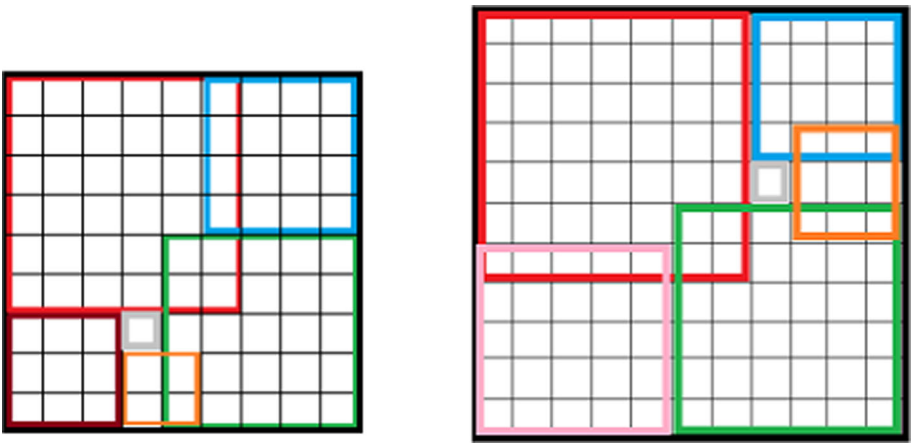


Fig. 1 $n = 6$ (left) and $n = 7$ (right)

model using a commercial solver. If the solver tells us that there is no solution for a certain L value, but there is one for $L-1$, then we know that in fact the optimal solution is $K = L-1$.

3.1 Mathematical programming formulation

Our problem is related to the board packing problem (Abraham et al., 2023; Dósa et al., 2020). We are given a rectangular board with rows $I = \{1, \ldots, |I|\}$ and columns $J = \{1, \ldots, |J|\}$. Each position $(i, j) \in I \times J$ of the board has an integer value $g_{i,j}$ representing a revenue obtained if the position is covered. A set R of rectangles is given, each $r \in R$ with a given height h_r, width w_r, and cost c_r. The objective is to purchase rectangles and place them on the board so as to maximize the profit, which is equal to the values of the covered positions minus the cost of purchased rectangles. The revenues, heights, widths, and costs are expressed as integers. Rectangles must be placed with their sides parallel to the sides of the board. They are allowed to overlap, but the revenue from a given board position can only be collected once.

By setting $|I| = |J| = K$, the revenue equal to 2 for every cell, and defining the set of rectangles R as the n squares of consecutive sizes, each with a cost of 1, then if the whole board can be covered, the optimal value is at least $2K^2 - n$. On the other hand, if the optimal value is less than $2K^2 - n$, we can conclude that no cover of $K \times K$ exists using small squares up to size n. In Section 4 we will adapt a heuristic developed for the board packing problem to handle big cases. However, for medium cases, we will apply a specialized mathematical programming formulation.

In the specialized model, we need to place all the available objects, and we need to cover all cells. We can utilize some symmetry, or even try to solve the model with certain squares in fixed positions. Let K be the size of the square that needs to be covered, and let $n \times n$ be the size of the largest consecutive square. Let T_r be the set of coordinates such that square $r \times r$ can be placed with its upper left corner at such coordinates. E.g., $T_r = \{(i, j) : i = 1, \ldots, K - r + 1, j = 1, \ldots, K - r + 1\}$. Let B_{ijr} be defined as the set of positions (u, v) for the top left corner of the square of size $r \times r$ that will result in position (i, j) being covered by that square. Define the binary variable x_{ijr} to be 1 if and only if the square of size $r \times r$ is located with the upper left coordinate at (i, j). Let continuous variable

Table 2 The cases $8 \leq n \leq 21$

n	8	9	10	11	12	13	14	15	16	17	18	19	20	21
UB_n	14	16	19	22	25	28	31	35	38	42	45	49	53	57
K	13	16	18	21	24	28	31	34	38	41	45	49	53	57
Δ_K	2	3	2	3	3	4	3	3	4	3	4	4	4	4

y be 0 if all cells are covered, and at least 1 otherwise. The model can be written as:

$$\min \ y \tag{2}$$

$$\sum_{(i,j)\in T_r} x_{ijr} = 1, \qquad r \in \{1, \ldots, n\} \tag{3}$$

$$y + \sum_{r\in\{1,\ldots,n\}} \sum_{(u,v)\in B_{ijr}} x_{uvr} \geq 1, \qquad i = 1, \ldots, K, \ j = 1, \ldots, K \tag{4}$$

$$x_{ijr} \in \{0, 1\}, \quad r \in \{1, \ldots, n\}, (i, j) \in T_r \tag{5}$$

$$y \geq 0. \tag{6}$$

The problem has some symmetry that can be exploited. For example, one may require that the largest square should be placed to the right and above the middle of the covered square. E.g., we reduce T_n without loss of generality as follows. Let $D_n = \lceil (K - n + 1)/2 \rceil$. Then, we set $T_n = \{(i, j) : i = 1, \ldots, D_n, j = 1, \ldots, D_n\}$.

3.2 Results using the mathematical models

We solve the models using the commercial mixed-integer programming solver CPLEX. When using the models for values in the range of $n = 8, \ldots, 21$, their performance do not differ very much. However, the second, specialized model, typically has slightly lower running times when attempting to prove optimality. In Table 2 we provide the optimal results for $8 \leq n \leq 21$. Any case with $UB_n = K$ is simple, or at least seems simple. It is enough to find a good cover for the value of UB_n. But, note that this good cover cannot be easily found for a relatively big n, like $n = 21$. These are the cases $n \in \{9, 13, 14, 16, 18, 19, 20, 21\}$.

For the other cases, i.e. for $n \in \{8, 10, 11, 12, 15, 17\}$, the optimal value of L can be obtained so that CPLEX returns the non-existence of a complete cover for $L + 1$. The running time of CPLEX was about 7 seconds for $n = 11$ (to conclude that $K = 22$ is not possible), but for $K = 21$ CPLEX found an optimal solution soon. For $n = 12$, excluding $K = 25$ needed about 12 seconds. For $n = 15$, excluding $K = 35$ needed about 214 seconds. For $n = 17$, CPLEX needed more than 10 hours of running time to conclude that $K = 42$ cannot be covered, thus proving that $K = 41$ is optimal. Thus, for larger values of n, solving models using CPLEX becomes too hard.

The experience is that finding a good cover is easy if it exists, but excluding some K value is harder and harder as n grows. The case of $n = 21$ with a cover for $K = 57$ is detailed in Table 3, and is illustrated in Figure 2. In the table we give the top-left coordinates of each small square $r \times r$ in the covering, $r \leq n$, where "-" means that the square in question is not needed for the covering.

Some further results can be found in Table 4 for $22 \leq n \leq 25$. Here CPLEX cannot provide an optimal result in reasonable time. For all cases covered in Table 4, the exact value

Table 3 Coordinates for $n = 21$, $K = 57$, a complete covering

r	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Top	-	-	43	42	41	30	35	35	34	21	21	46	45	31	43
Left	-	-	27	23	18	38	20	27	35	48	37	18	45	44	31

r	16	17	18	19	20	21
Top	1	41	17	22	1	1
Left	22	1	20	1	38	1

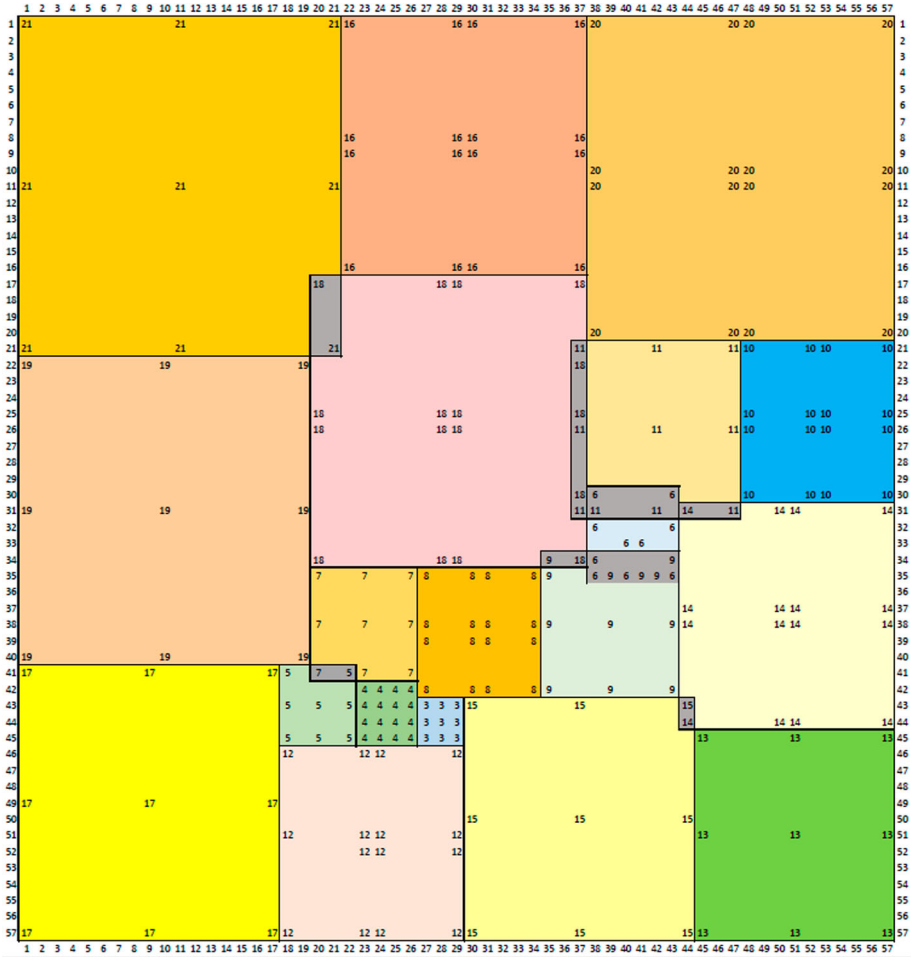


Fig. 2 Cover of $K = 57$ using squares up to size $n = 21$

Table 4 The cases $22 \leq n \leq 25$

n	22	23	24	25
UB_n	61	65	70	74
L	60	64	68	72

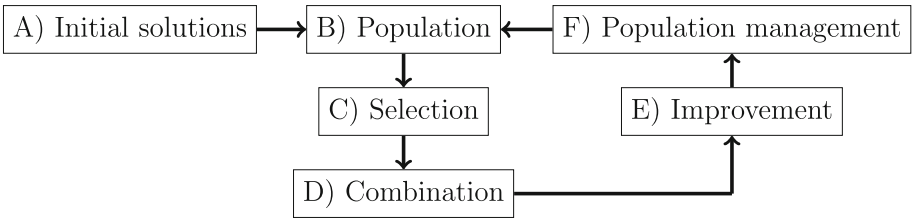


Fig. 3 Schematic overview of the heuristic used to find large coverings

of K is between L and UB_n, and has yet to be determined. For example, in case of $n = 22$, we have $L = 60 \leq K \leq UB_{22} = 61$.

4 Big cases

We define big cases as those where CPLEX is far from being able to prove optimality: CPLEX may be able to find some lower bounds, although even this can be time-consuming when the square to cover is larger than 70×70. We then look for other means to generate lower bounds, using a heuristic algorithm as well as an expansion algorithm.

4.1 Heuristic search

We have seen that after some time the solver is not enough. We still can apply a heuristic to find valid lower bounds on K for a given n. That is, the board packing problem (Abraham et al., 2023) presented in Section 3.1 can be generated for given values of K and n, and a computer search can attempt to find a feasible solution with an objective function value that implies that a cover has been found.

The heuristic that we use is an evolutionary algorithm, modified from the algorithm proposed in Abraham et al. (2023) to solve general instances of the board packing problem. Figure 3 shows the outline of the method, which is based on the leading metaheuristic for the highly competitive combinatorial task of solving vehicle routing problems (Vidal, 2022), the hybrid genetic search.

First, a set of initial solutions is generated. Some of these solutions are generated by a construction heuristic, where the squares are sorted in decreasing size, and then inserted in a location that maximizes the number of covered cells. The construction heuristic is deterministic, but can be repeated several times by moving the first square to the last position to modify the insertion sequence. Other initial solutions are simply generated by randomly placing all the squares onto the grid. As a refinement of this, we force the four largest squares to be initially placed in a corner. The method is allowed to restart. In that case, the single best solution found is kept, and the other initial solutions are generated at random, without forcing the largest squares to be placed in the corners.

In a regular iteration of the heuristic, two solutions are selected from the current population using two binary tournaments. These are then combined into two new solutions using a type of uniform crossover, where each square and its placement is assigned at random from one parent to one child. The new solutions are then subjected to a local search improvement. Here, each square can be moved one step in either Cartesian direction, or if this does not lead to an improvement after considering all the squares, one square is selected to be temporarily removed and then inserted into the best possible position (covering as many cells as possible).

The initial population has a fixed number of solutions included. All new solutions generated are included into the population, as long as their objective function values are different from other solutions already included (this is to ensure a diverse population). When the population reaches a certain size, it is reduced down to the same number of solutions as the initial population had. The procedure is based on (Vidal, 2022), where we combine measures of solution quality and solution diversity. If this reduction happens several times without the search having found any new best solution, the search is restarted.

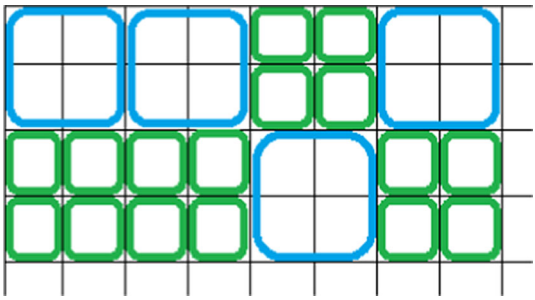
4.2 Expansion algorithm

Our next idea to obtain lower bounds is an *expansion algorithm*. Given an original cover of a square of dimensions $K \times K$, using consecutive squares up to size n, we can apply an expansion algorithm to find a covering of a larger square $K' = K + s$ using consecutive squares of sizes from 2 to $n + 1$. The 1×1 square can then be added arbitrarily to this cover. The size of s and thus K' depends on the minimum number of squares to appear in any given row or column in the original cover of the $K \times K$ square.

The idea of the algorithm is easier to express using rectangles. Let us assume that we have a rectangle of size $K^I \times K^J$ that is covered by a set of n rectangles where rectangle r has the size $I_r \times J_r$. We will then compute the number of rectangles in each row that are actively used to cover the bigger rectangle. Let s be the minimum number of such rectangles across all rows. Then, we will find a cover of a rectangle of size $K^I \times (K^J + s)$ by using n rectangles of sizes $I_r \times (J_r + 1)$. This will be achieved by increasing the size of each individual rectangle by one column and shifting them horizontally to cover the enlarged area.

By applying this procedure to a $K \times K$ square covered with n consecutive squares, we can first expand the squares horizontally. Then, by transposing the result and applying the procedure again, we can obtain a figure of size $(K + s) \times (K + s')$ covered by consecutive squares of sizes from 2 to $n + 1$. In the case that $s \neq s'$, we may then have to redo the process while ensuring that the square is extended in equal length $\min\{s, s'\}$ in both directions.

Fig. 4 Initial 4×8 cover



Algorithm 1 Pseudo-code for expansion algorithm.

Input: the original cover of a $K^I \times K^J$ rectangle using n rectangles of sizes $I_r \times J_r$.
for rectangles r appearing in the first column **do**
  $h(r) \leftarrow 0$ ▷ Cannot move these rectangles horizontally
for column $j = 2, 3, \ldots, K$ **do**
  for rectangles r appearing in column j but not in column $j-1$ **do**
    Let Q be the set of rectangles appearing in column $j-1$ and overlapping rows of r
    $h(r) \leftarrow \min_{q \in Q} h(q) + 1$ ▷ Can move this rectangles one more than neighbors
Let H be the set of squares appearing in column K
$s \leftarrow \min_{r \in H} h(r) + 1$ ▷ New size is $K^I \times (K^J + s)$
for rectangle $r = 1, 2, \ldots, n$ **do**
  Move rectangle r to the right by $\min\{h(r), s-1\}$ units
  Resize the rectangle by increasing its width by 1 unit
Output: a covering of $(K^I) \times (K^J + s)$ using n rectangles of sizes $I_r \times (J_r + 1)$

Proposition 1 *The expansion algorithm works, i.e. provides a covering for each n and s.*

However, this is only a cover, we do not know whether it is optimal or not.

Remark 2 It is essential in executing Algorithm 1 horizontally for s and vertically for s' that s' be computed only after the determination of s and construction of the $K^I \times (K^J + s)$ cover. Computing both s and s' from the initial configuration and executing expansion in both directions simultaneously may or may not work. Among the covers constructed for $\mathcal{Q}[n]$ we have not found a counterexample so far. However in Fig. 4 we show an arrangement of 20 squares that cover an area of 4 rows and 8 columns, but its expansion leaves an uncovered cell (crossed in Fig. 5) if s' is calculated from the original cover rather than from the s-expanded one.

4.3 Results using the heuristic and the expansion-algorithm

For small values of K, the heuristic finds lower bounds as good as those determined by using the commercial MIP solver. However, since the heuristic does not provide any upper bounds, the commercial MIP solver is preferred up to values of K around 65. For larger K, the MIP solver is unable to provide conclusive results, and the heuristic is faster at finding lower bounds.

Table 5 shows results for $25 \leq n \leq 30$. The time taken to find a cover for a given combination of L and n by the heuristic for this range is typically less than one hour. However, when the heuristic fails to find a cover, we simply do not know whether the method just failed

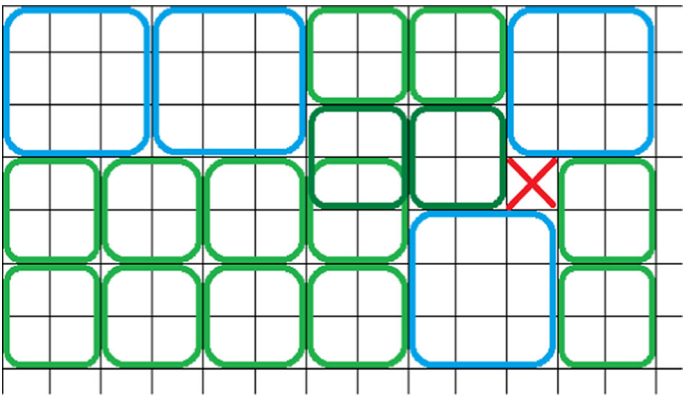


Fig. 5 Simultaneous computation of s and s' leaves a cell uncovered

Table 5 The cases $25 \leq n \leq 30$ and results using the heuristic (L) and the expansion algorithm (L')

n	25	26	27	28	29	30
UB_n	74	78	83	87	92	97
L	72	76	81	85	90	95
L'	NA	76	80	85	89	94

to find the cover or whether a cover did not exist. For $n = 25$, which was the largest case considered by CPLEX, the heuristic is able to find the same lower bound. The table also shows lower bounds produced by applying the expansion algorithm. To generate these, we consider the cover found by the heuristic for n and create a new cover using $n + 1$ small squares. This is able to find the same lower bounds as the heuristic for $n = 26$ and $n = 28$ by using the output from the heuristic for $n = 25$ and $n = 27$, respectively.

To stress test the heuristic, we chose $n = 100$, and attempted to find the largest value of K such that we can prove a feasible cover using the consecutive squares from 1 to $n = 100$. The heuristic succeeded at finding a solution for $K = 559 \leq UB_{100} = 581$. The run that succeeded terminated after 3, 166 seconds, but before this, two other runs with different random seeds had failed after running for four days each. Thus, even finding lower bounds is challenging for such large values of n.

In the cover for $n = 100$ and $K = 559$ that was found by the heuristic, there are at least $s = 6$ squares in each row and each column (after performing the extension in one direction). Thus, by the expansion algorithm we also get a cover for $K' = 565$ using $n = 101$; for $K'' = 571$ using $n = 102$, and so on.

5 The asymptotic case

Recall that $N = \sqrt{A_n}$ with $A_n = \frac{n(n+1)(2n+1)}{6}$ is an upper bound on the size of a square coverable by the squares $1 \times 1, 2 \times 2, \ldots, n \times n$. In this section we prove that this bound is asymptotically tight, i.e., a cover can be constructed for a square of area at least $(1 - o(1))A_n$ as $n \to \infty$. More explicitly, the square of side length $N - 2n$ admits a cover with those smaller squares. We prove this fact in the following more general form.

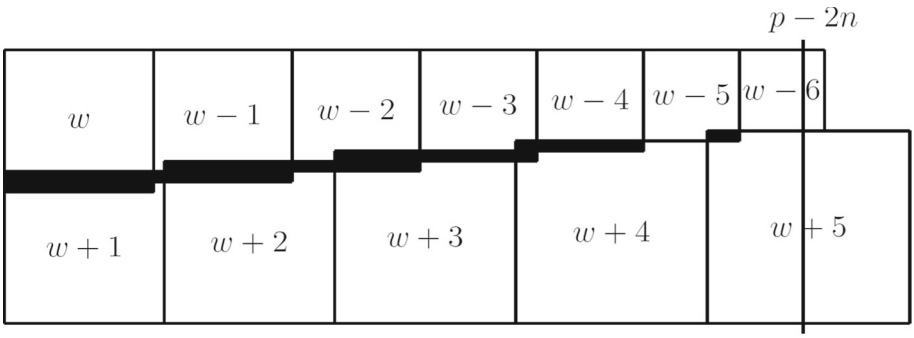


Fig. 6 Example of a slice in which $5 + 7$ squares cover total width at least $p - 2n$; i.e., $s = 5, t = 7$ (cf. text) and w denotes $n - 5$. Dark areas are covered twice

Theorem 3 *If $p \cdot q \le A_n$ and $|p - q| < 3n$, then a $(p - 2n) \times (q - 2n)$ rectangle can be covered with the squares of side lengths $1, 2, \ldots, n$.*

Proof We apply induction on n, assuming that for all $n' < n$, every $(p' - 2n') \times (q' - 2n')$ rectangle can be covered with the squares of side lengths $1, 2, \ldots, n'$ provided that $p' \cdot q' \le A_{n'}$ and $|p' - q'| < 3n'$.

Let R denote a $p \times q$ rectangle (p wide and q high), and assume $p \le q$. The task is to cover the $(p - 2n) \times (q - 2n)$ rectangle obtained from R by trimming a $2n$-wide rectangle from its right end and a $2n$-high rectangle from the top. If $p \le 2n$, then the task is void, which is necessarily the case when $2n(2n + 1) > A_n$ (that means $n < 11$). Beyond that, the validity of the theorem can very easily be checked for a wider range of not too large n; we demonstrate this for $n = 24$. We then have $A_{24} = 4900$, $p \le \sqrt{A_{24}} = 70$, and assuming $p \ge 2n + 1 = 49$, $q \le \sqrt{A_{24}}/49 = 100$ holds. Hence $p - 2n \le 22$ and $q - 2n \le 52$. On the other hand, even the rectangle 22×69 is obviously coverable with just the three largest squares 24, 23, 22.

For general n we take the $s + t$ largest squares, i.e. of sizes $n, n - 1, \ldots, n - s - t + 1$, determined as the smallest natural numbers s and t satisfying the inequalities

$$p - 2n \le \sum_{i=n-s+1}^{n} i \quad \text{and} \quad p - 2n \le \sum_{i=n-s-t+1}^{n-s} i\,.$$

From these squares we create a "slice" that completely covers a rectangle of width $p - 2n$ and height $2n - s - t + 1$, as sketched in Figure 6; or height $2n - s - t$ if the case explained in the next paragraph occurs. The unified notation $2n - s - t + \varepsilon$ will be used, with the understanding that ε can mean either of 0 and 1. We arrange the s largest squares in increasing order—with their bottom sides aligned—and the next t largest squares in decreasing order—with their top sides aligned—both from left to right, in the way that square $n - s - t + 1$ touches square n from above, and the vertical left sides of the squares $n - s + 1$ and $n - s$ touch the left side of R, the lower left corner of square $n - s + 1$ being the same as that of R.

It may occur that the bottom-left corner of square $n - s - t + 1$ is located to the left of the top-left corner of square n. In that case an uncovered area would be surrounded by the four squares $n, n - 1, n - s - t + 2, n - s - t + 1$. This situation can be handled by aligning the top row lower by 1; that is, square $n - s - t + 1$ should then touch square $n - 1$, rather than n. This yields a slice of height $2n - s - t$, as one of the options in the notation $2n - s - t + \varepsilon$. For

instance, concerning the example of Figure 6, if $\sum_{i=1}^{4}(w+i) < p-2n \leq \sum_{j=0}^{5}(w-j)$ holds instead of $p-2n > \sum_{j=0}^{5}(w-j)$ then $w-6$ does not belong to the slice.

Since the bottom squares are larger than the top ones, we certainly have $t \geq s$. Moreover, the height of the slice is $2n-s-t+\varepsilon$, hence the largest vertical overlap between top and bottom squares is $t-s+(1-\varepsilon) \leq t-s+1$. A key point in the computation will be that we have

$$t-s \leq 2.$$

To prove this, we first observe that for every n the following chain of inequalities is valid:

$$\begin{aligned} p-2n \leq N-2n &= \sqrt{\frac{n(n+1)(2n+1)}{6}} - 2n \\ &< \frac{3}{5}(n+1)\sqrt{n} - 2n < (n-\sqrt{n}+2)\sqrt{n} - 2n \\ &= (n-2\sqrt{n})(\sqrt{n}-1)\,. \end{aligned}$$

By the choice of s, the sum of the $s-1$ largest squares does not reach $p-2n$, and the smallest of them is bigger than $n-s$ (hence all are), thus

$$(n-s)(s-1) < p-2n < (n-2\sqrt{n})(\sqrt{n}-1)\,,$$

implying $s < \sqrt{n}$. Similarly, by the choice of t, the sum of the $t-1$ squares between $n-s$ and $n-s-t+2$ does not reach $p-2n$, and the smallest of them is bigger than $n-s-t > n-\sqrt{n}-t$, thus

$$(n-\sqrt{n}-t)(t-1) < p-2n < (n-2\sqrt{n})(\sqrt{n}-1)\,,$$

implying $t < \sqrt{n}$. Moreover, we obviously have

$$s \geq \left\lceil \frac{p-2n}{n} \right\rceil \geq \frac{p-2n}{n}$$

and from $s \leq t < \sqrt{n}$ it follows that the size $n-s-t+1$ of the smallest square in the top row is larger than $n-2\sqrt{n}$, thus

$$t \leq \left\lceil \frac{p-2n}{n-2\sqrt{n}} \right\rceil < \frac{p-2n}{n-2\sqrt{n}} + 1.$$

As a consequence,

$$t-s < 1 + \frac{(p-2n)2\sqrt{n}}{n^2-2n\sqrt{n}} \leq 1 + \frac{2(N-2n)\sqrt{n}}{(n^{3/2}-2n)\sqrt{n}} < 3.$$

Here the last step is equivalent to $N < n^{3/2}$, what clearly is a valid inequality because $6N^2/n = (n+1)(2n+1) < 6n^2$ holds for all $n > 1$. Of course, $t-s < 3$ means $t-s \leq 2$, as claimed.

By the choice of s and t, the two rightmost squares, namely n and $n-s-t+1$, reach $p-2n$ but do not reach $p-n$. Hence a rectangle at least $(n+1) \times (2n-s-t+\varepsilon)$, with area bigger than $2n^2 - 2n\sqrt{n} + 2n - 2\sqrt{n}$, remains uncovered in R up to height $2n-s-t+\varepsilon$, within which the width $p-2n$ is entirely covered. On the other hand, as the largest vertical overlap is at most $t-s+1 \leq 3$, the area covered twice within width $p-n$ by the squares

used so far is not larger than $3p - 3n$. It is smaller than $\frac{9}{5}n\sqrt{n} - 3n + \frac{9}{5}\sqrt{n}$, using the upper bound $p < \frac{3}{5}(n+1)\sqrt{n}$. Thus, the blank area exceeds the double-covered one by more than

$$n^{1/2} \cdot \left(2n^{3/2} - \tfrac{19}{5}n + 5n^{1/2} - \tfrac{19}{5}\right) .$$

Here one can see that for all $n \geq 1$ both factors are increasing functions of $\sqrt{n}$ (and also of n), thus the product is positive already for $n = 4$ (hence for all $n \geq 4$).

Consequently, cutting off the $p \times (2n - s - t + \varepsilon)$ rectangle from the bottom of R, we obtain a $p \times q'$ rectangle with $q' = q + s + t - 2n - \varepsilon$, that should be covered with the squares from 1 to $n' := n - s - t$. The above calculations show that

$$p(q - q') = A_n - A_{n'} - |\text{double}| + |\text{blank}| > A_n - A_{n'} ;$$

thus, as $pq < A_n$ holds by assumption, we obtain $pq' < A_{n'}$. Consequently, the theorem follows by induction, once we verify that $|p - q'| < 3n'$ also holds.

Since $0 \leq q - p < 3n$ has been assumed, we now have $-2n + s + t - \varepsilon \leq q' - p < n + s + t - \varepsilon$. In this way, if n is not very small, we obtain

$$\begin{aligned} 0 < n + s + t - \varepsilon < n + 2\sqrt{n} \leq\ & 2n - 2\sqrt{n} < |-2n + s + t - \varepsilon| \\ \leq\ & 2n + 1 - s - t < 2n + 1 - 3(s + t) + 4\sqrt{n} < 3(n - s - t) = 3n' ; \end{aligned}$$

e.g., $n \geq 18$ is sufficient.

This inductive proof also provides a linear-time algorithm that generates a square cover of any given rectangle with the parameters p, q, n satisfying the conditions of the theorem. (In many steps of the procedure, consecutive slices will alternate between horizontal and vertical position.) □

6 Conclusions

We studied the following question. Given n squares, one of size 1, one of size 2, and so on, up to one of size n, what is the biggest square size $K \times K$ that can be completely covered by the set of small squares?

Up to $n \leq 6$ we can solve the problem easily by hand. Already for $n = 7$ the case is not trivial. Between $n = 8$ and $n = 21$ we are able to determine the optimal solutions by using a commercial mathematical programming solver, and for some even bigger values of n, we still can get close lower and upper bounds on K.

We also introduced a heuristic algorithm that is able to provide lower bounds for relatively big n values, such as $n = 100$. It is not clear yet, how far we are here from the optimal values. Some other heuristic algorithms that are more specialized may be able to find improved lower bounds.

We introduced an expansion-like algorithm, which is able to make a cover for any $n' > n$ value, if a cover is given for n. Here there are at least two interesting questions. One is that we do not know how good this algorithm is in the sense that, e.g., if we have an optimal solution for n, how far will the constructed cover be for $n + 1$ (in the worst case) from the optimal solution. The other question is the following: We have seen that if there are at least s squares in the cover of a square of size L in the horizontal (vertical) direction, then we can expand the squares in the horizontal (vertical) direction to get a cover for a bigger square of size $L + s$. But we also gave a small counterexample against horizontal and vertical expansion in parallel, to show that this expansion procedure should be made in two phases: first we count s in one direction and make the expansion, then we count s again in the other

direction and make the expansion in the other direction. If we count s for both directions before the expansions, this can be a problem, as it is possible that some hole is created. But our counterexample has some squares with the same size, so not one 1×1, one 2×2, and so on. Can this phenomenon (creating a hole) appear also for the set of $1 \times 1, 2 \times 2, \ldots, n \times n$ squares, and determining s before the expansions in both directions?

During our investigation, we used the trivial upper bound (for the size of K of the biggest covered square) as $K \leq UB_n = \left\lfloor \sqrt{\sum_{k=1}^{n} k^2} \right\rfloor$. Naturally, if we do have a covering of a square of size L where $L = UB_n$, we know for sure that this is the optimal value. For some small values of n, $K = UB_n$ really is the optimal value, but unfortunately, from some bigger values of n, this is not the case. Thus, it would be useful to know some other, more effective upper bound. At moment we have only this upper bound UB_n, which we know by example is not always tight, so this is a question whether one can find some tighter bound.

It is also an interesting question about the nature of the "gap" between UB_n, the trivial upper bound, and K, the optimal value. It is clear (following from our asymptotic result) that $\frac{UB_n - K}{UB_n} \to 0$ as $n \to \infty$. It would be nice to give a more exact characterization about the gap.

Finally, we hope that our paper will be interesting for other researchers and opens a new direction of research, similarly to the packing version of this problem.

Acknowledgements This research was supported in part by the National Research, Development and Innovation Office – NKFIH under the grant SNN 129364, and by the grant TKP2021-NVA-09 of the Ministry for Innovation and Technology, Hungary.

Funding Open access funding provided by University of Pannonia.

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Not applicable.

References

Abraham, G., Dosa, G., Hvattum, L. M., Olaj, T. A., & Tuza, Z. S. (2023). The board packing problem. *European Journal of Operational Research, 308*, 1056–1073.

Balogh, J., Dosa, G., Hvattum, L. M., Olaj, T., & Tuza, Z. S. (2022). Guillotine cutting is asymptotically optimal for packing consecutive squares. *Optimization Letters, 16*, 2775–2785. https://doi.org/10.1007/s11590-022-01858-w

Buchwald, T., & Scheithauer, G. (2016). *A 5/9 theorem on packing squares into a square*. Preprint MATH-NM-04-2016, TU Dresden.

Coffman, E. G., Jr., Garey, M. R., Johnson, D. S., & Tarjan, R. E. (1980). Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing, 9*, 808–826.

Dósa, G., Hvattum, L. M., Olaj, T., & Tuza, Z. S. (2020). The board packing problem: Packing rectangles into a board to maximize profit. In I. Vassányi (Ed.), *Proceedings of the Pannonian Conference on Advances in Information Technology (PCIT 2020)* (pp. 10–16). Veszprém, Hungary: University of Pannonia.

Gardner, M. (1966). Mathematical games: The problem of Mrs. Perkins' quilt, and answers to last month's puzzles. *Scientific American, 215*(3), 264–272.

Gardner, M. (1975). Mrs. Perkins' quilt and other square-packing problems. In *Mathematical Carnival*. New York: Alfred A. Knopf (pp. 139–149).

Korf, R. E. (2003). Optimal rectangle packing: Initial results. In: *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS 2003)*, (pp. 287–295).

Korf, R. E. (2004). Optimal rectangle packing: New results. In: *Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS 2004)*, (pp. 142–149).

Moffitt, M. D., & Pollack, M. E. (2006). Optimal rectangle packing: a meta-CSP approach. In: *Proceedings of the 16th International Conference on Automated Planning and Scheduling (ICAPS 2006)*, (pp. 93–102).

Simonis, H., & O'Sullivan, B. (2008). Search strategies for rectangle packing. In: *Constraint Programming 2008, Lecture Notes in Computer Science*, (vol. 5202 pp. 52–66).

Vidal, T. (2022). Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers and Operations Research, 140*, Article 105643.

The on-line encyclopedia of integer sequences, sequence a005842. Published electronically at http://oeis.org, accessed: 26 March 2021.

Watson, G. (1918). The problem of the square pyramid. *Messenger of Mathematics, New Series, 48*, 1–22.