Encyclopedia of Algorithms
DOI 10.1007/978-3-642-27848-8_660-1

Self-Assembly of Squares and Scaled Shapes

Robert Schweller*
Department of Computer Science, University of Texas Rio Grande Valley, Edinburg, TX, USA

Keywords Self-assembly • Tile assembly model • Algorithmic self-assembly • Kolmogorov complexity • Tile complexity • Scaled shapes

Years and Authors of Summarized Original Work

2000; Rothemund, Winfree
2001; Adleman, Cheng, Goel, Huang
2005; Cheng, Aggarwal, Goldwasser, Kao, Schweller, Espanes
2007; Soloveichik, Winfree

Problem Definition

Abstract Tile Assembly Model

The abstract Tile Assembly Model (aTAM) [3] is a mathematical model of self-assembly in which system components are four-sided Wang tiles with glue types assigned to each tile edge. Any pair of glue types are assigned some nonnegative interaction strength denoting how strongly the pair of glues bind. An aTAM system is an ordered triplet (T, τ, σ) consisting of a set of tiles T, a positive integer threshold parameter τ called the system's *temperature*, and a special tile $\sigma \in T$ denoted as the *seed* tile. Assembly proceeds by attaching copies of tiles from T to a growing seed assembly whenever the placement of a tile on the 2D grid achieves a total strength of attachment from abutting edges, determined by the sum of pairwise glue interactions, that meets or exceeds the temperature parameter τ. The pairwise strength assignment between glues on tile edges is often restricted to be "linear" in that identical glue pairs may be assigned arbitrary positive values, while non-equal pairs are required to have interaction strengths of 0. We denote this restricted version of the model as the *standard* aTAM. When this restriction is not applied, i.e., any pair of glues may be assigned any positive integer strength, we call the model the *flexible glue* aTAM.

Given the aTAM's model of growth, we may consider the problem of designing an aTAM system which is guaranteed to grow into a target shape S, given by a set of 2D integer coordinates, and stop growing. Such systems are guaranteed to exist for any finite shape S, but solutions will typically vary in the number of tiles $|T|$ used. For a given shape S, an interesting problem is to design a system that assembles S while using the fewest, or close to the fewest, number of tiles $|T|$ possible. This fewest possible number of tiles required for the assembly of a given shape S is termed the *program-size* complexity of S.

*E-mail: schwellerr@gmail.com

DOI 10.1007/978-3-642-27848-8_660-1

Problem 1. Let $K_{SA}(n)$ and $K_{SA}^{\sim}(n)$ denote the program-size complexity of an $n \times n$ square for the standard aTAM and the flexible glue aTAM, respectively. What are $K_{SA}(n)$ and $K_{SA}^{\sim}(n)$?

Problem 2. Let $K_{SA}(n,k)$ and $K_{SA}^{\sim}(n,k)$ denote the program-size complexity of a $k \times n$ rectangle for the standard aTAM and the flexible glue aTAM, respectively. What are $K_{SA}(n,k)$ and $K_{SA}^{\sim}(n,k)$?

Problem 3. For an arbitrary given shape S, what is the program-size complexity of S? Let the *scale-free* program size of S be the smallest tile set system that uniquely builds some scaled-up version S. Let $K_{SA}(S)$ and $K_{SA}^{\sim}(S)$ denote the *scale-free* program size of S for the standard aTAM and the flexible glue aTAM, respectively. What are $K_{SA}(S)$ and $K_{SA}^{\sim}(S)$?

Key Results

The best known bounds for program-size complexity for squares, rectangles, and general scaled shapes are presented in this section.

$n \times n$ Squares

The efficient self-assembly of $n \times n$ squares has served as a benchmark for self-assembly algorithms within the aTAM and more general tile assembly models. Within the aTAM, the problem is well understood up to constant factors. The first result states a general upper bound for the program size of self-assembled squares for general n, which is matched by an information-theoretic lower bound that holds for almost all integers n. The precise bounds differ between the standard and flexible glue models but are tight in both cases. The lower bound of inequality (1) is proven in [3] and is based on the Kolmogorov complexity of the integer n. The lower bound of (2) is proven in [2] by the same approach. The upper bound of (1) is proven in [1] and offers an improvement over the initial upper bound of $O(\log n)$ from [3]. The $O(\log n)$ result of [3] is achieved by implementing a key primitive in tile self-assembly: a binary counter of $\log n$ tiles that grows to length n. The improvement of [1] is achieved by modifying the counter concept to work with an optimal, variable base. The upper bound of (2) is proven in [2] and is obtained by combining the aTAM counter primitives with a scheme for efficiently seeding the counter by extracting bits from the values of the flexible glue interactions.

Theorem 1. *There exist positive constants c_1 and c_2 such that for almost all integers $n \in \mathbb{N}$, the following inequalities hold. Moreover, the upper bounds hold for all $n \in \mathbb{N}$.*

$$c_1 \frac{\log n}{\log \log n} \leq K_{SA}(n) \leq c_2 \frac{\log n}{\log \log n}. \tag{1}$$

$$c_1 \sqrt{\log n} \leq K_{SA}^{\sim}(n) \leq c_2 \sqrt{\log n}. \tag{2}$$

While the above theorem presents a tight understanding of the program-size complexity for most self-assembled squares, the information-theoretic lower bound allows for special values of n to be assembled with a much smaller program size. The program size is in fact as small as one

DOI 10.1007/978-3-642-27848-8_660-1

could reasonably hope for. In [3], a tile system is presented that simulates a Busy Beaver Turing Machine and assembles correspondingly large squares for each tile set size. This construction yields the following theorem implying that the largest self-assembled square for a given number of tiles grows faster than any computable function!

Theorem 2. *There exists a positive constant c such that for infinitely many n, $K_{SA}(n) \leq cf(n)$ for $f(n)$ any nondecreasing unbounded computable function.*

Thin Rectangles

The program size of self-assembled squares and other thick rectangles is dictated by information-theoretic bounds which stem from the aTAM's ability to simulate arbitrary Turing machines given enough geometric space to work within. When this space is cut down, such as in the case of building a thin $k \times n$ rectangle, the program size is limited by geometric factors. The following upper and lower bounds are shown in [2] and represent the best known bounds for *thin* $k \times n$ rectangles in which $k = O(\log / \log \log n)$. The lower bound is achieved by a pigeon-hole pumping argument on the types of tiles placed, along with their order of placement, along a width k column of the target rectangle. The upper bound is based on the construction of a general-base, general-width counter, which generalizes the binary counter concept of [3].

Theorem 3. *There exist positive constants c_1 and c_2 such that for any $n, k \in \mathbb{N}$, the following inequalities hold.*

$$c_1 \frac{n^{1/k}}{k} \leq K^{\sim}_{SA}(n,k) \leq K_{SA}(n,k) \leq c_2(n^{1/k} + k).$$

Scaled Shapes

The program size of general shapes is difficult to analyze as it is highly dependent on geometric features of the target shape. However, if we consider the assembly of an arbitrarily scaled-up version of a target shape, these geometric difficulties can be eliminated and a very general result can be achieved. The next result from [4] shows that the scale-free program size of S is closely related to the Kolmogorov complexity of S. In particular, the scale-free program-size complexity of S is a log factor less than the Kolmogorov complexity of S for the standard model, and the scale-free program size complexity of S is the square root of the Kolmogorov complexity of S for the flexible glue model. The standard model result is shown in [4] and is achieved by encoding a compressed description of S in a small tile set which is extracted by a set of tiles simulating a Turing machine that extracts the pixels of S from this compressed representation. The need for the scale factor increase of S is to allow room for the Turing machine simulation. In fact, the required scale factor is the run time of the Turing machine that decompresses the optimal encoding of S. The flexible glue result is achieved by combining portions of the flexible glue construction for squares [2] with the construction of [4]. In the following theorem, $K(S)$ denotes the Kolmogorov complexity of S with respect to some fixed universal Turing machine.

DOI 10.1007/978-3-642-27848-8_660-1

Theorem 4. *For any shape S, there exist positive constants c_1 and c_2 such that*

$$c_1 \frac{K(S)}{\log K(S)} \leq K_{SA}(S) \leq c_2 \frac{K(S)}{\log K(S)}. \tag{3}$$

$$c_1 \sqrt{K(S)} \leq K_{SA}^{\sim}(S) \leq c_2 \sqrt{K(S)}. \tag{4}$$

Open Problems

A few important open problems in this area are as follows. In the case of squares, the program size is well understood as long as the temperature of the system is at least two. A long-standing open problem has been to determine the program-size complexity of $n \times n$ squares for temperature-1 self-assembly in which each positive glue force alone is sufficient to cause a tile attachment. To date, no known method is able to achieve $o(n)$ tile complexity at temperature-1 for an $n \times n$ square, but no proof exists that this cannot be done. With respect to thin $k \times n$ rectangles, the best upper and lower bound have a gap with respect to variable k. Does there exist a more efficient rectangle construction, or can a higher lower bound be derived? Finally, while the scale-free program-size complexity of general shapes is well understood, little is known about the (unscaled) program size of general shapes. What new tools and geometric classifications can be developed to analyze and bound this complexity for general shapes?

Cross-References

- Active Self-Assembly and Molecular Robotics with the Nubot Model
- Combinatorial Optimization and Verification in Self-Assembly
- Intrinsic Universality in Self-Assembly
- Patterned Self-Assembly Tile Set Synthesis
- Randomized Self-Assembly
- Robustness in Self-Assembly
- Self-Assembly at Temperature 1
- Self-Assembly of Fractals
- Self-Assembly of Squares and Scaled Shapes
- Self-Assembly with General Shaped Tiles
- Temperature Programming in Self-Assembly
- Two Handed Self-Assembly

Recommended Reading

1. Adleman L, Cheng Q, Goel A, Huang, M-D (2001) Running time and program size for self-assembled squares. In Proceedings of the thirty-third annual ACM symposium on theory of computing, New York. ACM, pp 740–748
2. Cheng Q, Aggarwal G, Goldwasser MH, Kao M-Y, Schweller RT, de Espanés PM (2005) Complexities for generalized models of self-assembly. SIAM J Comput 34:1493–1515

DOI 10.1007/978-3-642-27848-8_660-1

3. Rothemund PWK, Winfree E (2000) The program-size complexity of self-assembled squares (extended abstract). In Proceedings of the 32nd ACM symposium on theory of computing, STOC'00, Portland, pp 459–468
4. Soloveichik D, Winfree E (2007) Complexity of self-assembled shapes. SIAM J Comput 36(6):1544–1569