Complexity of Self-assembled Shapes
(Extended Abstract*)

David Soloveichik and Erik Winfree
California Institute of Technology, Pasadena, CA 91125, USA
{dsolov, winfree}@caltech.edu

Abstract. The connection between self-assembly and computation suggests that a shape can be considered the output of a self-assembly “program,” a set of tiles that fit together to create a shape. It seems plausible that the size of the smallest self-assembly program that builds a shape and the shape’s descriptional (Kolmogorov) complexity should be related. We show that under the notion of a shape that is independent of scale this is indeed so: in the Tile Assembly Model, the minimal number of distinct tile types necessary to self-assemble an arbitrarily scaled shape can be bounded both above and below in terms of the shape’s Kolmogorov complexity. As part of the proof of the main result, we sketch a general method for converting a program outputting a shape as a list of locations into a set of tile types that self-assembles into a scaled up version of that shape. Our result implies, somewhat counter-intuitively, that self-assembly of a scaled up version of a shape often requires fewer tile types, and suggests that the independence of scale in self-assembly theory plays the same crucial role as the independence of running time in the theory of computability.

1 Introduction

Self-assembly is the process by which an organized structure can spontaneously form from simple parts. The Tile Assembly Model [15][14], based on Wang tiling [13], formalizes the two-dimensional self-assembly of square units called “tiles” using a physically plausible abstraction of crystal growth. In this model, a new tile can adsorb to a growing complex if it binds strongly enough. Each of the four sides of a tile has an associated bond type that interacts with a certain strength with matching sides of other tiles. The process of self-assembly is initiated by a single seed tile and proceeds via the sequential addition of new tiles. Confirming the physical plausibility and relevance of the abstraction, simple self-assembling systems of tiles have been built out of certain types of DNA molecules [16][11][10][8]. The possibility of using self-assembly for nanofabrication of complex components such as circuits has been suggested as a promising application [5].

* A preprint of the full paper can be found at http://arxiv.org.
Complexity of Self-assembled Shapes

The view that the “shape” of a self-assembled complex can be considered the output of a computational process \[2\] has inspired recent interest \[7, 1, 3, 6, 4\]. While it was shown through specific examples that self-assembly could be used to construct interesting shapes and patterns, it was not known in general which shapes could be self-assembled from a small number of tile types. Understanding the complexity of shapes is facilitated by an appropriate definition of shape. In our model, a tile system generates a particular shape if it produces any scaled version of that shape (Sect. 3). This definition may be thought to formalize the idea that a structure can be made up of arbitrarily small pieces. Computationally, it is analogous to disregarding computation time and is thus more appropriate as a notion of output of an universal computation process.\[1\] Using this definition of shape, we show that for any shape \(\tilde{S}\), if \(K_{sa}(\tilde{S})\) is the minimal number of distinct tile types necessary to self-assemble it then \(K_{sa}(\tilde{S}) \log K_{sa}(\tilde{S})\) is within multiplicative and additive constants (independent of \(\tilde{S}\)) of the shape’s Kolmogorov complexity. This theorem is proved by construction (which might be of independent interest) of a general method for converting a program that outputs a fixed size shape as a list of locations into a tile system that self-assembles a scaled version of the shape. Our result ties the computation of a shape and its self-assembly, and, somewhat counter-intuitively, implies that scaling up a shape may often allow it to be self-assembled from fewer tile types. Another consequence of the theorem is that the minimal number of tile types necessary to self-assemble an arbitrary scaling of a shape is uncomputable. Answering the same question about shapes of a fixed size is computable but NP complete \[1\].

2 The Tile Assembly Model

We present a description of the Tile Assembly Model based on Rothemund and Winfree \[7\] and Rothemund \[6\]. We will be working on a \(\mathbb{Z} \times \mathbb{Z}\) grid of unit square locations. The directions \(\mathcal{D} = \{N, E, W, S\}\) are used to indicate relative positions in the grid. Formally, they are functions \(\mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z} \times \mathbb{Z}\):
\[
N(i, j) = (i, j + 1), \quad E(i, j) = (i + 1, j), \quad S(i, j) = (i, j - 1), \quad W(i, j) = (i - 1, j).
\]
The inverse directions are defined naturally: \(N^{-1}(i, j) = S(i, j)\), etc. Let \(\Sigma\) be a set of bond types. A tile type \(\emptyset\) is a 4-tuple \((\sigma_N, \sigma_E, \sigma_S, \sigma_W) \in \Sigma^4\) indicating the associated bond types on the north, east, south, and west sides. Note that tile types are oriented and a rotated version of a tile type is considered to be a different tile type. A special bond type null represents the lack of an interaction and the special tile type empty = (null, null, null, null) represents an empty space. If \(T\) is a set of tile types, a tile is a pair \((\emptyset, (i, j)) \in T \times \mathbb{Z}^2\) indicating that location \((i, j)\) contains the tile type \(\emptyset\). Given the tile \(t = (\emptyset, (i, j))\), \(type(t) = \emptyset\).

\[1\] The production of a shape of a fixed size cannot be considered the output of a universal computation process: whereas questions about the result of universal computation are often uncomputable, any question about a shape of a fixed-size can be answered with a finite simulation of the self-assembly process \[7\], because in the model considered here, once a tile is added, it cannot be removed. Thus questions about shapes of fixed size are decidable.