

Recovering Graphs from Their Witness Unit Square Representation

Maarten Löffler  

Department of Information and Computing Sciences, Utrecht University, The Netherlands

Frank Staals  

Department of Information and Computing Sciences, Utrecht University, The Netherlands

Soeren Terziadis  

Algorithms Group, TU Eindhoven, The Netherlands

Abstract

A w USR of a graph G is a set of unit squares in the plane, one per vertex, if two vertices have an edge in G if their squares overlap and the overlap contains no witness. We present an output sensitive algorithm to compute a graph G based on its given witness unit square representation.

2012 ACM Subject Classification Mathematics of computing → Graph theory

Keywords and phrases proximity graphs, geometric intersection graphs, witness representation, unit square intersection graph, output sensitive algorithm, range searching

Digital Object Identifier 10.4230/LIPIcs.GD.2025.44

Category Poster Abstract

Funding *Soeren Terziadis*: funded by the NWO Gravitation project NETWORKS (no. 024.002.003.)

1 Introduction

Witness representations of graphs are a variant of geometric intersection representations of graphs, where we are given a set of convex objects and a set of points (*witnesses*). Every vertex is represented by one object and two vertices are connected if they intersect and their intersection contains no witness. Given a witness representation of n objects and m witnesses, we want to compute (*recover*) the graph it represents. Explicitly computing the intersection graph of all n objects which intersect at most $k \in O(n^2)$ times and checking for every intersection if it contains a witness requires $O(n^2 \cdot m)$ time. If the intersection graph is sparse, then the runtime can be reduced to $O(n \log m)$. However if there are $O(n^2)$ intersections between objects but the witness intersection graph has only $k \in O(n)$ edges, the naïve method would still require $O(n^2 \cdot m)$ time. Our goal is to reduce this to $\tilde{O}(n + m + k)$ by using an output sensitive algorithm. Towards this goal, we consider here the square version.

A witness unit square representation (w USR) – also shown in Figure 1a) – is a tuple $(\mathcal{S}, \mathcal{W})$ where \mathcal{S} is a set of n axis-aligned unit squares and \mathcal{W} is a set of m point features (*witnesses*). The witness unit square graph $G(R) = \{\mathcal{S}, E\}$ is shown in Figure 1b). If $\{S_i, S_j\}$ intersect and their intersection contains a witness W , the edge $\{S_i, S_j\}$ is *blocked* by W .

2 Initial Observations

To solve the problem we consider the dual of the representation, which represents any square in \mathcal{S} with its center point and every witness with a unit square centered at the witness (Figure 1c). The *large square* B_i of S is a square of side length 2 centered at P_i .



© Maarten Löffler, Frank Staals, and Soeren Terziadis;
licensed under Creative Commons License CC-BY 4.0

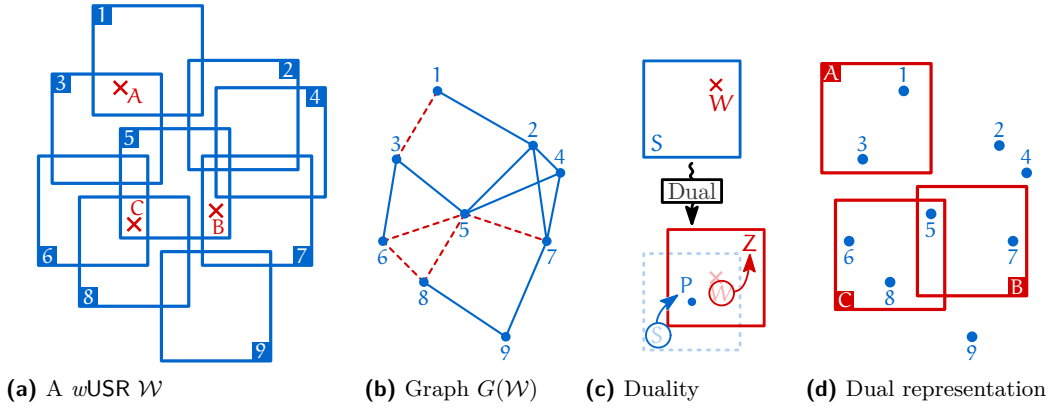
33rd International Symposium on Graph Drawing and Network Visualization (GD 2025).

Editors: Vida Dujmović and Fabrizio Montecchiani; Article No. 44; pp. 44:1–44:4

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 1** A $wUSR$ \mathcal{W} (a), its graph (b), the dual construction (c) and the dual of \mathcal{W} (d).

► **Observation 1.** Let $S_i, S_j \in \mathcal{S}$ be two intersecting squares and let $W_k \in \mathcal{W}$ be a witness such that $W_k \in S_i \cap S_j$. Then $P_i \in Z_k$ and $P_j \in Z_k$. Or in other words, if a witness is contained in the intersection of two squares, the dual points of both squares are contained in the dual square of the witness.

► **Observation 2.** Let $S_i, S_j \in \mathcal{S}$ be two intersecting squares. Then $P_j \in B_i$. Or in other words, if two squares intersect, then the dual point of one is in the large square of the other.

► **Observation 3.** Let $S_i \in \mathcal{S}$ be a squares and let $W_j \in \mathcal{W}$ be a witness such that $W_j \in S_i$. Then $Z_j \subset B_i$. Or in other words, if a square contains a witness, the large square of the square contains the dual square of the witness.

Intuitively the previous observations state that everything relevant for a dual point of a square S is contained in its large square. We assume general position. The edges of $G(R)$ are partitioned into a set with positive slope (*right* edges) and negative slope (*left* edges). For the remainder of this paper we focus on right edges; left edges are handled symmetrically.

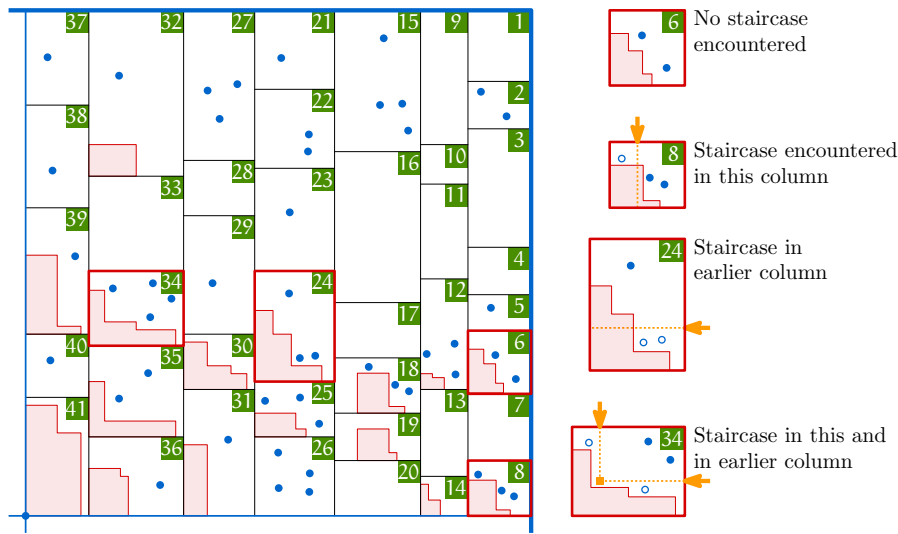
Consider the top-right quadrant B^\nearrow of a large square B of \mathcal{S} . For every right edge $\{S, S'\}$, with S lower than S' , the dual point P' of S' is in B^\nearrow . For every witness W that is contained within S , we know by Observation 3 that it's dual square $Z \subset B$ and in particular the top-right corner of Z is contained in B^\nearrow . Let \mathcal{T} be the set of top-right corners of witness dual squares that are contained in B^\nearrow . We say a point $A \in B^\nearrow$ dominates a point $A' \in B^\nearrow$ if A is higher and further right than A' .

▷ **Claim 4.** Let P, P' be two dual points of the squares S and S' , respectively. Let B^\nearrow be the top-right quadrant of the large square of S . The edge $\{P, P'\}$ is a right edge in $G(R)$, if and only if $P' \in B^\nearrow$ and there is no top right corner $T \in \mathcal{T} \cap B^\nearrow$, s.t., T dominates P' .

Proof. By Observation 2 S and S' overlap if and only if $P \in B^\nearrow$. By Observation 3 any witness contained in S has a dual square whose top right corner is contained in B^\nearrow , therefore no corners outside of B^\nearrow have to be considered. Finally by Observation 1 any witness contained in both S and S' would have a top-right corner that dominates P' ◁

3 Recovery Algorithm

We build a two dimensional range tree on all points in the set \mathcal{P} , i.e., all dual points of squares in \mathcal{S} . Every secondary level node N corresponds to some region R_N , a (canonical) subset of points $P_N \subseteq \mathcal{P}$, and a subset $T_N \subseteq \mathcal{T}$. We order T_N to form a staircase. Let P'_N be



■ **Figure 2** Query result as partition of query square with dual points (blue) and staircases of witness square corners (red). Cells are processed as numbered. Four cases for processing the cells are shown on the right.

the subset of P_N not dominated by a point in T_N , i.e. P'_N is the subset above the staircase. We store P'_N in a priority search tree [1] enabling efficient reporting of all points in P'_N , above and to the right of a query point. This can be done in $O((n + m) \log(n + m))$ time

For every point $P_i \in \mathcal{P}$ of a square $S_i \in \mathcal{S}$, we query this data-structure with a search window corresponding to B_i^\nearrow of the large square of S_i in $O(\log^2 n)$. The query will report a number of subsets of \mathcal{P} . The returned sets can be interpreted as a partition of the search quadrant into columns, which are again subdivided into rows (forming *cells*); see Figure 2. The cells are processed column by column, right to left and top to bottom within a column. When processing these cells, we track \hat{x} and \hat{y} , i.e., the highest x and y coordinates over all staircases and report the canonical subsets without points dominated by (\hat{x}, \hat{y}) . Four cases of previous staircase encounters are distinguished (none, previously this column, previously in earlier column, previously this and earlier column); we refer again to Figure 2.

The last case (at most once per column, i.e., $O(\log n)$ times) requires an additional query in the priority search tree. Since the query square is subdivided into at most $O(\log^2 n)$ cells – let \mathcal{C} be the set of all cells – in the query result of the range query, the entire time requirement for a single query plus processing of the cells is

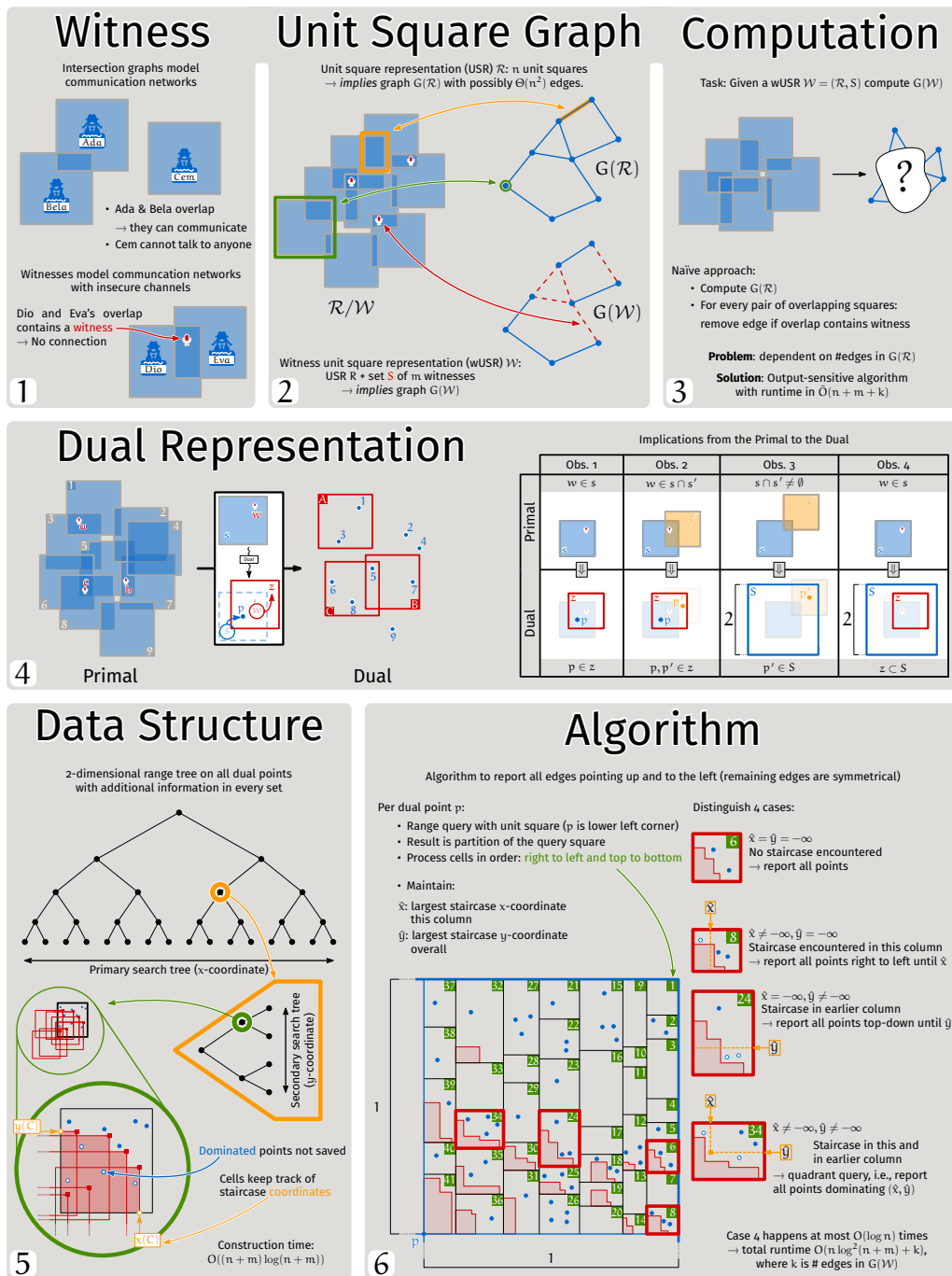
$$O(\log^2 n) + O(\log n \cdot \log(n + m)) + O\left(\sum_{C \in \mathcal{C}} k_C\right) = O(\log^2(n + m) + k_i)$$

since $\sum_{C \in \mathcal{C}} k_C = k_i$, the number of reported edges. We need one query per dual point of a square in S and correctly compute all k right edges in time $O(n \log^2(n + m) + \sum_{i=1}^n k_i)$.

► **Theorem 5.** *Given a wUSR \mathcal{W} with n unit squares and m witnesses, we can compute the implied graph $G(R)$ in $\tilde{O}(n + m + k)$ time, where k is the number of edges in $G(\mathcal{W})$.*

References

1 Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational geometry: Algorithms and applications, 3rd Edition*. Springer, 2008. doi: 10.1007/978-3-540-77974-2.



■ Figure 3 The poster.