On central placements of new vertices in a planar point set

Citation for the original published paper (version of record):
Damaschke, P., Ekstedt, F., Salman, R. (2025). On central placements of new vertices in a planar point set . Theoretical Computer Science, 1025. http://dx.doi.org/10.1016/j.tcs.2024.114973

(article starts on next page)

Theoretical Computer Science 1025 (2025) 114973

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

On central placements of new vertices in a planar point set ☆

Peter Damaschke [a,*], Fredrik Ekstedt [b], Raad Salman [b]

[a] Department of Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, SE-41296 Gothenburg, Sweden
[b] Fraunhofer-Chalmers Research Centre for Industrial Mathematics, SE-41288 Gothenburg, Sweden

ARTICLE INFO

Keywords:
Grid graph
Weighted sum of distances
Median

ABSTRACT

The vertices of an edge-weighted clique shall be placed in the plane so as to minimize the sum of all weighted distances, called the spread. Driven by practical applications in factory layout planning, we consider this problem under several constraints. First we show, in the Manhattan metric, the NP-completeness of the version where some vertices are already placed, and some minimum distance is prescribed between any two vertices. However, we can optimally append one new vertex to n placed vertices in $O(n^2)$ time. For the problem without minimum distance requirements but with many unplaced vertices, we give some structural properties of optimal solutions.

1. Introduction

Let a plane be equipped with a Cartesian coordinate system and with either the Manhattan metric or the Euclidean metric. We are given a set of vertices, and non-negative weights of all pairs of vertices. The problem is to place the vertices at points in the plane, so as to minimize the *spread*, defined as the weighted sum of their pairwise distances. The unconstrained problem would be trivial, as the optimum solution would place all vertices at the same point. But we will consider constraints, e.g.: some vertices are already placed at fixed points, or the vertices must be placed at pairwise distinct points of a square grid, or some minimum distance is prescribed for each vertex pair.

There can be different reasons why some vertices are already placed at fixed points. The simplest practical reason is that some machines have fixed locations and cannot be moved. Placed vertices also appear when we solve a problem instance by iteratively deciding on placements, e.g., during a branch-and-bound procedure. In the following, n and m denotes the number of placed and unplaced vertices, respectively.

This type of problems appears in factory layout planning [1,2]. In this application domain, the vertices are reference points of machines or other resources that shall be placed in a factory hall. Weights represent the strength of the demand to place two machines close to each other. For instance, a weight can directly model the amount of material flow between the two machines. Minimum distances can be required simply because the machines need space, or for security reasons. We mainly consider the Manhattan metric, corresponding to movements of material and workers in aisles in the four main directions. We are primarily interested in optimal solutions rather than in approximations: Practical use cases are often small, such that we can afford the necessary computation time, keeping in mind that the computed and deployed solutions stay for a long time.

☆ This article belongs to Section A: Algorithms, automata, complexity and games, Edited by Paul Spirakis.

* Corresponding author.
E-mail addresses: ptr@chalmers.se (P. Damaschke), fredrik.ekstedt@fcc.chalmers.se (F. Ekstedt), raad.salman@fcc.chalmers.se (R. Salman).

https://doi.org/10.1016/j.tcs.2024.114973
Received 24 April 2024; Received in revised form 21 October 2024; Accepted 13 November 2024

Available online 17 November 2024

For the Euclidean metric, the problem of minimizing the sum of weighted distances to $n > 2$ given points is known as the Fermat-Weber problem. It has an interesting history dating back to the 17th century, and beautiful exact geometric solution methods for $n = 3$ [4]. For $n > 3$, usually, iterative methods based on the convexity of the objective function are applied.

Next, we summarize our results.

The version with distinct grid points is NP-complete, even if only the weights 0 and 1 appear, and every vertex pair with weight 1 must be placed on two neighbored grid points [3]. Actually, this can be rephrased as the problem of embedding a graph into a square grid, such that all graph edges are mapped onto grid edges. We will conclude in Section 3 that the version with arbitrary real coordinates but with required minimum distances and some placed vertices is also NP-complete. More precisely, NP-completeness holds for the decision version of the problem with a given upper limit for the spread.

In Section 4 we give an $O(n^2)$-time algorithm for placing just one new vertex so as to minimize the weighted sum of Manhattan distances to n already placed vertices, under the constraint that it must also keep at least some prescribed distances to all the placed vertices. Without these constraints, the optimal solution would simply be the weighted median, and our algorithm is based on that fact and on the structure of the objective function. This turns out to be quite technical already for $m = 1$. In the light of NP-completeness for general m it may be interesting to study the complexity parameterized by m in future work. There we have the additional difficulty of weights and minimum distance requirements between all pairs of the m unplaced vertices. However, the algorithm for $m = 1$ can yield quick lower bounds in branch-and-bound methods for instances on a grid and with general m. (For example, one lower bound is given by the sum of optimal spreads for all m instances containing all placed vertices and one of the unplaced vertices, plus the sum of the $\binom{m}{2}$ products of weights and required distances between the unplaced vertices. In the case of high weights between unplaced vertices it can be further raised, but we do not go into the details of such bounds.)

In Section 5 we briefly look into the problem for general n and m without minimum distance constraints. We give a structural property that helps limit the optimal solutions. Again, this can be used for lower bounds on the spread in branch-and-bound algorithms. For the case $n = 2$ we also mention a neat connection to minimum cuts in graphs.

2. Notation

Input to our problems is the following structure that we call a *partially embedded edge-weighted clique with safety distances*, admittedly a bulky name, but with the advantage of being specific. It consists of the following elements:

We have an underlying metric space. (While we will only consider the plane with either Manhattan or Euclidean metric, this could in principle be any metric space.) We are also given $n + m$ vertices, n of which are *placed*, and m are *unplaced*. Being placed means that the vertex is already located at some point in the metric space. Pairs of vertices, also called edges, have non-negative real weights. Some safety distances (that may also be zero) are also required between pairs of vertices. Items are denoted by these names:

$\mathcal{P} = \{P_1, \dots, P_n\}$: a set of placed vertices.
$\mathcal{Q} = \{Q_1, \dots, Q_m\}$: a set of unplaced vertices.
w_{ij}: the given weight of the pair (P_i, Q_j).
u_{ij}: the given weight of the pair (Q_i, Q_j).
r_{ij}: the required minimum distance between P_i and Q_j in a placement.
s_{ij}: the required minimum distance between Q_i and Q_j in a placement.
d_{ij}: the distance between P_i and Q_j in a placement.
e_{ij}: the distance between Q_i and Q_j in a placement.

Note that some of these terms are symmetric ($u_{ij} = u_{ji}$, $e_{ij} = e_{ji}$, $s_{ij} = s_{ji}$), and that the d_{ij} and e_{ij} are not part of a problem instance, as they depend on the placement of the vertices $Q_1, \dots, Q_m$ on points of the metric space.

With these notations, we define the *spread* of such a placement as

$$\sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} d_{ij} + \sum_{1 \le i < j \le m} u_{ij} e_{ij}.$$

We wish to minimize the spread under the constraints $d_{ij} \ge r_{ij}$ and $e_{ij} \ge s_{ij}$ for all indices.

We remark that the pairs of placed vertices may have weights as well, and their weighted distances may also be counted in the spread, but since their distances are already fixed, we can disregard this constant part of the spread.

When only one vertex is unplaced ($m = 1$), we will simplify the variable names and suppress the index j and some variables that become meaningless.

For clearer reference we give the problem a name:

MinSpread
Given: a partially embedded edge-weighted clique with safety distances.
Find: a placement of the unplaced vertices that minimizes the spread, under the constraint that all pairs of vertices respect their required minimum distances.

A *grid point* in the plane with Cartesian coordinate system is a point with integer coordinates. The following problem is NP-complete; see e.g. [3] for further complexity results for special cases.

GridGraph
Given: an undirected graph.
Find: a placement of the vertices on distinct grid points such that every pair of adjacent vertices is placed on neighbored grid points, i.e., grid points with distance 1 (or report that no such placement exists).

3. A hardness result

Theorem 1. *MinSpread is NP-complete in the plane with Manhattan distance.*

Proof. We give a polynomial-time reduction from GridGraph to MinSpread. Consider an instance of GridGraph, that is, graph G with m vertices and k edges. If a placement as required exists, then it fits in an $m \times m$ grid, since rows or columns not containing any vertices can be removed without violating the condition.

The Manhattan distance between two points with Cartesian coordinates (x, y) and (x', y') is $|x - x'| + |y - y'|$. From G we construct an instance of MinSpread as follows. The vertex set of G becomes Q, that is, for every vertex of G we create an unplaced vertex. We fix some $(m+2) \times (m+2)$ grid S of points with integer coordinates. In the convex hull of S, we take every point with coordinates $(x + 1/2, y + 1/2)$ where x and y are integers, and place a newly created vertex there (which does not belong to G). These placed vertices form an $(m+1) \times (m+1)$ grid S', that is, $\mathcal{P} = S'$. This grid has four corner vertices, in an obvious sense. The weight of any vertex pair is defined as follows:

1, if both vertices are unplaced and joined by an edge;
k, if one vertex is unplaced, and the other one is a corner vertex in S';
0, for all other pairs.

The idea of these high edge weights k is that they will force the vertices of Q to be in the fixed finite grid, in any placement with a small spread.

We set the minimum distance between any placed and unplaced vertex to 1. The minimum distances of all other vertex pairs are set to some small positive number δ, just to make sure that all vertices of Q will be mapped onto n distinct points. The construction is obviously done in polynomial time.

Let S'' denote the $m \times m$ grid consisting of the inner grid points of S. Note that the points in the convex hull of S' being able to host unplaced vertices are exactly the points of S''. Now we show equivalence of the instances.

If G has an embedding into the grid, then our instance has a solution with spread $4m^2k + k$. Indeed, we put the vertices in S'' precisely as in the given embedding. It fits in this $m \times m$ grid, due to the remark in the beginning. The weighted sum of distances of every vertex of G to the corner vertices is exactly $4mk$, regardless of the point where we placed it, the second factor m comes from the presence of m vertices, and the last term k is the spread of the embedding of G itself.

Conversely, assume that our instance has a solution with spread $4m^2k + k$. If some vertex of G is outside the convex hull of S', then its weighted sum of distances to the corner vertices is at least $4mk + 2k$. The extra term $2k = 4(1/2)k$ must be added because the vertex is at last $1/2$ length unit away from the border of the convex hull of S', in order to keep distances at least 1 to all points of S'. Thus, already because of that, the total spread is at least $4m^2k + 2k$, a contradiction. This shows that all vertices of G are in the convex hull of S', hence in S''. Since the weighted sum of distances of all vertices in G to the corner vertices is now exactly $4m^2k$, the embedding of G in S'' has spread k, meaning that all edges of G have length 1. □

4. Placing one vertex in the plane with Manhattan metric and minimum distance constraints

4.1. The problem and its restriction to the quadrants

Let now $P_1, \ldots, P_n$ denote the placed vertices, and Q another, unplaced vertex. Let $w_i > 0$ denote the weight of the pair (Q, P_i), and $r_i \geq 0$ the required minimum distance of Q and P_i. For a placement of vertex Q, let d_i denote the resulting Manhattan distance between Q and P_i. The MinSpread problem now asks to place Q so as to minimize $f(Q) := \sum_{i=1}^{n} w_i d_i$ under the constraints $d_i \geq r_i$ for all i. We do not strictly distinguish between a vertex and the point where it is placed, and we denote the vertices or points also by their coordinates: Let $P_i = (x_i, y_i)$ and $Q = (x, y)$. We use $f(x, y)$ as a shorthand for $f((x, y))$.

If $r_i = 0$ and $y_i = 0$ for all i, then f restricted to the x-axis has the form $f(Q) = \sum_{i=1}^{n} w_i d_i = \sum_{i=1}^{n} w_i \cdot |x_i - x|$ and is a convex function of x. Hence it attains its minimum on some interval, which can also be a single point. Optimal points $Q = (x, 0)$ are characterized by the property that $\sum_{x_i < x} w_i \cdot (x - x_i) = \sum_{x_i > x} w_i \cdot (x_i - x)$. We pick any such optimal point and call it a *weighted median*. It can be straightforwardly computed in $O(n)$ time by moving two points p and q inwards, starting at the leftmost and rightmost placed vertex, and keeping $\sum_{x_i < p} w_i \cdot (p - x_i) = \sum_{x_i > q} w_i \cdot (x_i - q)$, until p and q meet.

If still $r_i = 0$ for all i, but any (unrestricted) point set is given, then the expression $f(Q) = \sum_{i=1}^{n} w_i \cdot |x_i - x| + \sum_{i=1}^{n} w_i \cdot |y_i - y|$ consists of two independent terms, thus an optimal point (x, y) can be found by combining the weighted medians of the projections of the given point set to the x-axis and y-axis. We pick any such point and call it a *2D weighted median*. Afterwards we shift the coordinate system such that the chosen 2D weighted median becomes the origin $(0, 0)$. This entire preprocessing needs $O(n)$ time.

From now on we consider the general case with non-negative values r_i. We assume that our 2D weighted median is already in the origin, and we want to find an optimal point Q in the non-negative quadrant, i.e., the one with $x \geq 0$ and $y \geq 0$. We similarly treat all

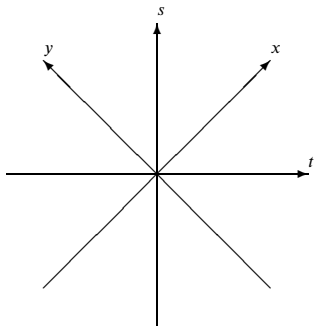

Fig. 1. Orientation of our coordinate axes.

four quadrants and finally take the overall optimum. In the following, all statements therefore refer only to the MinSpread problem and the function f restricted to the non-negative quadrant.

4.2. Preparations

First, we study some basic properties of f in the non-negative quadrant.

Lemma 2. *After an $O(n \log n)$ time preprocessing, we can compute $f(x, y)$ for every requested x and y in $O(\log n)$ time.*

Proof. Let $g(x) = \sum_{i=1}^{n} w_i \cdot |x_i - x|$ and $h(y) = \sum_{i=1}^{n} w_i \cdot |y_i - y|$. We describe the preprocessing for computations of the values of g; the procedure for h is then similar. Function g is piecewise linear and continuous, and it changes its slope only at points with coordinates x_i. We call them special points in the following. We sort the special points in ascending order of their x-coordinates, in $O(n \log n)$ time. The slope of g at any x equals the difference of the sums of weights of all special points to the left and to the right of x. Thus we obtain all values $g(x_i)$ in $O(n)$ time, by going through the sorted list of the special points, updating the slope, and adding to the previous function value the product of slope and distance (in x-direction) to the previous special point. In order to obtain $g(x)$ for any given x which is between special points, we find the two neighbored special points by binary search in $O(\log n)$ time, and do linear interpolation. Since $f(x, y) = g(x) + h(y)$, the claimed time bounds follow. □

Lemma 3. *In the non-negative quadrant, the function f is monotone increasing in both x and y.*

Proof. This follows instantly from the origin being the 2D weighted median. □

We define $s := x + y$ and $t := x - y$ and mainly work in the t-s-coordinate system henceforth, with the understanding that the t-axis goes to the right, and the s-axis goes upwards. The non-negative quadrant is now bounded from below by the rays $s = -t$ for $t \le 0$, and $s = t$ for $t \ge 0$, both starting in the point with $t = 0$ and $s = 0$. The y-axis and x-axis go diagonally upwards. See Fig. 1.

Lemma 4. *For every fixed value of s, the function f is convex in t. Furthermore, f is piecewise linear and can change its derivative only at points with coordinates $x = x_i$ or $y = y_i$, hence only $O(n)$ times.*

Proof. For any fixed index i, let $f_i(Q) = f_i(x, y) := w_i \cdot |x_i - x| + w_i \cdot |y_i - y|$, and let Q_x and Q_y be the two points where the lines $x = x_i$ and $y = y_i$, respectively, intersect the line $x + y = s$. (Notice that $Q_x = Q_y$ if $x_i + y_i = s$, and that Q_x or Q_y may be outside the non-negative quadrant.) When we move our point $Q = (x, y)$ along the line $x + y = s$, then f_i is constant between Q_x and Q_y, and f_i increases linearly outside this interval when we move Q away from it. In particular, this means that f_i is a convex function on the line $x + y = s$. Since a sum of convex functions is convex, the convexity of f follows. The other assertions are obvious from the definition of f. □

In the following, the *rhomb* with center Q and radius r means the set of all points at distances strictly less than r from Q. Note that a rhomb is always bounded by four sides of equal length, being parallel to the s-axis or t-axis, and the sides are not part of the rhomb, i.e., it is topologically open. Every constraint $d_i \ge r_i$ forbids the points in the rhomb with center (x_i, y_i) and radius r_i. Furthermore, geometric terminology refers to the t-s-coordinate system.

Lemma 5. *There exists some point Q minimizing $f(Q)$ under the constraints ($d_i \ge r_i$ for all i), which is either located on the upper side of some rhomb, or is the intersection of the y-axis (x-axis) with the left (right) side of some rhomb, or is the point $(0, 0)$.*

Proof. Let Q be any point that minimizes $f(Q)$ among all points not being in any rhomb. We move Q downwards, thus keeping t fixed and decreasing s. Since x and y decrease, $f(Q)$ can only decrease, due to Lemma 3. But since the original $f(Q)$ was already minimal, $f(Q)$ remains constant. The movement stops when Q reaches an upper side of some rhomb or the y-axis or x-axis. In the former case we are done. Now, let Q be on the y-axis (the case of the x-axis is similar). We move Q further diagonally downwards on the y-axis. Since y decreases and x remains fixed, we conclude as above that $f(Q)$ remains constant. This movement stops only when Q reaches the upper or left side of some rhomb or the point $(0,0)$. □

4.3. The algorithm

Theorem 6. *A point that minimizes the weighted sum of Manhattan distances to n given points in the plane, and keeps at least some prescribed Manhattan distance to each of these points, can be computed in $O(n^2)$ time. In other words, MinSpread for $m = 1$ in the plane with Manhattan metric is solvable in $O(n^2)$ time.*

Proof. As outlined earlier, we can restrict the problem to the non-negative quadrant. We call a point *valid* if it satisfies the constraints, i.e., is not located in a rhomb. Due to Lemma 5, it suffices to do the following:

(0) Check whether $(0,0)$ is valid, and if so, evaluate $f(0,0)$.
(1) For every intersection point Q of the y-axis (x-axis) with the left (right) side of some rhomb, check whether Q is valid, and if so, evaluate $f(Q)$.
(2) For every rhomb, find a valid point on its upper side that minimizes f among all valid points (or report that no valid point exists there).
(3) Finally, take the minimum of all these values of f.

Actually, we need not evaluate all these points and can instead speed up (3) by using the monotonicity properties again, but this would not improve the overall time bound which is dominated by (2), therefore, we have chosen this simplified description.

Every single function value can be computed in $O(\log n)$ time by Lemma 2. Hence we get all function values in (0) and (1) in $O(n \log n)$ time, including the preprocessing time from Lemma 2. Checking the validity of the $O(n)$ candidate points in a naive way takes $O(n^2)$ time.

Next, we go into the details of step (2). In the beginning we sort the values x_i and the values y_i, respectively, in $O(n \log n)$ time. The lines $x = x_i$ and $y = y_i$, respectively, intersect every horizontal line in the same ordering. We call these intersection points x-crossings and y-crossings, respectively, or simply crossings.

From Lemma 4 we know that f restricted to any horizontal line, e.g., to the upper side U of some rhomb, is a convex and piecewise linear function changing its derivative only at points with coordinates $x = x_i$ or $y = y_i$, that is, at crossings. Hence f attains its minimum on U either at some unique crossing c, or at two consecutive crossings a and b (a being to the left of b) and all points in between. In the former case we define $a := c$ and $b := c$. We can determine a and b in $O(\log^2 n)$ time, using Lemma 2 and unimodal search on the fixed orderings of x-crossings and y-crossings.

However, we have to find some optimal point on U among all valid points. Since f restricted to U is a convex function of t, it is monotone non-increasing to the left of b and monotone non-decreasing to the right of a. Thus, it suffices to determine the rightmost valid point b' not being to the right of b, and the leftmost valid point a' not being to the left of a. The better of these two points is an optimal point. Below we treat a', as the case of b' is symmetric.

We use a list of rhombs sorted by the t-coordinates of their left sides. We produce the list of their intersections with U (which are open intervals), already sorted by their left endpoints. Rhombs with empty intersection with U are ignored for now. The sorted list of rhombs can be produced once in the beginning in $O(n \log n)$ time. All their intersections with U are computed in $O(n)$ time, essentially by comparing some s-coordinates. Hence, this takes $O(n^2)$ time for all upper sides U of all rhombs together.

To find a' we proceed as follows. First let $a' := a$. We scan the mentioned sorted list of intervals (l, r). If $r \le a'$, we go to the next interval. If $l < a' < r$, we set $a' := r$ and go to the next interval. If $a' \le l$, we stop and return a'. We also stop if we get beyond the right end of U, and then we return that a' does not exist on U. Correctness is evident, and the procedure takes $O(n)$ time. □

5. Several unplaced vertices

Recall the notation from Section 2. In this section we consider MinSpread when $r_{ij} = 0$ holds for all indices. This is the problem of minimizing the spread when several unplaced vertices exist, but no minimum distance constraints apply. In particular, vertices are even permitted to be mapped to the same point. (But the placed vertices $P_1, \ldots, P_n$ can be assumed to occupy n distinct points, otherwise we could merge the vertices located at the same point and sum up the weights of their pairs with other vertices.)

First we formulate a lemma for the 1-dimensional version of this problem, that is, the metric space is a straight line. Euclidean and Manhattan distance are identical there. Afterwards we draw some conclusions for the plane equipped with one of these metrics.

Lemma 7. *On the line, there exists an optimal placement where every vertex Q_j is placed at some point already occupied by some vertex P_i.*

Proof. In the following, a cluster means any subset of vertices that are mapped to the same point. In particular, the vertices P_i belong to n distinct clusters. Given a placement, we pick any cluster C that only contains vertices Q_j. It divides the set of all other vertices in

two subsets that we call sides, namely, the vertices to the left and to the right of C. For each side we determine the sum of weights of all edges that join vertices of C with vertices on that side. We move C towards the side with the larger sum; if both sums are equal, then either side can be chosen. The move stops as soon as C collides with another cluster. Hence, a move has two effects: it decreases the spread and strictly decreases the number of clusters. Thus, iterating this procedure eventually implies the assertion. □

It remains to actually find an optimal placement of the form indicated by Lemma 7. Interestingly, in the case $n = 2$, these optimal layouts can be nicely characterized by minimum cuts. By a *cut* we mean any partitioning of the vertex set into two disjoint subsets, called the *sides* of the cut, such that P_1 and P_2 are on different sides. Every edge having its vertices on different sides is called a *cut edge*. The *capacity* of a cut is the sum of weights of all cut edges. A *minimum cut* is a cut of minimum capacity. The following is an immediate consequence of Lemma 7, in the special case $n = 2$:

Proposition 8. *In the case of exactly two placed vertices P_1 and P_2, an optimal placement on the line is obtained by computing a minimum cut and then mapping all vertices Q_j from the two sides to the points of P_1 and P_2, respectively.*

Next we use these observations to obtain a simple lower bound for the 2-dimensional Euclidean problem.

Proposition 9. *In the case of exactly two placed vertices P_1 and P_2 in the plane, the minimum cut capacity multiplied by the Euclidean distance between P_1 and P_2 is a lower bound on the spread in the Euclidean metric.*

Proof. For any given layout, projecting all vertices Q_j orthogonally to the straight line through P_1 and P_2 can only decrease the spread. Thus, Lemma 8 yields the assertion. □

For the Manhattan metric and general n we obtain:

Proposition 10. *In the plane with Manhattan metric, every instance of MinSpread has some optimal solution where every vertex Q_j ends up at some point whose coordinates appear among the coordinates of some placed vertices P_i.*

Proof. Since the weighted Manhattan distance is the sum of the weighted coordinate differences in x-direction and y-direction, we can consider the projections of points to both coordinate axes separately, and apply Lemma 7 twice. □

For finding an optimal layout due to Proposition 10, it still remains to find the optimal placement of m vertices Q_j on n^2 candidate points. However, for a small number of vertices (e.g., for the use as a partition-based lower bound in branch-and-bound), we can afford exhaustive search on these candidate points.

6. Further research

It may be possible to improve the $O(n^2)$ time bound in Theorem 6. The bottleneck is to check the membership of points in rhombs, where some sweep line technique might beat the quadratic time.

Applying Lemma 7 to more than two placed vertices would require the computation of certain multiway cuts, and in Proposition 10, it should be possible to avoid exhaustive search on the grid of the n^2 candidate points for the yet unplaced vertices, by using the geometry of this grid.

MinSpread with different metrics for the spread and for the safety distances could also be of interest for more accurate modeling: For instance, the Manhattan metric can be appropriate for the movements of workers and material, whereas noise and chemicals do not go around corners but they are propagated according to the Euclidean metric.

CRediT authorship contribution statement

Peter Damaschke: Writing – review & editing, Writing – original draft, Methodology, Investigation, Formal analysis, Conceptualization. **Fredrik Ekstedt:** Writing – review & editing, Conceptualization. **Raad Salman:** Writing – review & editing, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work of the second and third author has benefited from funding from the Swedish Innovation Agency Vinnova, as part of the Artificial Intelligence supported Tool Chain in Manufacturing Engineering (AITOC) research project, grant 2020-01947, within the ITEA3 cluster.

Data availability

No data was used for the research described in the article.

References

[1] P. Mårdberg, J. Fredby, K. Engström, Y. Li, R. Bohlin, J. Berglund, J.S. Carlson, J. Vallhagen, A Novel Tool for Optimization and Verification of Layout and Human Logistics in Digital Factories, Proc. CIRP 72 (2018) 545–550.
[2] R. Muther, J.D. Wheeler, Simplified Systematic Layout Planning. Management and Industr, Res. Publ., Univ. of Michigan, 1994.
[3] V.G.P. de Sá, G.D. da Fonseca, R. Machado, C.M.H. de Figueiredo, Complexity Dichotomy on Partial Grid Recognition, Theor. Comput. Sci. 412 (2011) 2370–2379.
[4] L.N. Tellier, The Weber Problem: Solution and Interpretation, Geogr. Anal. 4 (1972) 215–233.