A 2D Shortest Superstring (Tile Canvas) Problem: Exact Models and an Ant Colony Optimization Heuristic

Tran Thanh Dat

September 24, 2025

Abstract

We study a 2D analogue of the Shortest Superstring problem. Given a set of T square tiles of size $n \times n$ over a finite alphabet, the task is to place one translated copy of each tile on an integer grid so that overlapping cells agree and the side length m of the bounding canvas is minimized. We present: (i) a precise problem statement; (ii) an exact Mixed-Integer Linear Programming (MILP) model solved iteratively over m; (iii) a complete brute-force backtracking oracle for small instances; and (iv) a practical Ant Colony Optimization (ACO) heuristic with sparse pheromones, overlap-based heuristics, and compactness bias. The formulations match the provided construction paradigm (edges encode relative placements), and the ACO is designed to scale for $n < 10$ and moderate T.

1 Problem definition

Let $\mathcal{T} = \{1, \ldots, T\}$ be tile indices. Each tile $t \in \mathcal{T}$ is an array $A^{(t)} \in \Sigma^{n \times n}$ over a finite alphabet Σ (e.g., $\{0, 1\}$). A placement assigns to each tile t an integer top-left offset $p_t = (x_t, y_t) \in \mathbb{Z}^2$. The induced canvas labeling is

$$C(X, Y) \; = \; A^{(t)}(X - x_t,\, Y - y_t) \quad \text{for any } (X, Y) \text{ covered by tile } t,$$

which must be *well-defined*: whenever two tiles cover the same cell, they must agree: $A^{(u)}(i, j) = A^{(v)}(i + \Delta x, j + \Delta y)$ for all overlapping indices. The (axis-aligned) bounding box of a placement is

$$[X_{\min}, X_{\max}] \times [Y_{\min}, Y_{\max}] \;\text{ where }\; X_{\min} = \min_t x_t,\; X_{\max} = \max_t (x_t + n - 1), \text{ etc.}$$

We define the *canvas side* $m = \max\{X_{\max} - X_{\min} + 1,\; Y_{\max} - Y_{\min} + 1\}$, and aim to minimize m.

Decision Given $m \in \mathbb{Z}_{\geq n}$, does there exist a conflict-free placement with bounding square contained in $[0, m-1]^2$? The optimization task is to find the minimal feasible m.

2 Exact models

2.1 Iterative MILP feasibility (Gurobi)

For a fixed m, each tile must be placed at a top-left integer coordinate $(x, y) \in \{0, \ldots, m - n\}^2$. Introduce binary variables

$$p_{txy} = \begin{cases} 1 & \text{if tile } t \text{ is placed at } (x, y), \\ 0 & \text{otherwise.} \end{cases}$$

Assignment constraints Each tile placed exactly once:

$$\sum_{x=0}^{m-n} \sum_{y=0}^{m-n} p_{txy} = 1 \quad \forall t \in \mathcal{T}. \tag{1}$$

Pairwise conflict constraints For any two tiles $u < v$ and placements (x, y), (x', y'), if the two translated tiles overlap at any cell with different symbols, forbid selecting both:

$$p_{uxy} + p_{vx'y'} \leq 1 \quad \text{for all conflicting pairs.} \tag{2}$$

Conflicts are precomputed in $O(n^2)$ per pair of placements by checking the intersection of their $n \times n$ supports.

The model has $T \cdot (m - n + 1)^2$ binaries. We iterate $m = n, n+1, \ldots, \bar{m}$ (with a simple upper bound like $\bar{m} = n\lceil\sqrt{T}\rceil$) and solve (1)–(2) for feasibility. The first feasible m is optimal.

2.2 Backtracking and Greedy Algorithms

For small n and moderate T, an exact search over placements is practical. For each m from n upward:

(i) Enumerate the domain $\{0, \ldots, m-n\}^2$ for every tile.

(ii) Backtrack with MRV (place the tile with the fewest currently feasible positions), using a hash-based occupancy to test conflicts in $O(n^2)$ per attempt.

(iii) Order candidate positions by descending current overlap with the partial canvas to find solutions quickly.

This exactly matches the feasibility decision and returns optimal m at the first success.

Algorithm 1 Greedy Overlap Insertion (GOI)

Require: Tiles $\mathcal{T}$ ($n \times n$), All feasible offsets Δd for u

$L \leftarrow \{(r, 0, 0)\}$; $P \leftarrow \{r\}$; $U \leftarrow \mathcal{T} \setminus \{r\}$; init BBox
while $U \neq \emptyset$ **do**
    Best $\leftarrow$ None
    for all $v \in U$ **do**
        $\mathcal{C}_v \leftarrow \{(x_u + d_x, y_u + d_y)\}$ from $u \in P$
        if $\mathcal{C}_v = \emptyset$ **then** add perimeter candidates
        end if
        for all $(x, y) \in \mathcal{C}_v$ **do**
            if placing v at (x, y) is conflict-free **then**
                $H \leftarrow \sum_{u \in P} ov\big(u, v, x - x_u, y - y_u\big)$
                $\Delta m \leftarrow$ enlargement if v placed at (x, y)
                update Best by max H, tie-break min Δm
            end if
        end for
    end for
    if Best = None **then break**
    end if
    place Best into L; update BBox; move v^* from U to P
end while
return L, $m(L)$

Algorithm 2 Greedy Lowest Enlargement Insertion (GLEI)

Require: Same inputs as GOI

init L, P, U, BBox as before
while $U \neq \emptyset$ **do**
  $\mathsf{Best} \leftarrow$ None
  for all $v \in U$ **do**
    build $\mathcal{C}_v$ as in GOI
    for all $(x, y) \in \mathcal{C}_v$ **do**
      if conflict-free **then**
        $\Delta m \leftarrow$ enlargement of canvas side
        $H \leftarrow \sum_{u \in P} ov(u, v, x - x_u, y - y_u)$
        update Best by min Δm, tie-break max H
      end if
    end for
  end for
  if Best = None **then break**
  end if
  place Best; update BBox; move v^* from U to P
end while
return L, $m(L)$

3 ACO heuristic (constructive layout)

We cast construction as repeatedly adding an unplaced tile at an absolute coordinate. Fix tile r at $(0, 0)$ to break translation symmetry. Let P be the set of placed tiles with positions $p_u = (x_u, y_u)$ and U the unplaced set.

3.1 Sparse relative moves

For each ordered pair (u, v), we precompute *feasible offsets* $\Delta = (\Delta x, \Delta y)$ with $|\Delta x|, |\Delta y| \leq n - 1$ such that, when v is placed at $p_u + \Delta$, all overlapping cells match. We score each by the consistent overlap count $\mathrm{ov}(u, v, \Delta)$ and retain only the top-K offsets (typically $K \in [8, 64]$) to sparsify the move space. These define a sparse pheromone tensor $\tau[u, v, \Delta]$.

3.2 Heuristic and pheromone aggregation

When considering placing v at absolute position (x, y), each placed $u \in P$ implies a relative offset $\Delta_{u \to v} = (x - x_u,\ y - y_u)$. We define

$$\mathrm{Heuristic}(v, x, y) = \sum_{u \in P} \eta(u, v, \Delta_{u \to v}), \quad \text{where } \eta(u, v, \Delta) = \mathrm{ov}(u, v, \Delta) \text{ if feasible, else } 0,$$

$$\mathrm{Phero}(v, x, y) = \sum_{u \in P} \tau[u, v, \Delta_{u \to v}],$$

with ΔBBox the increase in the bounding box's larger side from adding (v, x, y). The sampling weight is

$$w(v, x, y) \;=\; \big(\mathrm{Phero}(v, x, y)\big)^{\alpha} \cdot \big(\mathrm{Heuristic}(v, x, y) + \epsilon\big)^{\beta}. \tag{3}$$

Optionally, we could multiply the weight with the weighted solution cost increased value $\big(\mathrm{Comp}(v, x, y)\big)^{\gamma}$

3.3 Candidate generation

Rather than scanning all (x, y), we propose a small set per v:

- For each $u \in P$ and each retained $\Delta \in \mathrm{TopK}(u, v)$, propose $(x, y) = p_u + \Delta$.
- Deduplicate and keep only conflict-free positions (checked in $O(n^2)$ via occupancy).
- If a v receives no proposals, add a small set of perimeter positions just outside the current bounding box (allows bridging disconnected components).

This yields $O(|P| \cdot K)$ proposals per v.

3.3.1 Augmenting the candidate offsets with adjacency

For each ordered pair (u, v) we precompute the set of feasible overlap offsets

$$\mathcal{F}(u, v) \subseteq \{(\Delta x, \Delta y) : |\Delta x|, |\Delta y| \leq n - 1\},$$

where placing v at $p_u + \Delta$ yields only consistent symbol matches on the overlap. Each $\Delta \in \mathcal{F}(u, v)$ is scored by its consistent overlap count $\mathrm{ov}(u, v, \Delta)$, and we retain the top–$K$ by this score.

To preserve completeness, we always augment these with all edge-adjacent (non-overlapping) offsets

$$\mathcal{A}_n \;=\; \{(\pm n, t) : t \in [-(n-1), n-1]\} \;\cup\; \{(t, \pm n) : t \in [-(n-1), n-1]\},$$

(optionally also including the four corner-touch offsets $\{(\pm n, \pm n)\}$ if point contacts are allowed). These offsets do not create any overlap, so we set $\mathrm{ov}(u, v, \Delta) = 0$ for $\Delta \in \mathcal{A}_n$.

The final candidate set and pheromone domain for (u, v) is

$$\mathcal{C}(u, v) \;=\; \mathrm{TopK}\big(\mathcal{F}(u, v), K\big) \;\cup\; \mathcal{A}_n, \qquad \text{and} \quad \tau[u, v, \Delta] \text{ is defined only for } \Delta \in \mathcal{C}(u, v).$$

In practice $|\mathcal{A}_n| = 4(2n-1)$ (or $4(2n-1)+4$ with corners), which keeps the move space sparse while ensuring that purely adjacent placements remain available even when K is small. If desired, initialize adjacency pheromones with a mild discount, e.g. $\tau_0^{(\mathrm{adj})} = \alpha\,\tau_0$ with $\alpha \in (0, 1)$, so ants prefer informative overlaps but can still chain components via adjacency when necessary.

3.4 Pheromone updates

We record a parent edge (u^*, v, Δ^*) per placement, where u^* maximizes $\eta(\cdot)$ at the chosen (x, y). After an ant completes a solution with canvas side $m(\mathcal{S})$, we perform evaporation and reinforcement:

$$\tau \;\leftarrow\; (1 - \rho)\,\tau, \qquad \tau[u^*, v, \Delta^*] \;\leftarrow\; \tau[u^*, v, \Delta^*] + \frac{Q}{m(\mathcal{S})}. \tag{4}$$

An additional elitist boost on the global-best layout improves stability.

3.5 Pseudocode

Algorithm 3 Sparse-ACO for 2D Tile Canvas Minimization (with adjacency offsets)

Require: tiles $A^{(t)}$, size n, parameters $K, \alpha, \beta, \gamma, \lambda, \rho, Q, \alpha_{\text{adj}}$

Precompute the adjacency set $\mathcal{A}_n = \{(\pm n, t) : t \in [-(n-1), n-1]\} \cup \{(t, \pm n) : t \in [-(n-1), n-1]\}$ (optionally $\{(\pm n, \pm n)\}$).
For each ordered pair (u, v):
  Compute feasible overlap offsets $\mathcal{F}(u,v) = \{\Delta : |\Delta_x|, |\Delta_y| \le n-1, \text{ overlap matches}\}$ and scores $\text{ov}(u, v, \Delta)$.
  Let $\textsc{TopK}(u,v) = \text{TopK}(\mathcal{F}(u,v), K)$ by ov.
  Define candidate-offset set $\mathcal{C}(u,v) = \textsc{TopK}(u,v) \ \cup \ \mathcal{A}_n$.
Initialize sparse pheromone tensor τ only on $\{(u,v,\Delta) : \Delta \in \mathcal{C}(u,v)\}$ with
  $\tau[u,v,\Delta] \leftarrow \begin{cases} \tau_0 & \Delta \in \textsc{TopK}(u,v), \\ \alpha_{\text{adj}} \cdot \tau_0 & \Delta \in \mathcal{A}_n \end{cases}$ and set $\text{ov}(u,v,\Delta) = 0$ for $\Delta \in \mathcal{A}_n$.
for iteration $= 1..I$ **do**
  for ant $= 1..N$ **do**
    $P \leftarrow \{r\}$ with $p_r \leftarrow (0,0)$; init occupancy; record parent-edges $E \leftarrow \emptyset$.
    while $|P| < T$ **do**
      $\mathcal{C} \leftarrow \emptyset$ ▷ candidate moves
      for each $v \notin P$ **do**
        Propose positions from $\{\, p_u + \Delta : \ u \in P, \ \Delta \in \mathcal{C}(u,v) \,\}$.
        for each feasible placement (x, y) for v (no conflicts) **do**
          For this (v, x, y), let $\Delta_{u \to v} = (x, y) - p_u$ for each $u \in P$ with $\Delta_{u \to v} \in \mathcal{C}(u,v)$.
          Compute heuristic and pheromone aggregates

$$H = \sum_{u \in P} \eta(u, v, \Delta_{u \to v}), \qquad T = \sum_{u \in P} \tau[u, v, \Delta_{u \to v}].$$

          Add (v, x, y) to $\mathcal{C}$ with weight $w = (T^{\alpha})(H + \epsilon)^{\beta}$.
        end for
      end for
      Sample one $(v^*, x^*, y^*) \in \mathcal{C}$ by normalized weights; place v^*; update occupancy and bbox.
    end while
    $m \leftarrow$ final canvas side; evaporate $\tau \leftarrow (1-\rho)\tau$; reinforce edges in E by Q/m.
  end for
  Optionally reinforce global-best edges (elitist).
end for

3.6 Complexity and practical settings

Per construction step, candidate generation is $O(|P| \cdot K)$; each feasibility check is $O(n^2)$, so overall roughly $O(T \cdot |P| \cdot K \cdot n^2)$ per ant per iteration (typically modest for $n < 10$). Recommended defaults: $K = 16$–32, $\alpha = 1$, $\beta \in [2, 4]$, $\gamma = 1$, $\lambda \in [0.01, 0.05]$, $\rho = 0.1$, $Q \approx n^2$.

4 Implementation notes

Exact solvers. The MILP feasibility model (Section 2.1) solves to optimality by increasing m. A backtracking oracle (Section 2.2) provides an independent ground-truth for very small instances.

Heuristic. The ACO uses a sparse pheromone tensor on only the best pairwise offsets, drastically shrinking the move space. Perimeter fallback proposals let ants connect components even when pairwise overlaps offer no guidance.

5 Summary

We formalized the 2D tile canvas minimization problem, presented two exact methods (MILP and backtracking) for ground-truth, and detailed a scalable ACO with strong overlap-aware heuristics and compactness bias.