13th International Conference on Applied Informatics
Eszterházy Károly Catholic University
Eger, Hungary, February 19–21, 2026

Experimental Study of Building a Nest by an Automaton (Abstract)

Tamás Lukovszki, Gábor Pusztai[a]

[a]Eötvös Loránd University (ELTE), Faculty of Informatics, Hungary
lukovszki@inf.elte.hu, pusztaigabor@inf.elte.hu

Introduction

In this work, we empirically investigate the construction of a nest by a finite-state robot, based on the work of Czyzowicz et al. [1, 2]. In the nest-building problem, a robot modeled as a deterministic finite automaton must organize scattered objects ("bricks") into a compact **nest** (a connected structure of a given shape).

We describe a set of practical modifications of the original algorithm in [1, 2] leading to a constant-factor improvement of the running time of the nest-building. The work also presents the Java-based interactive simulation environment ("Nest-Builder simulator") that makes the algorithm observable step-by-step and enables controlled experimentation across design variants.

Model

The robot is modeled as a deterministic Mealy automaton navigating an infinite oriented $\mathbb{Z} \times \mathbb{Z}$ grid. The environment consists of *full* cells containing "bricks" and *empty* cells. The subgraph of the grid induced by full cells, called the *shape*. Initially, the shape is connected. Any maximal connected subgraph of the current shape is called a component. The number of cells of a shape is called its *size*, and the (Manhattan-) distance between the two farthest cells of a shape is called its *span*. A *nest* of size z is a shape that has the minimum span among all shapes of size z. The robot can perceive the state of the cells, but has no prior knowledge of the initial shape, its span, or the number of bricks. The goal of the robot is to construct a nest from the bricks.

The core algorithm of [2] operates in a four-phase cycle:

1. Moving bricks out of the way (SWEEP): Clearing bricks within a bounded distance around the nest to maintain a structured gap.
2. Finding the next brick (FINDNEXTBRICK): Utilizing *search walks* (W), *switches*, and *shifting* to retrieve a "free brick" while preserving connectivity of the components.
3. Back to the marker (RETURNTOMARKER): Backtracking along the *reversal* of the search walk ($\varphi(W)$) to return to the marker.
4. Extending the rough disc (EXTENDROUGHDISC): Adding the retrieved brick to the growing nest following a defined geometric progression.

In [2], the authors show that this algorithm can be implemented with a finite-state robot that builds a nest in $O(sz)$ time steps, where s is the initial span and z is the number of bricks. They also show that this complexity is optimal.

Implementation and Practical Refinements

We propose and evaluate several optimizations to critical phases.

The NestBuilder application manages the infinite grid using a dynamic $n \times n$ collection that expands as the robot explores new coordinates. A mediator class translates between the global coordinate system and the robot's local orientation (North, East, South, West). We introduced several optimizations to improve operational efficiency:

- Distance-Aware Sweeping: The theoretical model instructs the robot to "sweep" any brick c found within a distance of 7 during its counterclockwise traversal of the nest border. However, implementation showed the robot might pick up and immediately replace a brick already at an acceptable distance. By verifying the distance between the target brick and the nest prior to movement, we can reduce the average moves in SWEEP.
- State-Based Component Discovery: Due to limited visibility (range of 8), components on the opposite side of a large nest may be missed. We expanded the automaton to store a constant number of bits to record if any bricks were seen during boundary traversal, which reduces the "fault ratio" of left-behind bricks.
- Optimized Rough Disc Extension: Placing bricks to maintain the strict shape of the nest requires finding specific locations through in EXTENDROUGHDISC. We evaluated four search heuristics: Naive Search: Basic search starting from the easternmost cell. Rotation-Aware: Accounting for cases where the robot arrives specifically at the first-placed nest cell to avoid redundant rotations. Direct Detection: Identifying the placement position based on the configuration of adjacent bricks and direction changes. Combined Heuristic: A hybrid approach that selects the optimal path based on the robot's approach vector.

- Search-Walk Robustness: The logic was refined to handle "dead-ends" where directions cannot be decided, such as bricks appearing only in a straight line. In such cases, the robot treats the path as a segment and reverses direction to ensure the shifting procedure remains feasible.

Experimental Evaluation and Conclusion

Testing was conducted on 100 randomly generated connected fields using Kruskal's algorithm to ensure initial connectivity.

- Distance-Aware Sweeping: The distance-aware optimization yielded a $\approx 92\%$ reduction in movement overhead compared to the naive Implementation, dropping average moves from 1082 to 83.
- State-Based Component Discovery: The state-based component discovery significantly improved reliability. The naive method left an average of ≈ 30 bricks unconnected (failed to build a complete nest), whereas the optimized method achieved near-perfect completion (average ≈ 0.01 errors).
- Optimized Rough Disc Extension: The Combined Heuristic for nest extension outperformed the basic iterative search by approximately 23% in terms of movement steps.

Finally, our experiments confirmed that this automaton model is not suitable for parallelization with multiple robots. Multiple agents would inadvertently disconnect each other's components or destroy the marker-based backtracking system. Future research could explore the impact of increased memory to facilitate multi-robot coordination and further accelerate construction.

References

[1] J. Czyzowicz, D. Dereniowski, A. Pelc: *Building a Nest by an Automaton*, in: 27th Annual European Symposium on Algorithms, ESA 2019, 2019, 35:1–35:14.

[2] J. Czyzowicz, D. Dereniowski, A. Pelc: *Building a Nest by an Automaton*, Algorithmica 83.1 (2021), pp. 144–176, DOI: 10.1007/S00453-020-00752-0.