DNA Computation of Solutions to Edge-Matching Puzzles

Ali A Atiia

A Thesis

In the Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science at

Concordia University

Montreal, Quebec, Canada

January 2011

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Ali A Atiia**

Entitled: **DNA Computation of Solutions to Edge-Matching Puzzles**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. N. Shiri

_____ Examiner
Dr. V. Chvátal

_____ Examiner
Dr. P. Grogono

_____ Co-Supervisor
Dr. N. Kharma

_____ Co-Supervisor
Dr. C. Y. Suen

Approved by _____
Chair of Department or Graduate Program Director

_____
Dr. Robin A. L. Drew, Dean
Faculty of Engineering and Computer Science

Date _____

ABSTRACT

DNA Computation of Solutions to Edge-Matching Puzzles

Ali A Atiia

The resilient, ancient, and fine-tuned DNA (deoxyribonucleic acid) has inspired many researchers to harness its power for material purposes. In this work, we use synthesized DNA strands to compute the solution to an instance of edge-matching puzzles (EMP), where the challenge is to pack a collection of edge-coloured square tiles on a square board such that all adjacent edges match in colour. We encode tiles with DNA strands and make use of structural, chemical and enzymatic properties of DNA to effectively carry out a brute-force search of the solution to the puzzle. The solution ultimately results as a 2-dimensional DNA lattice encoding the position and orientation of each tile on the solution board. Our approach has been to represent a tile as the union of two half-tiles. This conceptual representation allows for the use of a supremely powerful heuristic: polymerase chain reaction (PCR), which can be inserted at any step of the protocol to selectively amplify certain strands to exponential quantities. Our abstract formalization of half-tiles and the DNA protocol we use to manipulate them have relevance in three ways. First, by solving an instance of the (NP-Complete) EMP problem we make precise characterizations of the processing power of DNA Computing. Second, the 2-dimensional self-assembly of half-tiles is Turing-complete. Thirdly, the 2-dimensional self-assembly of half-tiles can serve as a PCR-powered model for massive nano-scale fabrication of 2-dimensional DNA nano-shapes.

Acknowledgements

I would like to thank Nawwaf Kharma for his unlimited support and guidance throughout the life time of this work. Many thanks go to Luc Varin for letting me work in his lab and for the valuable insights he provided. Thanks to Ching Y. Suen for extra funding. Special gratitude to Samar Elzein, Ming Sun and Juan Francisco Leyva Illa, who helped me learn molecular biology and take my first baby steps in the lab; I owe a great deal to their mentoring. Partial funding for this work was graciously provided by Centre hospitalier de l'Université de Montréal (CHUM), special thanks to director Guy Rouleau, and to Patrick Dion and Daniel Rochefort for the many valuable discussions. Thanks to Reginald Storms and Christopher Wilds for meeting with me and hearing my ideas; thanks to Vanessa Blandford and Christina Sawchyn for help and discussions; thanks to Hala Khalil for her generous lending of lab equipment in desperate times.

Table of Contents

List of Figures

List of Tables

List of Acronyms

BEMP: bounded edge-matching puzzle
bp: base-pair
dsDNA: double-stranded DNA
EMP: edge-matching puzzle
GE: gel electrophoresis
hTAM: half-Tile Assembly Model
PCR: polymerase chain reaction
ssDNA: single-stranded DNA
TAM: Tile Assembly Model

Chapter 1 Introduction

The scientific understanding of the structural, functional, chemical and enzymatic properties of the deoxyribonucleic acid (DNA) has grown rapidly over the past sixty years. Since discovered in 1953 [57], the DNA double-helix has become one of the most recognizable icons and indeed one of the hallmarks of scientific progress in human history. It is, after all, the robust and efficient information-carrier encoding the blueprint of living organisms –including ourselves. And it has been so for billions of years.

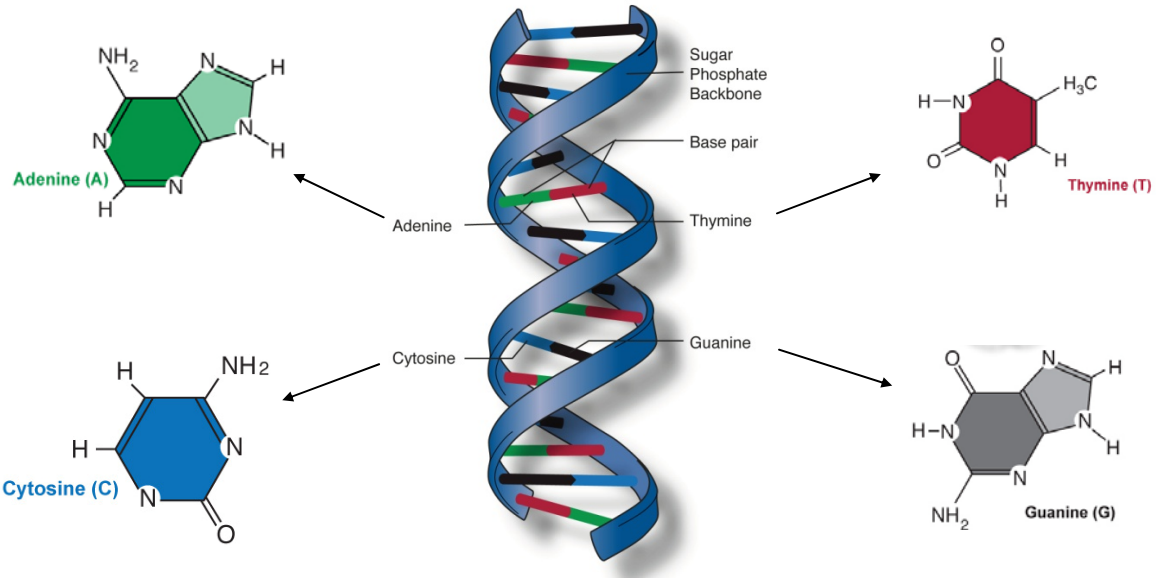


Figure 1.1: The DNA double-helix. A strand of DNA is made up of series of four compounds (Adenine, Thymine, Cytosine, Guanine, or A, T, C, G for short) connected together with a sugar-phosphate backbone[1] (a sequence of DNA is therefore a string over an alphabet $\Sigma = \{A, T, C, G\}$). Two strands hybridize: always A, T, C, G in one strands mating with T, A, G, C in the other, respectively, to form the notorious double-helix.[2]

If computation is a process by which new information/states result from previous ones in accordance with some "rules", then the central dogma of molecular biology (DNA-to-mRNA-

[1] Recent discoveries [64] found a strain of bacterium which substitutes arsenic for phosphorus in their nucleic acids.

[2] Graphics courtesy of Darryl Leja, graciously provided to the public domain by the National Human Genome Research Institute, Bethesda, MD.

to-protein [18], or simply “life as we know it”) is no less than a computational classic. Therefore, as we synthesize DNA sequences and harness them for computation, we should not forget that DNA has been computing long before we even existed. In fact, there is no structural, chemical or enzymatic operation used in DNA Computing which does not already occur in nature. Perhaps it is no surprise that such a marvellous molecule has enticed researchers to use it for various material purposes. It should also not come as a surprise that DNA has proven to be a heavy-weight contender for emerging molecular paradigms of computing and nanotechnological material fabrication. In DNA Computing, DNA strands’ base-composition and the chemical and enzymatic reactions that govern their interactions are given new *semantic* meanings that represent real-world pieces of information –whatever we may desire that to be.

Three obvious questions arise: *what* exactly is DNA Computing? *How* would we use it to solve an edge-matching puzzle (EMP)? And *why* would we want to do that anyway? In what remains of this chapter, we present pioneering works in the field (1.2), while expanding further on those relating directly to our discussion. Next we present a brief overview (the *how*) of our DNA-based approach to solving EMPs: from representation (1.3.1) to various manipulations that we harness to find the solution (1.3.2-5). We conclude this chapter in 1.4 with an attempt an answer the *why* question by elaborating on the motivation.

1.1 What is DNA Computing?

DNA Computing is a new *perspective*, not a technological invention. The field was born as a result of an insightful [4] new way of attributing to DNA strands physical or abstract real-world *objects/relations*. DNA strands have come to represent, say, nodes/edges in a graph, variables/truth values in a SAT formula, or tiles/tiling in a recreational edge-matching puzzle (Chapter 4). The three elementary operations in DNA Computing are: 1) the hybridization of two

strands through Watson-Crick complementarity (T-A, C-G base pairing) to form a double-helix, 2) the concatenation of two strands by a ligase enzyme, effectively rendering the two into one continuous strand, and 3) the selective and exponential replication of strands by a polymerase enzyme. All of these operations occur in nature and so DNA Computing has nothing to show for in this respect. However, our *interpretation* of the outcome of such operations on a pool of DNA strands –each of which now stands for "something" of interest to us– can indeed lead to useful and meaningful information/states.

When two strands X and Y are ligated into one continuous strand Z, we can interpret the existence of Z as "there exists a relation between X and Y". The relation (or *rule*) is itself a strand that is Watson-Crick complementary to X in part and to Y in another. Figure 1.2a depicts a schematic illustration of an example in which two real-world concepts, "Toronto" and "Montreal", were represented with random 16-base-pair (bp) single-stranded DNA (ssDNA) oligonucleotides (or strands). A third 16-bp sequence, which is Watson-Crick complementary to two 8-bp subsequences of "Toronto" and "Montreal" at their 5'- and 3'-ends, respectively[3], is (from our high-level viewpoint) the relation "connected-by-highway".

[3] The 5' (pronounced "5 prime") denotes the strand's end having the fifth carbon of the ribose at its terminus. The 3' denotes the end terminating with the third carbon atom connected to the hydroxyl compound. DNA strands hybridize while in opposite 5'/3' conformations.

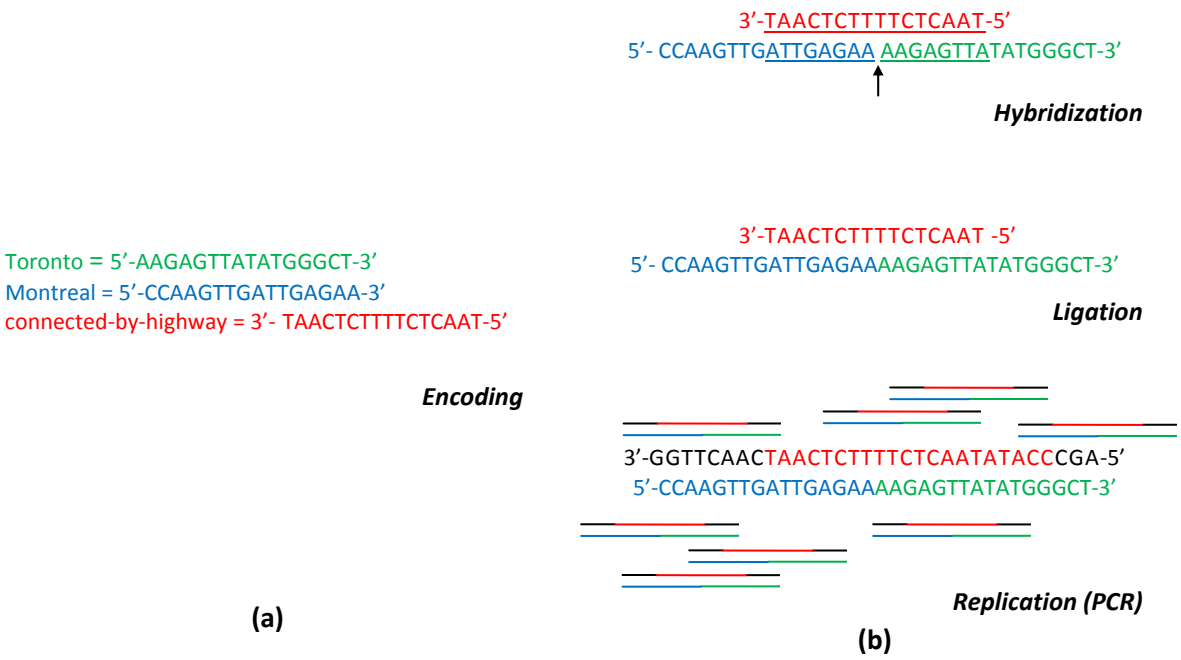


Figure 1.2: Elementary operations in DNA Computing. a) Encoding of two objects (cities) and a relation ("connected-by-highway") with 16-bp sequence each, the latter being partially WC-complementary to the former two. **b)** The three fundamental operations of DNA Computing. The WC-complementarity brings two strands together, with a "nick" at their meeting point (indicated by an arrow) which is sealed by ligation. Notice the reverse 5'/3' directionality of hybridizing strands. The ligation product can be exponentially amplified by PCR. The replication results in fully double-stranded DNA (dsDNA), see 1.3.3 for PCR details.

Figure 1.2b illustrates the three fundamental operations of DNA Computing. The two strands, "Montreal" (blue) and "Toronto" (green) are brought together by the "connected-by-highway" (red) strand by virtue of hybridization according to Watson-Crick complementarity. The post-hybridization gap (arrow in Figure 1.1b) at the meeting point of the two strands is called a DNA "nick" which is subsequently sealed by the ligation process (by adding a missing phosphodiester bond), effectively concatenating the two into one continuous strand.

This assembly can be exponentially replicated by polymerase chain reaction (PCR), which is one of the most powerful heuristic tools in DNA Computing. Notice in Figure 1.2b that

replication produces perfect double-stranded DNA (notice the black regions in the complementary strand, see 1.3.3 for details). There are other lab techniques that are involved in DNA Computing (1.3.4 for example), but our criteria of an "operation" is restricted to those involving alterations of DNA sequences.

1.2 Related Work

In a seminal paper in 1994, Adleman [4] demonstrated a DNA-based solution to an instance of the Hamiltonian path problem: given a set of cities and routes between them, find a path –if it exists– that starts in city A and ends in city B while in the process visiting all other cities exactly once [24]. Using mainly DNA encoding and operations depicted in Figure 1.2 above[4], the solution to the problem resulted at the end as a long DNA sequence.

This generated considerable interest and inspired a wide range of new research proposing [31][9][13][40][6][8] and implementing [45][46][14] DNA-based solutions to other problems. Moreover, new avenues for DNA in the material world subsequently sprung up, ranging from Turing-universal DNA models [63][56][35], to DNA finite state machines for control of gene expressions (or what can be referred to as *in vivo* molecular automata) [11][55][38], to DNA nanotechnological constructions [34][47][48][65][61].

Three aspects of the emerging field of DNA Computing are relevant to our work on EMPs: DNA-based solution to NP-Complete problems, Turing-universal computations through 2-dimensional DNA self-assembly, and DNA models for nanotechnological fabrication. In what follows we expand further on research progress in these three aspects of DNA Computing.

[4] Other techniques that Adleman used included gel electrophoresis whereby strands are separated by sequence length (see 1.3.4), and successive filtrations using magnetic beads; see [4] for details.

1.2.1 DNA Computing for Combinatorial Optimization

No known algorithm can solve all instances of NP-Complete problems efficiently [24], and whether there will ever be such an algorithm remains one of the foremost quest of computer science and mathematics today [21]. Problems in this class are polynomial-time reducible to each other [17][24], and so to solve one efficiently is to so solve them all. However, no matter how hard computer scientists attempt to find fast algorithms for such problems, those algorithms would □at least in worst case instances□ always require exponential amount of computational resources as the problem size grows larger. For better or worse, the fact that Adleman demonstrated a DNA solution to an instance of an NP-Complete problem –the Hamiltonian path problem– accounted for the initial excitement [9] and the surging research into the field. DNA-Based solutions to small instances of the notorious SAT problem were demonstrated in [45][46][14].

The excitement brought new and interesting ideas around molecular computation in general [9][11][63], but it also gave a misleading impression that a magic remedy for the frustratingly elusive NP-Complete problems might be on the horizon. Some have suggested that we may even be able to break the DES code![13]. In 1.4 we present a brief critique of such misleading (or misled) impression, which still continues to manifest (see [10] for example).

1.2.2 Turing-universal Computation Using 2-Dimensional DNA Self-Assembly:

One of the earliest questions that arose in DNA Computing was whether DNA-based Turing-universal computational models are feasible. Winfree [63] articulated Turing-universal computation using DNA Wang tiles (assembled by virtue of flexible DNA structures known as double-crossover assemblies, see 4.1.3), which he and collaborators later demonstrated [41].

Wang tiles are exactly the same tiles in edge-matching puzzles (EMP): edge-attributed square plates. In 1963, Wang proved that a set of tiles can in fact simulate the execution of universal Turing machines [59] and his work was the theoretical foundation for Winfree's. The fundamental challenge for such a model is that the task of finding tile sets is itself computationally hard [62].

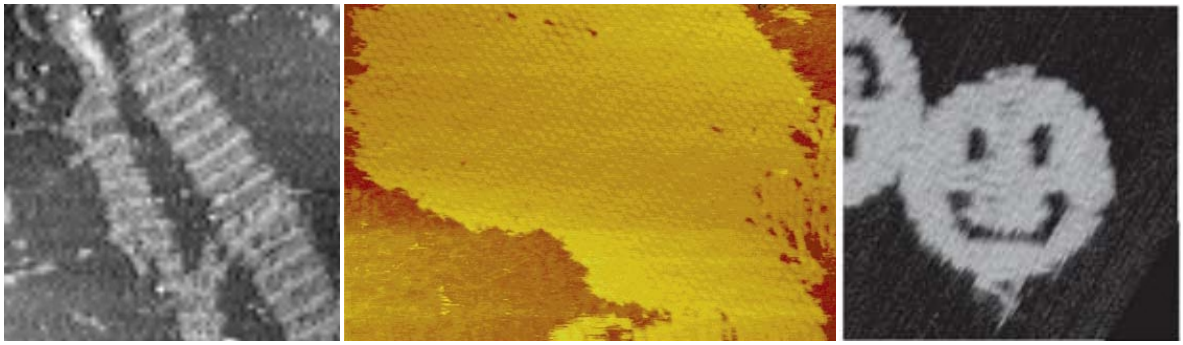
Another interesting and more recent demonstration was done by Su et. al. [56] who proposed a DNA implementation of the NOR logical operation which can be the universal primitive for a molecular computer (all other Boolean logic gates can be built from it [36]). The model, termed DNA "surface computer", involves the immobilization of DNA strands on a surface and proof-of-concept logical operations were demonstrated.

1.2.3 Nanotechnological Constructions Using 2-Dimensional DNA Self-Assembly:

Investigations into the potential of DNA structures that could be used for nanotechnological purposes have been pioneered by N. Seeman, who showed as early as 1982 the flexibility of DNA structures that could serve as basic units in the fabrication of nanostructures [49]. The DNA double-crossover (DX) assemblies, which manifest in various conformations [47] and were initially inspired by naturally occurring Holliday junctions [26][32], have been the basis for many demonstrated 2-dimensional DNA nanotechnological constructions [61][34][42][37] (see Figure 1.3).

Wang tiles and the well-established mathematical theory of tiling (see Chapter 2) were also used as a theoretical model for creating sets of DNA DX units (which, through their four "sticky ends" translate the four edges of a tile), and the growth of these molecular tiles would of course follow the rules of edge-matching of Wang tiles and thus a specified shape would result [61]. Clearly there is an overlap between these works and those mentioned in the previous

subsection. In fact, many pioneering researchers are involved in the two (see Seeman's elegant discussion in [52]). Detailed overview of DNA nanotechnology in general can also be found in [48]; see also [53] on DNA nanodevices.



(a) **(b)** **(c)**

Figure 1.3 Examples of pioneering work in DNA 2-dimensional nanotechnological constructions by **(a)** Winfree et al. [61] (1998) **(b)** Reishus et al. [37] (2005) and **(c)** Rothmund [42] (2006). The three snapshots are atomic force microscopy (AFM) images at 300, 600 and 165 nanometre resolution, respectively.

1.3 How DNA Can Solve a Bounded Edge-Matching Puzzle

An edge-matching puzzle is a recreational tiling challenge in which a collection of edge-coloured square tiles must be aligned on a square grid such that abutting edges have matching colours. Figure 1.4a shows an example puzzle set of 9 tiles. In this particular EMP variant, there is no constraint on the colour of boundary edges. Although solving an EMP of this size is trivial (Figure 1.4b), EMPs are in fact computationally hard problems (NP-Complete) [24] as the resources needed to solve them grow exponentially large as the number of tiles grows larger[5] (see 2.1.3.2 on the computational complexity of EMPs).

[5] Of course the exact resources for a *specific* instance of the puzzle would also depend on the colour frequency distribution; and so a puzzle where there are only 2 copies of each colour can clearly be solved in $O(1)$. See also [7] for an empirical study of hardness vs. colour distribution.

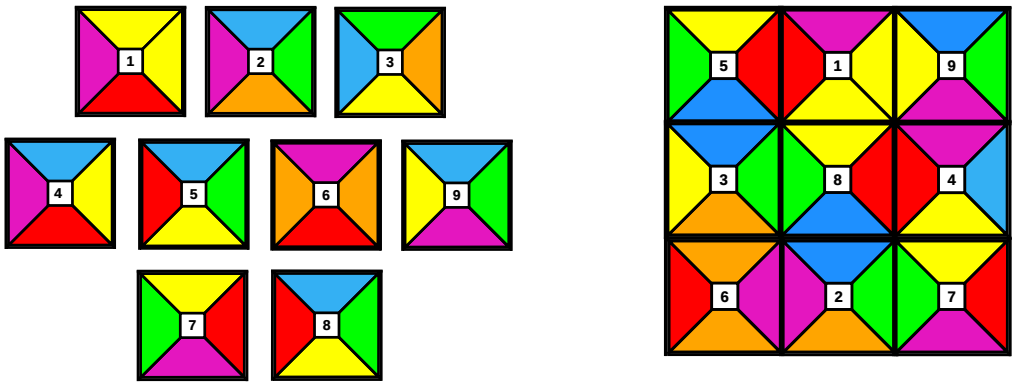


Figure 1.4 An instance of a 3x3 edge-matching puzzle. a) A 9-tile puzzle set. The challenge is to arrange them on the 3x3 grid such that all abutting edges match in colour –no constraints on the boundary edges. **b)** The solution to the set. In the process of solving the puzzle, tiles may be rotated but not reflected (flipped).

Furthermore, these tiles (and tiling in general) are also of important and intriguing mathematical relevance (see 2.1.3.1), and as such, their DNA representation has been at the centre of investigations into the 2-dimensional DNA self-assembly for both molecular Turing-complete models and directed nanotechnological constructions (see 1.2.2 and 1.2.3 above). In what follows we discuss –in general terms– our approach to a DNA-based solution to such puzzles: how a tile is encoded with DNA strands; how various operations (hybridization/ligation/PCR) and laboratory techniques on these strands amount to computations and heuristics that eventually lead to an answer to the puzzle.

1.3.1 DNA Strands as Half-Tiles

Our approach will be to conceptually re-think of a tile as a union of two half-tiles. That conceptual view of tiles is then translated into DNA strand representation. That is our semantic mapping between the puzzles' objects and DNA strands (see 3.2.1). If a tile is represented as the union of two half-tiles along the diagonal, and given that there are two diagonals in a square tile, then clearly there are two possible pairs of half-tiles (Figure 1.5a). As will be laid out formally and in details in Chapter 3, and after defining a tiling grid in Cartesian space in Chapter

2, these pairs can reproduce the tile on the two dimensional tiling grid *in one of the four orientations* (3.2.1). In addition to the two strands representing the pair of half-tiles, two DNA "bridging" strands are used to bring the half-tiles together and so the full tile translates into as a DNA double-crossover (DX) molecule (Figure 1.5b). It is called a double-crossover because the bridging strands hybridize to one half-tile's strand then *cross over* and hybridize with the other half-tile in the opposite directionality (see red strands in Figure 1.5b).

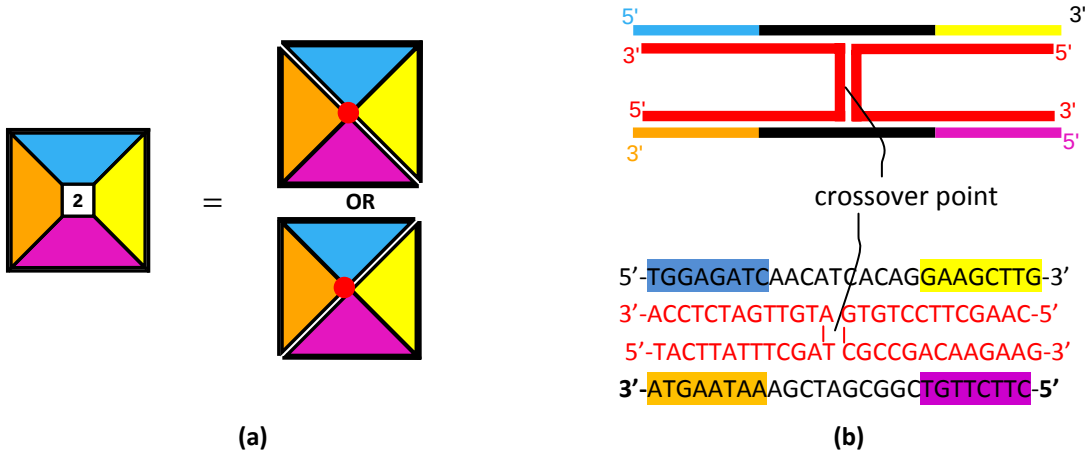


Figure 1.5 DNA representations of tiles. a) An example tile and the corresponding pairs of half-tiles that result from dissecting the tile along the diagonals. **b)** Strands involved in encoding one pair (up in (a)) of half-tiles. Each half-tile is encoded as a strand, and the two half-tiles are brought together by a pair of DNA "bridging" strands (red) to reproduce the mother tile; schematic depiction (up) and example strands (down). The bridging strands are unique per pair per tile (see 3.2.1 on colour-sign compound attributes). A bridging strand hybridizes to one half-tile then crosses over and hybridizes to the other half-tile in opposite directionality (note the 5'/3' directionality of all strands; hybridizing strands are always in opposite directionality); hence the name "double-crossover" (DX) molecule. The schematic representation must be viewed with a caveat: in reality there is no "space" between the last base up and the first base down at the crossover point.

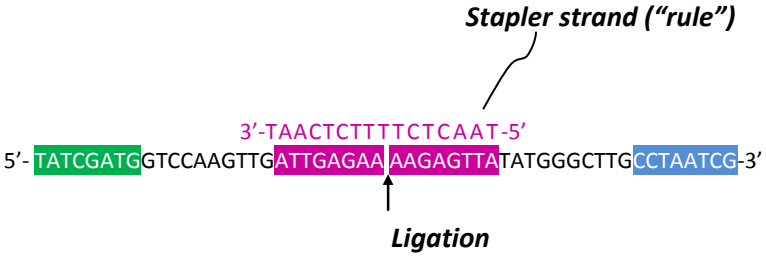
1.3.2 DNA Hybridizations/Ligations as Unit Computations

In 1.1 we showed how a single hybridization/ligation step can result in a meaningful piece of information –we called it a unit DNA computation. And the meaning of this unit computation in the context of EMP is straight forward: it's the act of aligning two tiles together as if by a child attempting to solve the puzzle (Figure 1.6a). For the time being, let us ignore the

DNA bridging strands mentioned above (red in Figure 1.5b; revisited in 1.3.5) and assume having only the DNA strands representing half-tiles. The act of aligning two tiles is the same as aligning two half-tiles along that common edge (Figure 1.6b).

Of course, enforcing the no-flipping constraint and other requirements (simply: a well-defined solution) will be laid out formally and in details in Chapter 2 and 3 (see 3.2.2 in particular). But the important aspect to emphasize at this point is the notion of a pool of half-tiles that are "let loose" (absent the bridging strands) and allowed to align (or "staple") with other half-tiles to create "**lanes**" of half-tiles, while of course observing edge-matching constraints (Figure 1.6c; see also Figure 4.5 for an expanded view).

(a) **(b)**



(c)

Figure 1.6 A unit DNA computation in the context of EMPs. (a) The alignment of two tiles as a unit action in the process of solving an EMP. **b)** The same action conceptually perceived as the alignment of the two relevant half-tiles. **c)** The alignment ("stapling") of two half-tiles by a stapler strand (or the edge-matching "rule") through strand complementarity. Ligation follows, sealing the 5'/3' meeting point of the two strands.

This has profound implications on the DNA solution to the puzzle (and to the application of Wang tiles in DNA models for Turing-universal computation or nanotechnological

constructions). It means that, *before* we bridge pairs of half-tiles –that have now together formed lanes– to their pair complements to re-produce mother tiles, we can employ two powerful computational heuristics on these lanes: PCR and Gel Electrophoresis.

1.3.3 Polymerase Chain Reaction as a DNA Computational Heuristic and Processing Power

Bounded edge-matching puzzles (BEMP), which will be the focus of this thesis, have a boundary constraint: edges of colour grey must appear on the boundary (see figures in 2.1.2). As we mix all half-tile and stapler strands and generate exhaustive stapling, we can make use of the fact that a valid lane must begin and end with a sequence encoding the "grey" colour. We use "grey primers" to exponentially amplify the molarity of these lanes. Figure 1.7 is a schematic depiction of how the PCR carries on.

The idea of *molarity* as the computational resource in DNA Computing was briefly touched upon in 1.2.1 and will be investigated in details in 4.2 and 4.3. PCR increases the molarity of certain strands (ruled by what primers we add to the reaction), and can therefore be viewed as a DNA computational heuristic (it favours certain strands against others in the mix) and also as a processing power (larger search space can exhaustively be searched the larger the molarity). It is worth noting that PCR is relatively cheap and available (it is a daily procedure in almost every molecular biology laboratory).

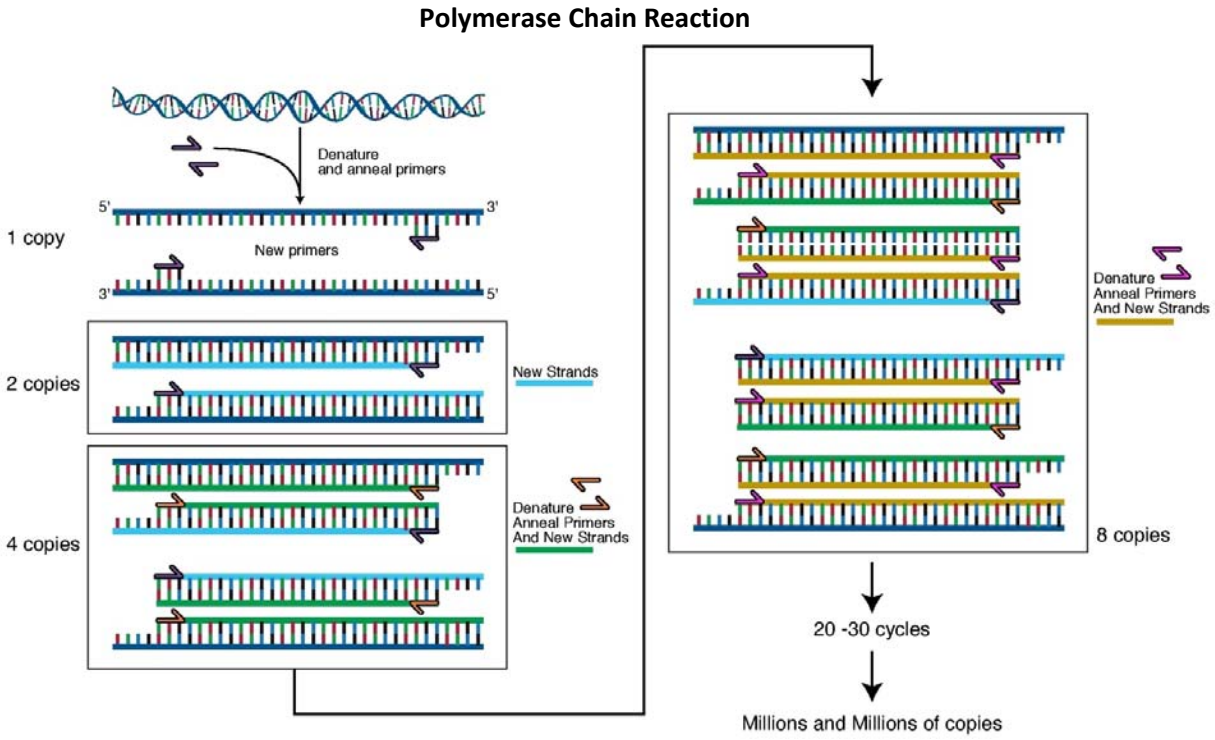


Figure 1.7 Polymerase Chain Reaction: A dsDNA strand can selectively be amplified to exponentially many copies. Forward and reverse primers (short sequences of ssDNA WC-complementary to the beginning and end of a strand and its complement, respectively) hybridize to denatured strands; the polymerase enzyme then makes the copying (making A, T, C, G base for every T, A, G, C in the sequence)[6].

1.3.4 Gel Electrophoresis as a DNA Computational Heuristic:

In our DNA-based protocol to solve BEMPs, we will take advantage of the fact that all valid lanes are of odd lengths 1, 3, 5 … $n-1$ half-tiles, where n is the size of the puzzle (see Figure 3.3 for a hint on what lanes would look like on the tiling grid). We know the length of each half-tile strand and so we know the length of a lane comprised of x half-tiles. Gel electrophoresis is a simple technique by which we can separate diverse products of double-stranded DNA in one sample according to base-pair (bp) length. A DNA sample is loaded into an

[6] Graphics courtesy of Darryl Leja (with some modifications) graciously provided to the public domain by the National Human Genome Research Institute, Bethesda, MD.

agarose or polyacrylamide gel and electrical field is applied, making DNA (which is negatively charged) migrate towards the positive anode. Because the speed at which a DNA strand can "zigzag" its way through the gel material is inversely proportional to the logarithm of its length [58], DNA strands with different bp lengths migrate at different speeds and are thus separated on the gel (see Figure 1.8). The gel is stained with DNA-binding fluorescent material (e.g. ethidium bromide) which shines up under ultraviolet light, enabling us to see the various bands and deduct their lengths by comparison to a DNA ladder with known band resolution (commercially available, left in Figure 1.8). Bands of interests can be excised (literally) from the gel and purified. Notice that we have not mentioned gel electrophoresis as a fundamental operation in DNA Computing because it's really more of a laboratory technique (and there exist many [44]) and doesn't involve any alterations of DNA per se.

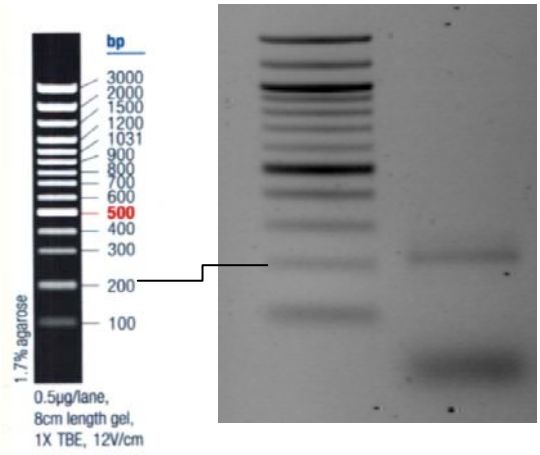


Figure 1.8 Gel electrophoresis: The lengths of DNA products loaded in well 2 can be identified by cross comparison with bands in well 1, which contains products from commercially available DNA "ladder". In this example (taken from a trial experiment in our lab) the upper band in W2 (at ~210 bp length) was the desired product. It can be excised and purified (see Chapter 4 for further details), while the lower band is discarded. Notice that the products of these two bands were originally in one sample (say, a PCR product).

Since the original mixing of half-tile and stapler strands results in random and exhaustive stapling (exhaustive that is if we have the required molarity of each strand in the mix), we are bound to eventually end up with erroneous lanes (say, lanes of even length). Those can be discarded through electrophoresis, while the desired ones can be retained (and further amplified for example) and carried on to the next step in the protocol: bridging.

1.3.5 Bridging of dsDNA

It is worth emphasizing again the notion that pairs of half-tiles (i.e. those belonging to the same dissection of a tile, see 3.2.1) have by this stage gone through hybridization/ligation, gel electrophoresis and PCR *independently* from complement half-tiles in the same pair, resulting in "lanes" of half-tiles in the form of dsDNA. Now, in the bridging phase, the *union* operation is applied to bring half-tiles in one lane to their pair complements in another – effectively reproducing mother tiles. But what does all this really mean in terms of solving a BEMP?

Consider the schematic representation in Figure 1.9 which depicts the (ultimately sought) 2-dimensional DNA assembly encoding the solution to a 4×4 BEMP. The lanes[7] are mixed with bridging strands (red) and they "stack up". The details of this process will be discussed extensively, both in abstract and practical terms (Chapter 3 and 4). But the essential point is this: if all pairs of half-tiles in a BEMP are encoded such that every tile can ultimately assume every ***position*** and every ***orientation*** on the DNA grid, then we have effectively carried out a brute force search for the solution of the puzzle.

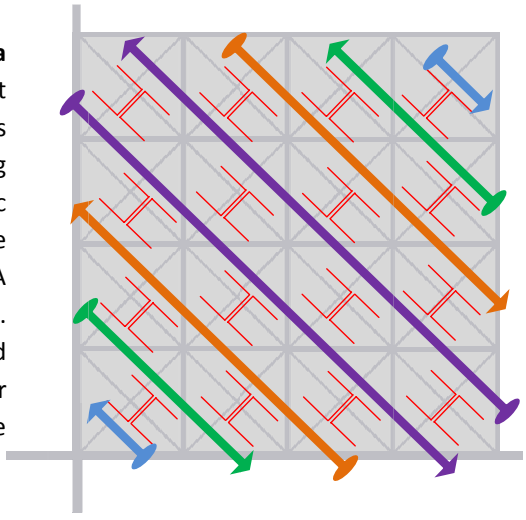


Figure 1.9 Schematic view of 2D DNA assembly encoding a solution to a 4×4 BEMP: The DNA assembly (foreground) as it relates to the 4×4 tiling grid (background). A half-tile on one lane is bridged to its pair half-tile in another lane, effectively re-producing the mother tile in a certain orientation. The schematic representation should be viewed with caveats: in reality there are no "gaps" between lanes, and the totality of DNA exists in dsDNA form. Bridged lanes are the result of hyb./lig., PCR and G.E. described above. The combination of stapling (producing lanes) and bridging (stacking up lanes) is effectively an exhaustive search over all positions and orientations. Eclipse and arrows in lanes indicate 5'/3' terminus.

[7] More precisely, one of the two strands of the dsDNAs representing lanes. The other strand is irrelevant.

1.4 Why Solve EMPs with DNA: Motivation

A DNA strand is made up of dozens of atoms, atoms are "small" and so the implicit assumption is that we might be able to solve larger instances of NP-Complete problems with the help of the massive parallelism of hybridizing DNA strands –we may even be able to break the DES code [13]. Unfortunately, an atom is still too heavy when we are dealing with computational resources that increase exponentially as problem sizes increase linearly. In a small note that didn't get the attention it deserved at the time, Hartmanis [25] elegantly showed in 1995 how an HPP of a 200-city tour solved using Adleman's method would require an amount of DNA that is more the weight of the Earth. Our analysis of upper-bound requirements in Chapter 4 also demonstrates the exponential molarity requirements as the (NP-Complete) edge-matching puzzle grows larger.

Thus, just as the required CPU and/or memory requirements in silicon-based computing grow exponentially as sizes of NP-Complete instances grow linearly, so too does the molarity of DNA strands in DNA Computing. Similar observations were also brought up in [30][33][15]. Said differently, there are no technological tricks around the inherent intractability of NP-Complete problems; it is efficient algorithms we are missing, and whether those will ever be found is indeed the most important question in computer science today (crystallized in the notorious P vs. NP question [21]).[8]

Nonetheless, solving an NP-Complete instance using DNA may actually have forced us to revisit fundamental notions of computability [5] and may have also helped us better realize

[8] Similar misconceptions seem to revolve around quantum computing too. Although one polynomial-time quantum algorithm exists for integer factorization[54], there is yet to be a proof that quantum computing can indeed solve NP-Complete in polynomial time [1][2][3].

the inherent intractability of NP-Complete problems in a new computational medium. Furthermore, the computational potential of DNA need not be judged by whether it can *surpass* silicon-based computing, but rather by how it can *supplement* it: a DNA molecular computer is indeed a more fitting (and a much needed in fact) choice for intelligent *in vivo* drug administration [11] than a silicon-based one, for example. And it is exactly for this reason that precise characterization of the power of DNA Computing is needed, and that is one of the goals of this thesis. Solving combinatorial optimization problems using DNA can actually be a means to making precise assessments about the computation power of hybridization/ligation and PCR, and to say a statement with clarity about "how much" can practically be computed.

The half-tile model in our discussion can also be viewed as PCR-enabled Turing-complete computational model through self-assembly. With minor modification to the procedure of encoding half-tiles of EMPs (namely, disabling rotations of tiles on the assembly[9], see Chapter 5), we can show the equivalence of half-Tile and the Tile Assembly Model [63]. The advantage of the former is that PCR can be inserted at any step of the protocol. PCR insertion is discussed in 4.3.1.

Moreover, just as Wang tiles were used as definitional units of 2-dimensional DNA nanotechnology, an assembly model based on the concept of half-tile is possible. In fact, the PCR-insertion admissibility of the half-tile assembly model makes it a supremely powerful model for nanotechnological constructions: not only can we make certain nano "stuff", we can make it in abundance. Lanes of half-tiles can be amplified to the desired molarity before stacking them into 2-dimensional lattices (see 4.3 on PCR cycles vs. molarity). In fact, PCR can even be inserted half-way through the bridging process (if bridging is carried out step-wise as our protocol

[9] Rotation is not allowed for Wang tiles, see [59].

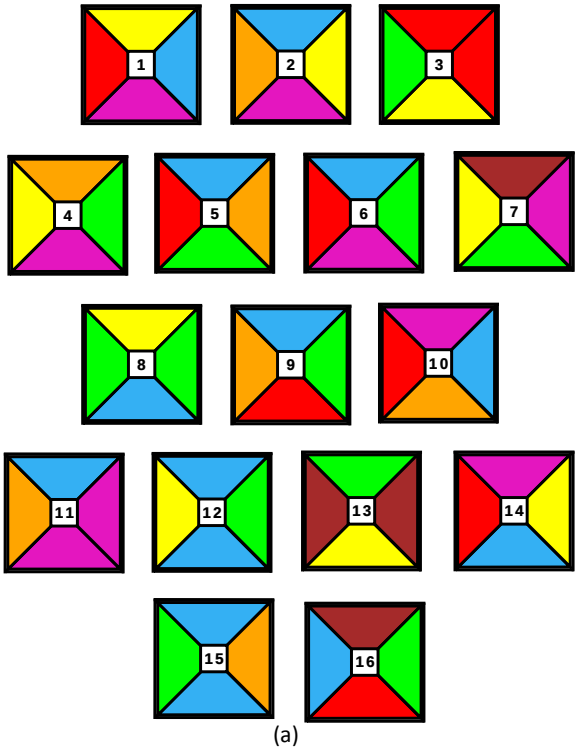
outlines in Chapter 4). The task of a nanotechnologist is to simply ask a computer program the following question: given a final desired nano-scale shape, what is the set of half-tiles the growth of which results in that shape. A harder question can be asked of a computer: what is the smallest set of half-tiles the growth of which produces that nano-shape and *only* that shape. What remains after is the DNA encoding of half-tiles and the execution of series of laboratory protocols, all of which are demonstrated in the presented DNA protocol for solving BEMPs.

Chapter 2 Problem Definition

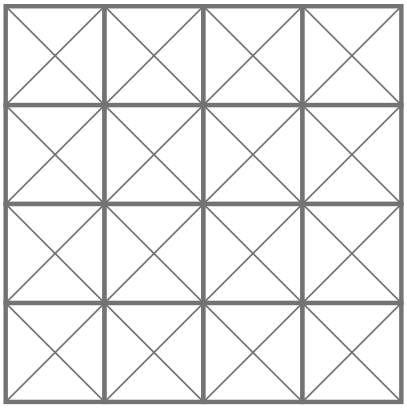
2.1 Introduction to Edge-Matching Puzzles

2.1.1 What is an Edge-Matching Puzzle?

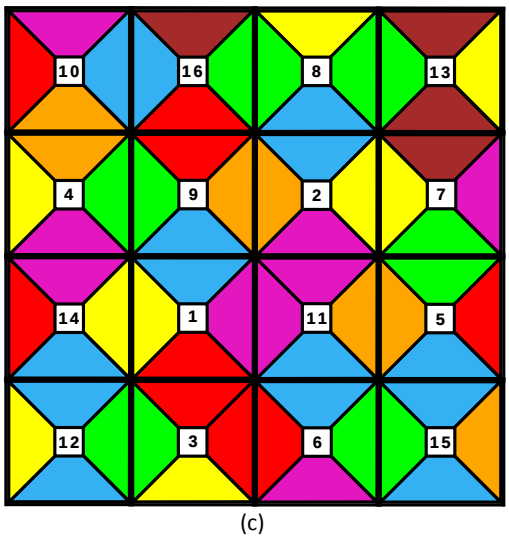
An edge-matching puzzle (EMP) is a recreational tiling puzzle which, in its most common form[1], requires packing a collection of identically-sized square tiles on a square grid such that contiguous edges of neighbouring tiles have matching attributes. The attributes assigned to edges can be colours and, in some cases, a combination of colour and sign. We focus our



(a)



(b)



(c)

Figure 2.1: An example edge-matching puzzle. **a)** A set of 16 tiles for a 4×4 edge-matching puzzle. A Tile's IDs serves as a reference only. **b)** A 4×4 grid on which the 16 tiles are to be packed observe ring the colour-matching constraint of the puzzle. **c)** The 16 tiles of (a) are successfully arranged on the grid with all abutting edges matching in colour. Notice that every edge matches colour with its opposite, unless the edge is on the boundary of the puzzle, in which case no further constraint is imposed.

[1] See [27] for examples of other variants.

attention here on one variant of EMPs in which the pieces are n square tiles where there is exactly one copy of each tile and all must appear in the solution. Figure 2.1a shows an example of an EMP set of 16 tiles with colour-attributed edges. The tiles are assigned IDs as serial numbers from 1 to n and serve as a reference only.

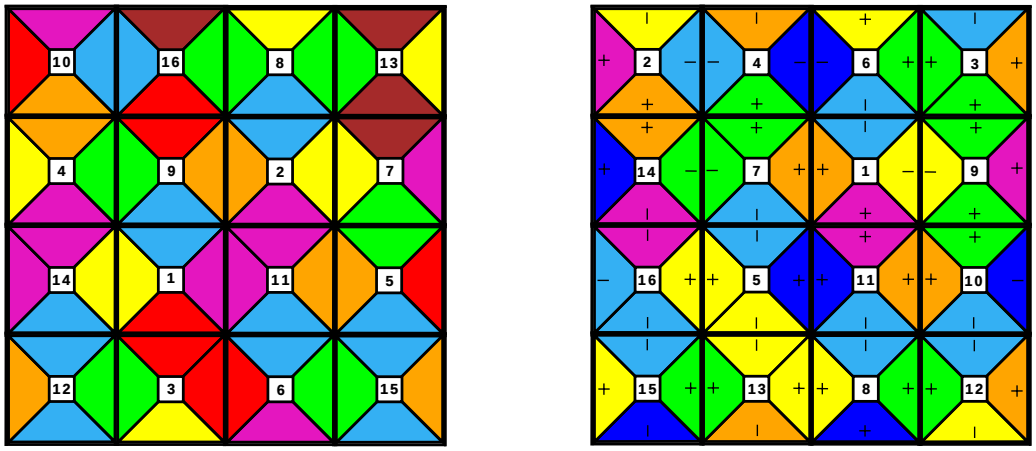
The puzzle board is a finite square grid whose unit area is one square tile (Figure 2.1b), and as such has an area of $\sqrt{n} \times \sqrt{n}$ units, where n is the total number of tiles in the puzzle and$\sqrt{n} \in \mathbb{N}$. The tiles can be rotated in the course of attempting to solve the puzzle but can not be reflected (i.e. a tile can appear in whatever orientation, see Figure 2.1c). Unlike jigsaw puzzles, the final arrangement of tiles in a solved EMPs of this type do not necessarily reveal a grand image on the grid, and so in the course of solving the puzzle, there is no hint as to whether a successfully placed tile in a partially-tiled grid is in the correct position ultimately.

EMPs can be traced as far back as 1890s [20] but have recently generated popular interest after the release of Eternity II puzzle [23], a commercial 16×16 (256 tiles in total) EMP with a hefty prize of $2 million for the first person to have successfully solved it.

2.1.2 Variations of Boundary and Edge-Matching Constraints

Two criteria can be used to distinguish different types of EMPs: 1) whether there exists a unique colour such that edges attributed with that colour must appear on the boundary of the solution grid [23], and 2) whether edges have, in addition to colour, a sign such that abutting edges must match in both colour *and* sign. In *unsigned-unbounded* EMPS, there is no designated colour for boundary edges, and the edge-matching constraint is simply colour-matching of abutting edges (Figure 2.2a). In *signed-unbounded* EMPs, abutting edges must match in colour

and sign[2], but just as in the previous type, there is no restriction on the colour for boundary edges (Figure 2.2b).

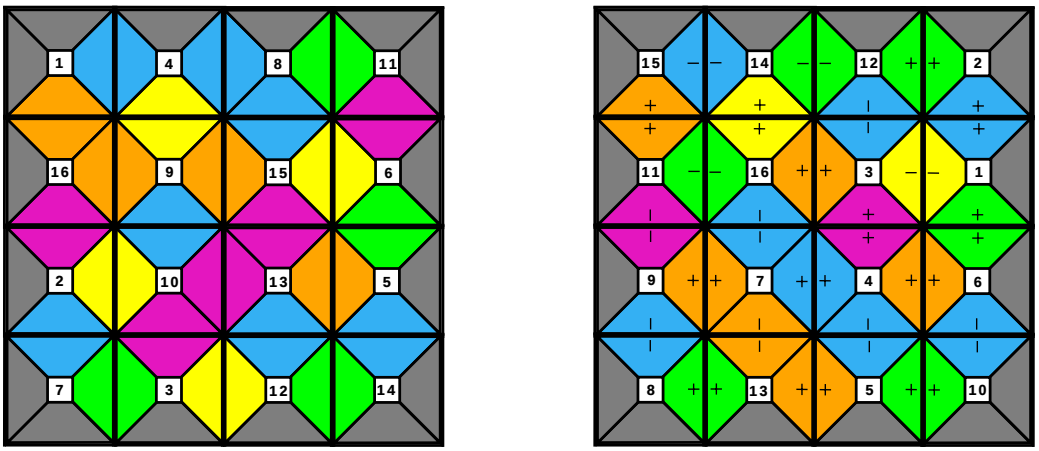


(a) (b)

Fig. 2.2: Unbounded edge-matching puzzles: no constraint on boundary edges. **a)** An example of a solved unsigned-unbounded EMP in which colour is the only edge-matching constraint. **b)** An example of a solved signed-unbounded EMP in which two constraints must be satisfied: colour and sign of abutting edges must match.

In bounded EMPs, which will be the focus of the rest of this thesis, boundary tiles are identified from a puzzle set as those having one edge (two edges) that must appear on the boundary (corner) of the solution to the puzzle. Since there are exactly $4\sqrt{n}$ boundary edges, the colour assigned for boundary edges (e.g. grey) in an $n \times n$-BEMP set occurs exactly $4\sqrt{n}$ times in the puzzle set. Bounded EMPs can also be unsigned (Figure 2.3a) or signed, (Figure 2.3b).

[2] The sign constraint can also be that abutting edges have *opposite* rather than matching signs (see [20] and [28] for examples).



(a) (b)

Fig. 2.3: Bounded edge-matching puzzles: boundary edges are constrained to a unique colour (grey)**. a)** An example of a solved unsigned-bounded EMP in which colour only is the edge-matching constraint. **b)** An example of a solved signed-bounded EMP in which colour and sign are the edge-matching constraints. Note that boundary edges are not signed since they must appear on the boundary and as such cannot abut with other edges.

It is worth noting that signed EMPs can simply be treated as having one edge-matching constraint that is the sign-colour compound attribute. Thus, in a signed EMP, two edges of the same colour but with different signs can be treated as two distinct edges each identified by a unique attribute that is the colour-sign combination. For example, in Figure 2.3b, green edges with (-) sign can be treated as one set of edges whose attribute is (-green) while green edges with (+) sign are treated as one set of edges whose attribute is (+green). Thus if E_1 E_2 are *signed* and *unsigned* EMPs, respectively, with edge colours of both drawn from a set of colours S, then the *attribute set* is S and $S \times \{+, -\}$ for E_1 and E_1, respectively. Therefore, from a solution approach perspective, unsigned and signed EMPs can be treated the same, as the latter differs by just having a larger attribute set.

In the case of bounded vs. unbounded EMPs, however, the boundary constraint presents a significant contrast between the two types and the solution approach to both would differ in fundamental strategies. For example, any solution approach to a bounded EMP would readily discard any partial solutions in which a grey-coloured edge is placed anywhere on the

board other than the boundary. Such a heuristic cannot be used, however, with unbounded EMPs since there is no hint as to what edges should ultimately appear on the boundary. Nonetheless, and from a computational complexity perspective, the two types (in fact, all variants of EMPs) belong to the same complexity class, as will be detailed later (see 2.1.3.2).

2.1.3 Computational Complexity of Edge-Matching Puzzles

2.1.3.1 Edge-Matching Puzzles and the Domino Problems

Unit square tiles with attributed edges such as those used in EMPs have in fact been the subject of a different kind of mathematical inquiry, under a class of problems known as the Domino Problems, first discussed by logician Hao Wang in 1961 [60]. A finite set $T = \{t_1, t_2, .., t_i\}$ where $t_k = \{c_n, c_w, c_s, c_e\},\ 1 \leq k \leq i,$ is an *oriented* square tile whose northern, western, southern and eastern edges are attributed, respectively, with colours $c_n, c_w, c_s, c_e \in C$, where C is a finite set of colours. Assuming there are available infinite copies of each tile (or domino) $t_k \in T$, and copies of tiles in T can be used to cover the infinite $\mathbb{Z} \times \mathbb{Z}$ grid plane observing the following conditions:

a) Tiles may not be reflected (i.e. tiles must be facing upward the plane).

b) Tiles may not be rotated (i.e. each tile has an assigned orientation and must always be placed in that orientation; hence tiles are moved on the plane by translation only).

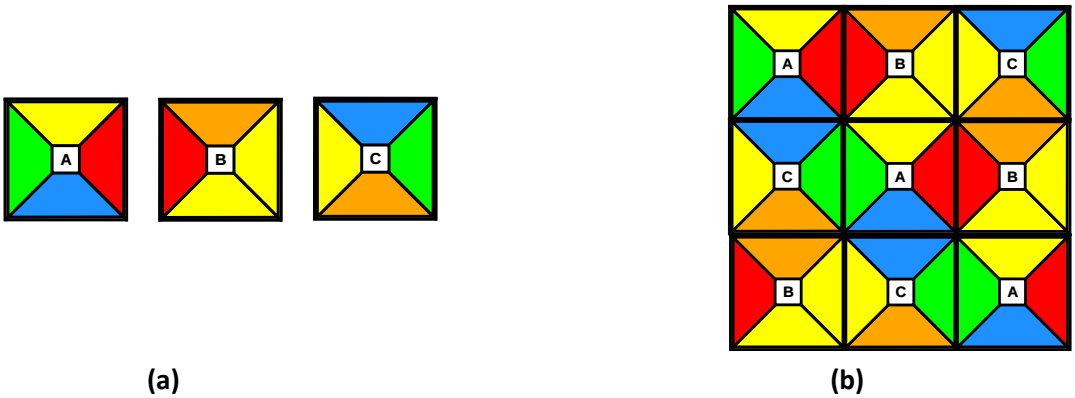
c) All abutting edges must match in colour.

then T is said to be ***solvable***. The Domino Problem is therefore stated as follows: given a finite set of tiles T, is T solvable?

Wang [60] conjectured that if T can tile the infinite plane in the manner described above (i.e. T is solvable) then it must do so *periodically*. In other words, if T is solvable then there must exist a rectangle R, whose unit area is one tile and whose dimensions are a and b

where $a \times b \leq |T|$, such that copies of R can fill the infinite plane. Informally stated, edges on the top (right) boundary of R match opposite edges on the bottom (left) boundary, and as such copies of R can be stacked verticall and horizentally to cover the entire plane while meeting the tiling constraints –namely, edge-matching, no-rotation, and no-reflection constraints.

If Wang's conjecture were true, then there would exist an algorithm for solving any given tile set T: simply find a rectangular arrangement R of *all or some* tiles in T where top (right) edges of R match bottom (left) ones. Copies of R can then tile the infinite plane and, and hence T is solvable [60]. As a consequence, the Domino Problem would be *decidable:* i.e. there would exist an algorithm which can determine whether a given set of dominoes is solvable. Also, if the conjecture is true then another consequence would be that *aperiodic* tiling of the infinite plane is impossible.



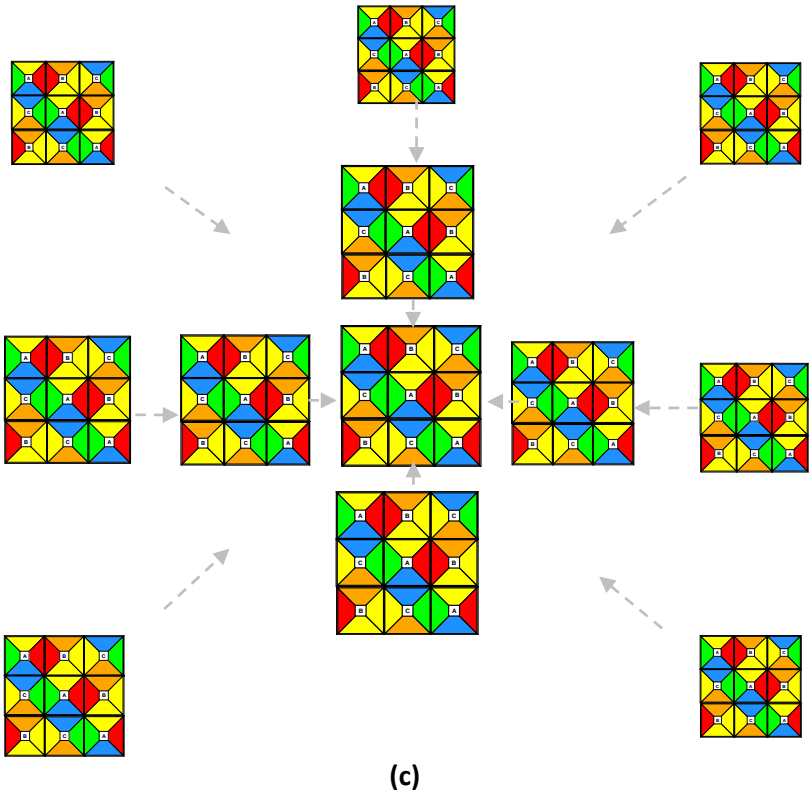


Figure 2.4: Wang's original periodic tile set. a) The three tiles used by Wang [60] when he conjectured that a set of tiles can tile the plane only periodically. Tiles have orientation (north being the top of the page) and cannot be rotated or reflected. **b)** A (3, 3)-periodic rectangle comprised of copies from the set in (a). Note that abutting edges in the rectangle all match, and that edges on the top (right) match those on the bottom (left). **c)** Tiling of the infinite plane by copies of the (3, 3)-periodic rectangle from (b). Note that the tiling is carried out by translation only, no rotation of reflection is allowed, whether of tiles in constructing the periodic rectangle or of the rectangle itself.

Berger [12] proved that Wang's conjecture is in fact false. He proved the existence of a set of tiles that can cover the infinite plane *only aperiodically.* Berger's set of over 20,000 tiles (which he reduced to 104 tiles shortly after) can provably tile the infinite plane but *only aperiodically*. It was therefore concluded that the Domino Problem is *undecidable:* i.e. there

does not exist an algorithm which can decide whether a given set of tiles is solvable or not [12]. Interestingly, even smaller sets of aperiodic Wang tiles have since been discovered. Kari [29] and Culik [19] (Figure 2.5) demonstrated aperiodic sets of 13 and 14 Wang tiles, respectively.

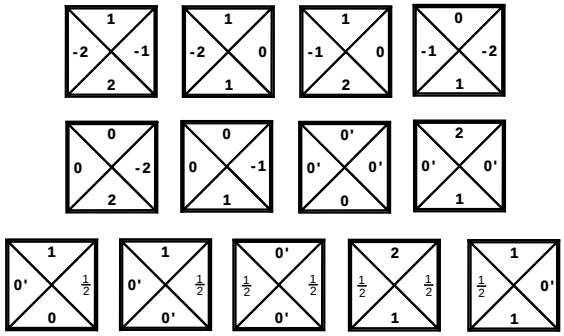


Figure 2.5: Culik's aperiodic tile set. Since Berger's proof of the undecidability of the domino problem and his demonstration of a set of over 20,000 tiles that can cover the infinite plane *only* aperiodically, smaller aperiodic sets of Wang tiles have since then been discovered, of which the 13 tiles shown in this figure, demonstrated by Culik [19].

In 1962 Wang [59] also proved that there exists a set of tiles the growth of which can simulate the execution history of any Turing machine. Further, he showed that the halting problem is reducible to the origin-constrained domino problem. It is based on Wang's investigation that the idea of Turing-complete computations using self-assemblies of DNA molecular tiles was used by Winfree to demonstrate a Turing-complete model of DNA Computing [63]. The fundamental challenge for such model is that the task of finding tile sets is itself computationally hard [62], and so the model remains far from practical realization. However, the mere computational approach to DNA 2-dimensional self-assembly has provided rigorous framework for other avenues such as DNA 2-dimensional construction of nano-scale lattices and shapes [61].

How do bounded edge-matching puzzles (BEMP) in particular relate to the domino problem? A creator of a $\sqrt{n} \times \sqrt{n}$-BEMP can in fact state the puzzle as follows: Given n colour-attributed Wang tiles, with a special boundary colour c that is attributed to exactly $4\sqrt{n}$ edges,

find a $(\sqrt{n}, \sqrt{n})$-periodic square with *c* occuring only on the boundary of the square. Copies of a solved $\sqrt{n} \times \sqrt{n}$ −BEMP can obviously tile the infinite plane, since its top, bottom, left and right boundaries have the same colour *c* and so any two copies can be adjoined top to bottom or left to right. However, a BEMP problem is technically *not* a domino problem. Firstly, while reflection of tiles is prohibited in both BEMP and the domino problem, rotation is however allowed in BEMPs. Secondly, BEMPs have a finite multi-set of Wang tiles that *can* contain duplicates. In contrast, a domino problem has a finite set of *unique* Wang tiles but with presumably infinite *copies* of each tile available. While the domino problem is provably undecidable, the BEMP problem is obviously decidable since the target tiling is not the infinite plane but rather a finite area, and as such a BEMP problem is decidable; one can simply attempt all possible permutations of tiles within that finite area.

2.1.3.2 Edge-Matching Puzzles are NP-Complete

The computational complexity theory concerns itself with the following question: how *hard* is it for an *algorithm* to find a solution to *any* given instance of a particular problem? The "hardness" of a problem is characterized by how much resources are needed by the algorithm solving it –if such an algorithm exists. The most important measures of resource consumption are *time* (how many steps the algorithm must take to achieve its task) and/or *space* (how much information the algorithm must memorize as it proceeds). In the context of modern digital computers, one can think of an *algorithm* as a computer program, *time* as the number of CPU steps taken by the algorithm, and of *space* as the computer memory needed by the algorithm. Resource consumption is typically characterized as a function of the problem's input size (e.g. if the problem is to sort n numbers, then the input size is n).

There are three aspects of computational complexity theory which are relevant to our discussion: non-determinism, polynomial-time, and completeness. Informally stated, an algorithm is said to be *non-deterministic* if, at one or more of its steps, there are more than one possible "next-step". Given a decision problem (i.e. a problem whose solution is "yes" or "no") with input size n, a non-deterministic algorithm is said to operate in *polynomial time* if its "yes" output can be *verified* using another algorithm that takes at most $p(n)$ to complete its task, and $p(n)$ is some polynomial on input size n (e.g. $p(n) = n^2 + 2n + 1$).

The two concepts of non-determinism and polynomial-time verifiability together define the complexity class NP (non-deterministic polynomial-time), one of the most fundamental classes in complexity theory. Note that polynomial-time *verifiability* does not necessarily imply polynomial-time *solvability* [24]. Hence, it may take a non-deterministic algorithm an exponential amount of time to *reach* a "yes" answer to a decision problem, but as long as its answer can be verified in polynomial time, then it belongs to NP.

A decision problem is called NP-Complete if 1) It is in NP, and 2) any other problem in NP can be reduced to it in polynomial time. The polynomial-time reduction of any problem in class NP to any NP-Complete problem implies that NP-Complete problems are as hard as the hardest problem in NP (because the hardest problem is polynomial-time reducible to them). There exists no known algorithm that can efficiently solve NP-Complete problems; the resources needed to reach an answer grow exponentially as the input size grows linearly [24]. It has even been argued that physical reality may impose a limit on how fast NP-Complete problems can ever be solved [1]. The edge-matching puzzle problem has in fact been proven to be NP-Complete (see [24] p. 257). Hence, any approach to solving an instance of an NP-Complete problem, such as EMPs, must take serious consideration to their inherent intractability.

2.2 Formal Definition of Bounded Edge-Matching Puzzles

Bounded Edge-Matching Puzzles will henceforth be the focus of our discussion. We begin in this section by providing formal definitions and precise characterizations of BEMPs. BEMPs and their solutions are formally defined with square tiles as their objects. These theoretical formalizations will be the foundation for the solution approach in the next chapter.

2.2.1 Definition of a BEMP:

A BEMP can in its essence be described as a problem of arranging a set of 2-Dimensional *objects* in a 2-Dimensional *spatial domain* in such a way that their *relative* positions meet certain constraints. A formal method of dealing with BEMPs is needed in order to facilitate qualitative and rigorous reasoning, not only about the objects themselves, but also about their inter-relation in the spatial domain (i.e. their relative *positions* and *orientations*). In other words, formal methods are developed in order to meaningfully describe 1) the inter-relation of edges within a tile (i.e. meaningful description of a tile's *orientation*), and 2) the inter-relation between tiles within a tiling grid (i.e. meaningful description of tiles' relative *positions* on a tiling grid).

In abstract set-theoretic terms, one can formalize a BEMP problem in terms of *sets* and some *relations* between elements in those sets. These two concepts of set theory are used to define yet another fundamental concept: *order,* which will provide the foundation of our formal definition of tiles[3]. We start by providing definitions of *order* in terms of *binary* relations on sets, as a prologue to subsequent definitions in terms of *ternary* relations:

Definition 2.2.1.1:
A (strictly) partially ***ordered set*** (S, R) *is a set* S *and a binary relation* $<$ *on* S *which is:*

- *irreflexive:* $\forall x \in S,\ (x, x) \notin R$, *and*
- *transitive:* $\forall x, y, z \in S, (x, y) \in R \wedge (y, z) \in R \Rightarrow (x, z) \in R$

We say that R *is* ***a partial order*** *on* S.

[3] For readers not familiar with set theory, see [30] for example.

Definition 2.2.1.2:
*A **totally ordered set** (S, R) is a set S and a binary relation R that is irreflexive, transitive, and:*

- *asymmetric:* $\forall x, y \in S, (x, y) \Rightarrow (y, x) \notin R$

*We say that R is **a total order** on S.*

And so binary relations of certain properties are used to define orders on sets of interest. For example, the *less-than* relation is a total order on the set of natural numbers and thus we say that $(\mathbb{N}, <)$ is a totally ordered set. While such binary relations might be useful for describing objects "on a line", they cannot provide meaningful information for "circular" sets [39]. Consider the set of four points on a circle, as depicted in Figure 2.6a. Even if a clockwise order is imposed, it is impossible for a binary relation to capture, for instance, the asymmetric property of a totally ordered set: both $(A, C) \in R$ and $(C, A) \in R$ can hold for $R =$ "*reached-before*" if the starting points were A and C, respectively.

Edges of a BEMP tile can be represented as points on a circle. Consider a Wang tile with edge colours c_1, c_2, c_3, c_4 depicted in Figure 2.6b. Each edge can be represented as a point on a circle, and the same argument applies: the relation of one edge relative to another edge of the same tile cannot be precisely defined without engaging a third edge as a reference.

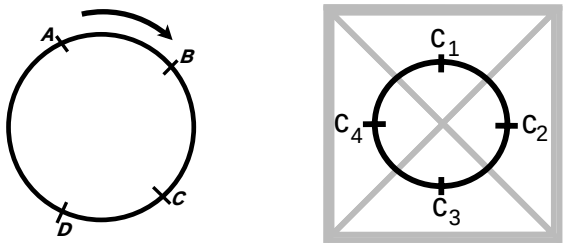


Figure 2.6: Circular sets and binary relations. a) A set of four points representing cities in a tour. Even if clockwise direction is imposed, the binary relation R = "comes-before" holds for both (ARC) and (CRA) if the starting points were A and C, respectively. Hence, asymmetry cannot be meaningfully defined on R. b) Edges of an EMP tile represented as points on a circle. The same conclusion of (a) can thus be inferred: binary relations are not sufficient to describe meaningful asymmetry on a set containing tile edges as elements.

What if circular sets are "linearized" by fixing an element as a reference? Consider for instance, a set (S, R) where $S = \{spring, summer, fall, winter\}$ and R is the binary relation *comes-before*, then one can say that $(spring, fall) \in R$ if the current season is *winter*, but $(fall, spring) \in R$ also holds if the current season is *summer;* hence, again, asymmetry is impossible to define meaningfully with a binary relation. However, if, say, *winter* is fixed as the starting point, then one can say that $(fall, spring) \in R$ but $(fall, spring) \notin R$. But R now has three operands: the reference element, and the two elements on which R is to be applied; hence R is now called a *ternary relation.*

Ternary relations will be the basis for our approach to formalizing tiles in a BEMP problem. Before defining tiles, we begin with some preliminary definitions:

Definition 2.2.1.3:
*A **ternary relation** R on a set S is a triplet $(a, b, c) \in S^3$. R is said to be:*

- *irreflexive* *iff* $(a, b, c) \in R \Rightarrow b \neq c$
- *asymmetric* *iff* $(a, b, c) \in R \Rightarrow (a, c, b) \notin R$
- *transitive* *iff* $(a, b, c) \in R, (a, c, d) \in R \Rightarrow (a, b, d) \in R$
- *cyclic* *iff* $(a, b, c) \in R \Rightarrow (b, c, a) \in R$
- *complete* *iff* $a, b, c \in S,\ a \neq b \neq c \neq a \Rightarrow (a, b, c) \in R \vee (c, b, a) \in R$

Definition 2.2.1.4:

*A **cyclic order** C on a set S is a ternary relation on S that is irreflexive, asymmetric, transitive and cyclic. We say that (S, C) is a **cyclically ordered set**.*

Definition 2.2.1.5:
*A **total cyclic order** C on S is a ternary relation on S that is irreflexive, asymmetric, transitive, cyclic, and complete. We say that (S, C) is a **totally cyclically ordered** set.*

Definition 2.2.1.6:
*Let $C_1 = (x_1, x_2 \dots, x_b), C_2 = (y_1, y_2 \dots, y_b)$ be two total cyclic orders on $S = \{h_1, h_2 \dots . h_b\}$. We say that C_1 and C_2 are **cyclically equivalent** if* $\exists k, 1 \leq k \leq b : j = (i + k)\ mod\ (b + 1) \Rightarrow x_j = x_i$ [35b]

Lemma 2.2.1.7:
*Let $(S, C) = \{e_1, e_2, e_3, e_4\}$ be a totally cyclically ordered set. There exist exactly four **quadruples** of cyclically-equivalent total orders o_1, o_2, o_3, o_4 on S, namely $o_1 = (e_1, e_2, e_3, e_4), o_2 = (e_2, e_3, e_4, e_1), o_3 = (e_3, e_4, e_1, e_2), o_4 = (e_4, e_1, e_2, e_3)$.*

Proof: Obvious. ■

Definition 2.2.1.8:
*Let (S, C) be a totally cyclically ordered set, $|S| = 4$, and $\boldsymbol{O_s} = \{o_1, o_2, o_3, o_4\}$ be the set of all quadruples of cyclically-equivelant orders on S. $\boldsymbol{O_s}$ is called the **set of orientations** of S. $o_i \in \boldsymbol{O_s}$ is refered to as **orientation** i of S, or simply S_i, where $1 \leq i \leq 4$.*

Definition 2.2.1.9:
*A **tile** t is a 4-tuple $\langle \Sigma, S, C, O \rangle$ where:*

- *Σ is a finite alphabet of attributes*
- *$S = \{a, b, c, d\}$ is a multi-set and $a, b, c, d \in \Sigma^4$,*
- *C is a total cyclic order on S, and*
- *O is the set of orientations of S*

Definition 2.2.1.10:
*A **puzzle set** P is collection of n tiles $t_1, t_2, \ldots t_n$ where $\sqrt{n} \in \mathbb{N}$.*

Having formally described the *objects (i.e. tiles)* of BEMPs, we now proceed to establishing formal definitions of their *spatial domain* (i.e. tiling grid).

Definition 2.2.1.11:
*Given a puzzle set $P = \{t_1, t_2, \ldots t_n\}$, a **tiling grid** $G(P)$ is a finite $\sqrt{n}x\sqrt{n}$ grid graph whose corners are coincedental to $(0,0), (0, \sqrt{n}), (\sqrt{n}, 0), (\sqrt{n}, \sqrt{n})$ coordinate points of the the first quadrant of the Cartesian plane.*

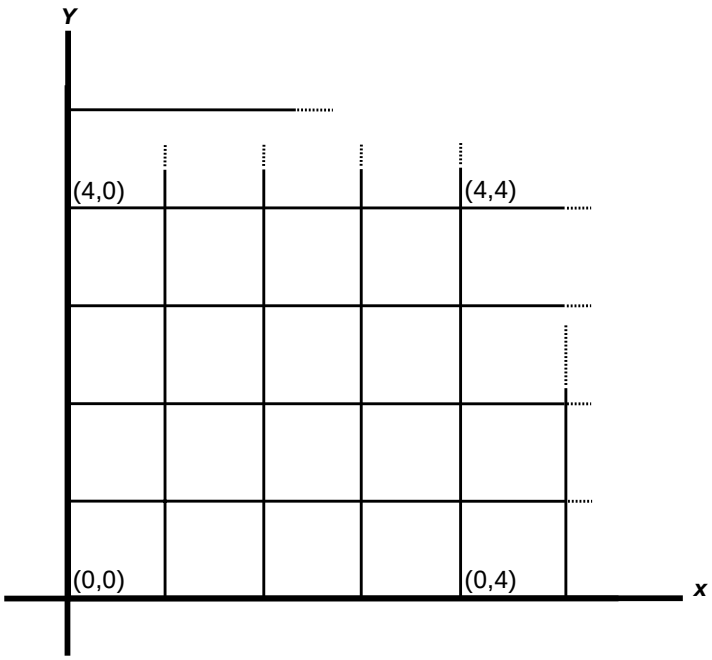


Figure 2.7: A tiling grid for a puzzle set of $|N| = 16$ tiles. The tiling grid of $G(P)$ is a fininte grid graph whose corners coincide with (0,0),(0,4),(4,0) and (4,4) points of the infinite grid in the first quadrant of the Cartesian plane.

Definition 2.2.1.13:
*Given a tiling grid $G(P)$, a tile **placeholder** is a unit square $h_r = \square v_1 v_2 v_3 v_4$ where $v_1 = (x, y)$, $v_2 = (x, y+1)$, $v_3 = (x+1, y+1)$, $v_4 = (x+1, y)$ for $0 \leq x, y \leq \sqrt{n} - 1$.*
*We denote $H = \{ h_0, h_1, .., h_{n-1}\}$ the **set of placeholders** of P. For any given placehoder h_r, we use $W(h_r), N(h_r), E(h_r)$, and $S(h_r)$ to refer to edges of h_r bounded by $(v_1, v_2), (v_2, v_3), (v_3, v_4)$ and (v_4, v_1) respectively.*

It follows from Definition 2.2.1.13 that $0 \leq r \leq \sqrt{n}(\sqrt{n} - 1) + \sqrt{n} - 1 \Rightarrow 0 \leq r \leq n - 1$. Figure 2.8 shows a tiling grid for a puzzle set P where $|P| = n = 16$ tiles, with placeholders h_0 to h_{15}. The inset illustrates an example of a placeholder: referencing individual edges of a placeholder facilitates establishing a relation between tiles and placeholders, as will be laid out in subsequent definitions.

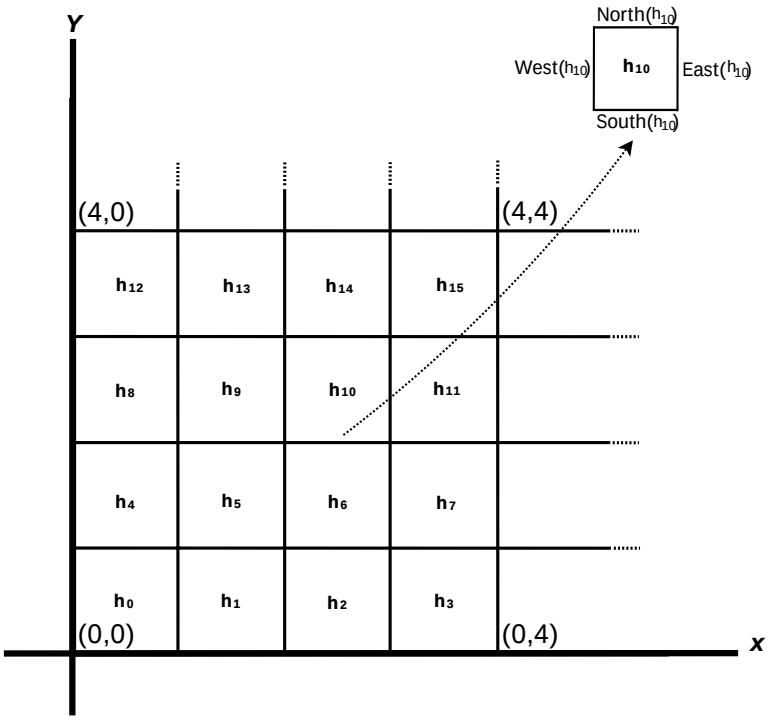


Figure 2.8: Placeholders in a tiling grid for a puzzle set of $|P| = 16$ tiles.
Placeholders are the squares making up the tiling grid graph. The systematic numbering of placeholder squares facilitates referencing each edge of each square. The inset shows an example placeholder, whose southern edge for example is bounded by (2,2) and (3,2) vertices of the grid.

Now that the objects and the spatial domain are defined, we can proceed to defining an instance of a BEMP as a function from the set of placeholders (i.e. the domain of the function) to

the set of all orientations of tiles (i.e. the codomain of the function). We will sometimes use the notation $\boldsymbol{t_{i/j}}$ to refer to orientation j or tile t_i.

Definition 2.2.1.14:
Given a puzzle set $P = \{ t_1, t_2, \dots t_n\}$ and placeholder set $H = \{ h_0, h_1, .., h_{n-1}\}$ defined on tiling grid $G(P)$, for any $t_i \in P$ and $o_j \in O_{t_i}$, let:

- *$t_{i/j}$ denote orientation o_j of tile t_i for any $i, j, 1 \leq i \leq n$ and $1 \leq j \leq 4$, and*
- *$\Theta = O_{t_1} \cup O_{t_2} \cup , .., \cup O_{t_n}$ be the superset of all orientations of all tiles.*

*A **placement** of a tile t_i in placeholder h_r is a function $\boldsymbol{p}$: $\boldsymbol{H \rightarrow \Theta}$. $p(h_k) = t_{i/j} \Rightarrow N(h_k) = e_n$, $E(h_k) = e_e$, $S(h_k) = e_s, W(h_k) = e_w$ where $(e_n, e_e, e_s, e_w) = o_j \in O_{t_i}$. We say that t_i is **placed** in h_k in orientation o_j.*

In other words, when a tile t_i is placed in a placeholder h_k in one of its four orientations $o_j \in O_{t_i}$ (see Definition 2.2.1.8 and 2.2.1.9), then northern, western, southern and eastern edges of h_k are (by definition) coincidental to the first, second, third and fourth element (i.e. edge) of o_j respectively. Notice that Definition 2.2.1.14 defines the *domain* and *range* of the placement function p, but does not provide the third ingredient of a valid function: the *rule* that maps a unique $t_{i/j} \in \Theta$ to each $h_k \in H$. This is because defining p is precisely the solution to a BEMP, as we see the following definitions.

Definition 2.2.1.15:
*A **bounded edge-matching puzzle** is a 7-tuple $\langle P, \Sigma, b, \tau, \Theta, G, H \rangle$ where:*

- *$P = \{ t_1, t_2, \dots t_n\}$ is as set of tiles all defined over an alphabet Σ and $\sqrt{n} \in \mathbb{N}$*
- *$b \in \Sigma$ is a **boundary** attribute*
- *$\tau = \{a_1, a_2, a_3, a_4\}$ is the **seed** tile where $\exists a_i, a_{i+1(mod4)} \in \tau$: $a_i = a_{i+1(mod4)} = b$*
- *$\Theta = O_{t_1} \cup O_{t_2} \cup , .., \cup O_{t_n}$ is the orientation superset over P*
- *$H = \{ h_0, h_1, .., h_{n-1}\}$ is the set of placeholders defined on tiling grid $G(P)$*

2.2.2 Definition of a BEMP Solution:

Definition 2.2.1.16:

A solution to a BEMP *is an injective function* $p: H \rightarrow \Theta$ *such that, for any* $p(h_k) = t_{i/j}$*:*

1. $p(h_0) = \tau_i, 1 \leq i \leq 4$
2. $N(h_k) = S(h_{k+\sqrt{n}})$ *for* $0 \leq k \leq n - \sqrt{n} - 1$
3. $E(h_k) = W(h_{k+1})$ *for* $[0 \leq k < n - 1] \wedge [(k+1) mod \sqrt{n} \neq 0]$
4. $N(h_k) = b$ *for* $[n - \sqrt{n} \leq k \leq n - 1]$
5. $E(h_k) = b$ *for* $\left[\sqrt{n} - 1 \leq k \leq n - 1\right] \wedge [(k+1)\ mod \sqrt{n} = 0]$
6. $S(h_k) = b$ *for* $[0 \leq k \leq \sqrt{n}]$
7. $W(h_k) = b$ *for* $\left[0 \leq k \leq n - \sqrt{n}\right] \wedge [k\ mod \sqrt{n} = 0]$

The first constraint restricts placeholder h_0 to one of the four corner tiles (i.e. tiles with two boundary edges), referred to as the ***seed*** tile τ. Any defined solution $p: H \rightarrow \Theta$ can equivelantly be expressed in three more different manners (corresponding to a 90, 180 and 270 degrees rotation of the whole solution, see Figure 2.9). However, if h_0 is resticted to an arbitrarly-chosen corner tile, then $p: H \rightarrow \Theta$ is unique. The second and third constraints in Definition 2.2.1.16 enforce that abutting edges match, while constraints 4-7 enforce the boundary constraint of a BEMP.

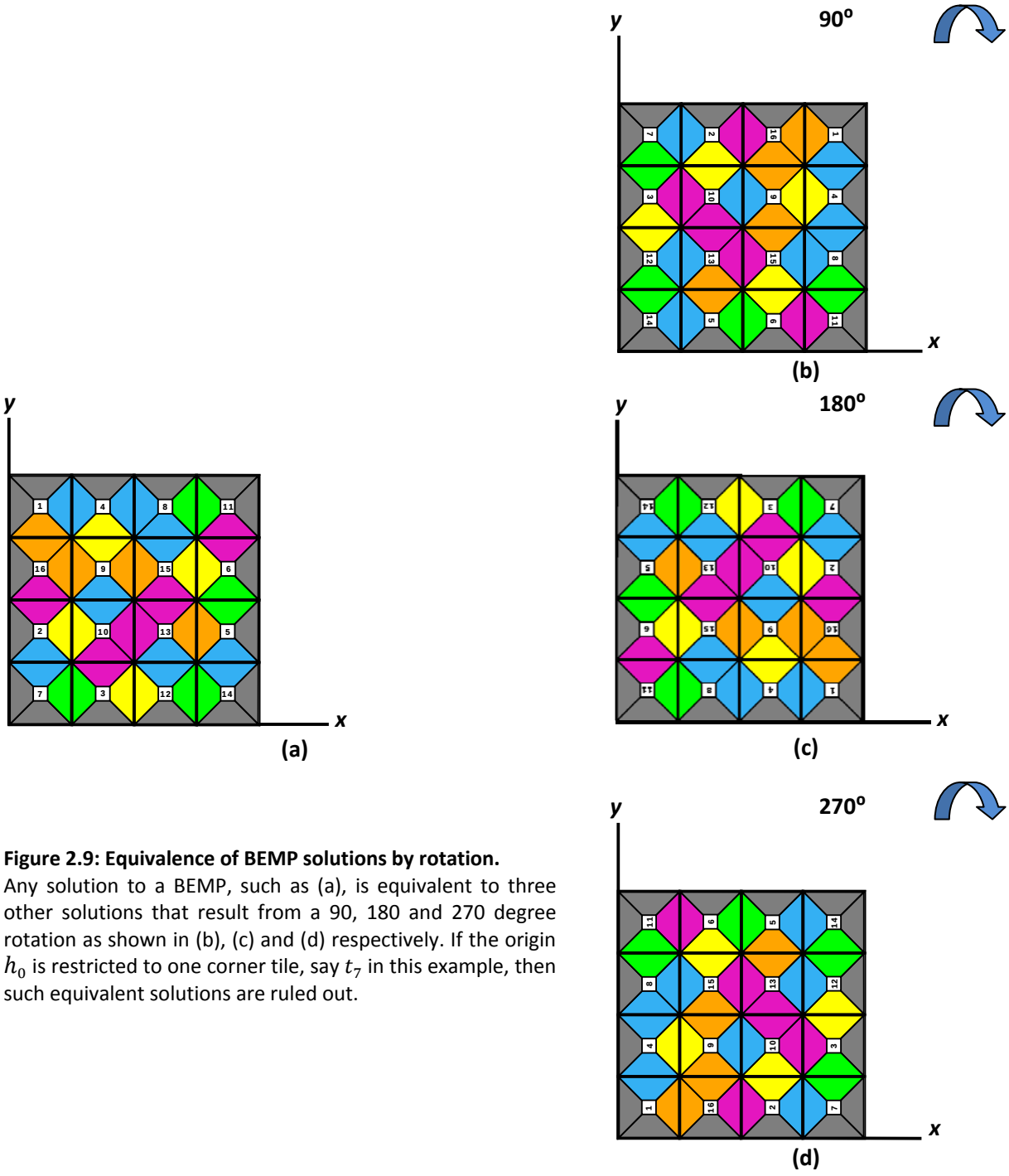


Figure 2.9: Equivalence of BEMP solutions by rotation.
Any solution to a BEMP, such as (a), is equivalent to three other solutions that result from a 90, 180 and 270 degree rotation as shown in (b), (c) and (d) respectively. If the origin h_0 is restricted to one corner tile, say t_7 in this example, then such equivalent solutions are ruled out.

2.3 Definition of a 4×4-BEMP Instance: Hypatia

Figure 2.10 shows a 16-tile puzzle set for a 4×4 bounded matching puzzle which will be the subject of our DNA-Computing protocol for solving BEMPs as will be laid out in the next two chapters. Following Definition 2.2.1.15, the definition of this particular instance, which we refer to as ***Hypatia***, is as follows:

- $P = \{ t_1, t_2, \ldots t_{16}\}$
- $\Sigma = \{red, green, yellow, blue, orange, purple, brown, grey\}$where $\boldsymbol{b} = grey$
- $\tau = t_{12}$
- $\Theta = O_{t_1} \cup O_{t_2} \cup , .. , \cup O_{t_{16}}$ [4]
- $H = \{ h_0, h_1, .. , h_{15}\}$ is the set of placeholders defined on tiling grid $G(16)$

Define function $p: H \to \Theta$, *assuming* ***1)*** p *is unique, and* ***2)*** $\nexists T \subset \Theta:$ T *tiles the 4×4 grid.*

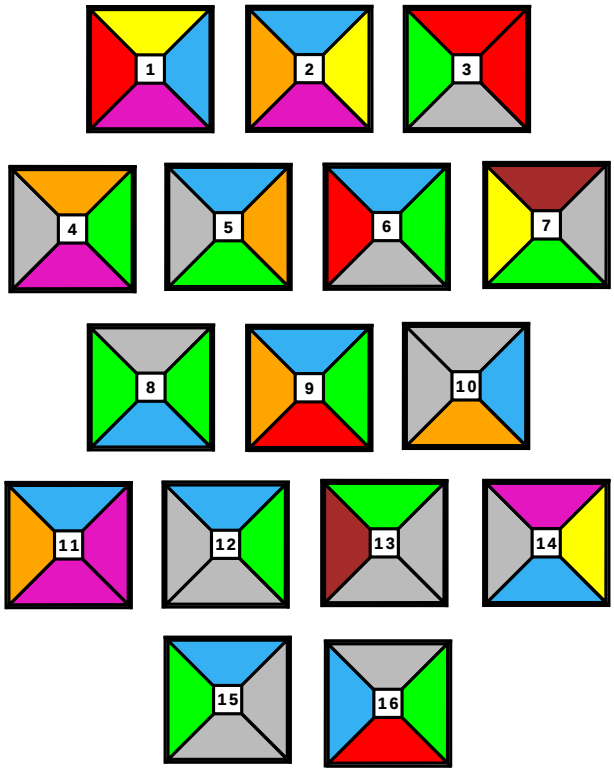


Figure 2.10: Hypatia Puzzle Set. A puzzle set of a 4×4 instance of a bounded edge-matching puzzle. The 16 tiles are edge-attributed from an alphabet of 8 colours: red, green, yellow, blue, orange, purple, brown and grey (as the boundary colour). Tiles with two grey edges are evidently corner tiles. Note that tile identifiers (1...16) are used as a reference and have no impact on the solution (typically real recreational EMPs have tile IDs printed on the back of the tiles).

Note that the choice of $\tau = t_{12}$ as the seed tile is arbitrary.

[4] For example: $O_{t_1} = \{o_1, o_2, o_3, o_4\} = \{(yellow, blue, purple, red), (blue, purple, red, yellow), (purple, red, yellow, blue), (red, yellow, blue, purple)\}$

Chapter 3 Solution Approach

A backtracking algorithm for solving BEMPs is presented in this chapter in abstract terms. We begin in 3.1 with an outline of the algorithm, followed by a formal treatise in 3.2 and 3.3 which builds upon the definitions of Chapter 2. Clearly, the presented algorithm is designed with the DNA-based implementation in mind, so we conclude this chapter with a discussion in 3.4 on the motivation behind our choice of a diagonal-wise stacking algorithm. Hypatia (the 4×4-BEMP instance defined in 3.3) will be used in particular for demonstrations throughout.

3.1 Diagonal-wise Tile Stacking:

Given the empty tiling grid $G(Hypatia)$, Θ the multi-set of orientations of Hypatia, $15 \leq w, x, y, z \leq 1$ and $4 \leq i, j, k, f, l, m \leq 1$, consider the following solution strategy:

Step 1: Find $\mathrm{E}_1 = \{\tau_i\}$, $\Phi_1 = \{t_{x/j}\} \in \Theta$ that can legally be placed in h_0, h_{15}, respectively.

Step 2: Find $\mathrm{E}_2 = \{(t_{x/i}, t_{y/j})\}, \Phi_2 = \{(t_{w/k}, t_{z/l})\} \in \Theta^2 \times \Theta^2$ that can legally be placed in $\{(h_1, h_4)\}, \{(h_{11}, h_{14})\}$ respectively.

Step 3: Find $\mathrm{E}_3 = \{(t_{x/i}, t_{y/j}, t_{z/k})\}, \Phi_3 = \{(t_{w/f}, t_{p/l}, t_{q/m})\} \in \Theta^3 \times \Theta^3$ that can legally be placed in (h_2, h_5, h_8), (h_7, h_{10}, h_{13}) respectively.

Step 4: Find $\mathrm{E}_4 = \Phi_4 = \{(t_{x/i}, t_{y/j}, t_{z/k}, t_{w/f})\} \in \Theta^4$ that can legally be placed in (h_3, h_6, h_9, h_{12}) respectively.

Step 5: Print G.

At each step, multi-sets of tiles are placed on parallel and symmetrical "diagonal" placeholders in the tiling board. The algorithm proceeds diagonal-wise until the last diagonal multi-set is successfully placed. Figure 3.1 illustrates a successful progression of the algorithm, showing a snapshot of the tiling grid after each successful step. Notice that Step $s + 1$ is not begun until Step s is successfully completed.

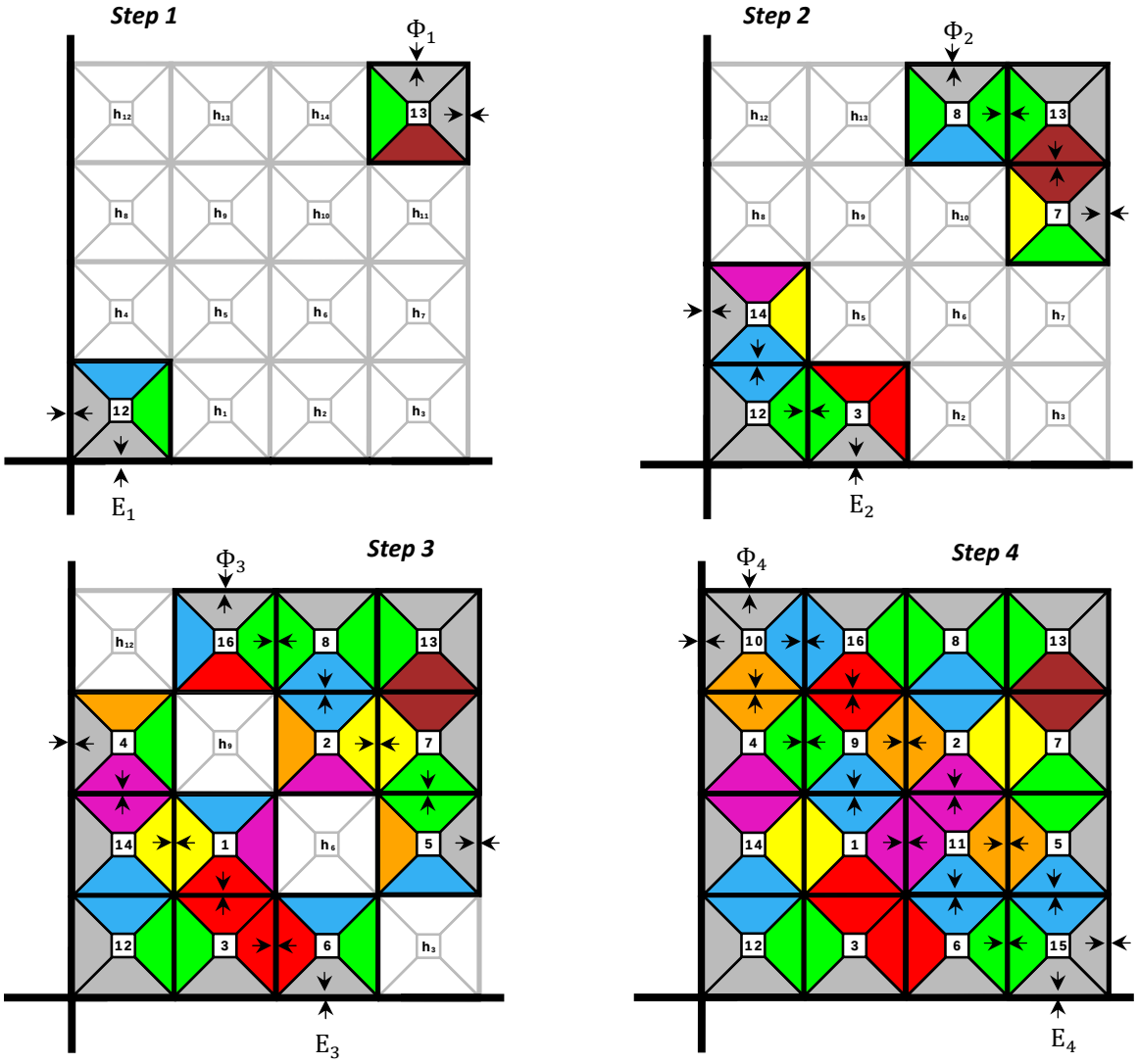


Figure 3.1: **Diagonal-wise tile stacking**.
Step-by-step progression of the diagonal-wise tile stacking algorithm. The arrows indicate edges that are relevant to edge-matching/boundary constraints at any given step. For example, placing $h_0 = (bl, gn, gr, gr) \in O_{12}$ in Step 1, the southern/western edges (gr, gr) are relevant while the northern/eastern edges (bl, gn) have no implications (it is only in Step 2 that they are relevant, namely when validated against western, southern edges of $(rd, rd, gr, \boldsymbol{gn}) \in O_3$ and $(pr, yl, \boldsymbol{bl}, gr) \in O_{14}$, respectively). Notice that t_{12} is Hypatia's seed tile and must therefore be placed in h_0. $\mathrm{E_i}/\Phi_\mathrm{i}$ denotes the diagonal set of tiles at each step, notice that $\mathrm{E}_4 = \Phi_4$.

It is instructive to recall the convention established in Definition 3.2.1.14 where $p(h_r) = t_{i/j}$ implies that the northern, eastern, southern and western edges of placeholder h_r are assigned the first, second, third and fourth elements (i.e. colours) of orientation o_j of tile t_i.

In Step 1 for example, recall from 3.3 that $\tau = t_{12} = \{bl, gn, gr, gr\}$[1] was designated as the seed tile of Hypatia. Given that $O_{12} = \{o_1 = (\, bl, gn, gr, gr), o_2 = (gn, gr, gr, bl), o_3 = (gr, gr, bl, gn),\ o_4 = (gr, bl, gn, gr)\}$ is the set of orientations of t_{12}, then clearly $p(h_0) = t_{12/1}$. Tile edges that are relevant to edge-matching and boundary constraints in each step are marked with arrows in Figure 3.1. Notice that, in Step 1-3, only two edges of a tile are *validated* against edge-matching/boundary constraints, while the other two are *validated against* in the next step. For example, when t_7 is placed during Step 2, only nothern (brown) and eastern (gray) edges are *validated* by the algorithm to ensure constraint-satisfaction; while the southern (green) and western (yellow) edges are *validated-against* in Step 3, namely against the northern and eastern edges of t_5 and t_2, respectively. Examining at the algorithm outlined above, some observations are in order:

1. Construction of diagonal multi-sets in this algorithm is a selection *with replacement* from Hypatia's puzzle set. As such, the algorithm does not rule out the possibility of a tile appearing in more than one diagonal multi-set (duplication).
2. The algorithm is of course non-deterministic: at step $p, 1 \leq p \leq 3$, there can exist more than one valid diagonal multi-set of tiles $\mathrm{E_i}/\Phi_\mathrm{i}$ that can legally be placed. As such, there is no guarantee that the choice at Step s is in fact the correct one ultimately.

These observations imply that the algorithm could make the wrong choice by placing a tile –though legally during that step– in a position that is not the correct one ultimately. Given that $\nexists T \subseteq \Theta:\ T$ can tile the 4×4 grid (see 2.3 and 2.1.3.2 on non-determinism) and that Hypatia's solution is unique, a partial solution containing duplications and/or globally-incorrect placements of a tile can never lead to a full tiling of the board. In such case, the stacking

[1] We use the shorthand notation
$\{rd, gn, yl, bl, ng, pr, br, gr\}$ respectivly corresponding to $\{red, green, yellow, blue, orange, purple, brown, grey\}$

algorithm is guaranteed to deadend at some subsequent step (i.e. it cannot construct a diagonal multi-set that can satisfy the edge-matching/boundary constraints in some subsequent step). Figure 3.2 illustrates a snapshot, in which the algorithm has made the wrong choice of tiles in Step 2, namely placing t_8 and t_3 in h_1 and h_{14} respectively. Although such placement is legal, and may even allow for one more legal step (notice the successful Step 3), the algorithm eventually deadends in Step 4: there exists no $(t_{x/i}, t_{y/j}, t_{z/k}, t_{w/f}) \in \Theta^4$ which can legally be placed in (h_3, h_6, h_9, h_{12}) respectively. A visual inspection of Hypatia puzzle set can reveal such impossibility: for example, when attempting to make a placement in h_6, one concludes that there exists no tile with an orientation (bl, ng, rd, ng), which is the orientation required to satisfy edge-matching constraints in h_6 —given the current partial solution.

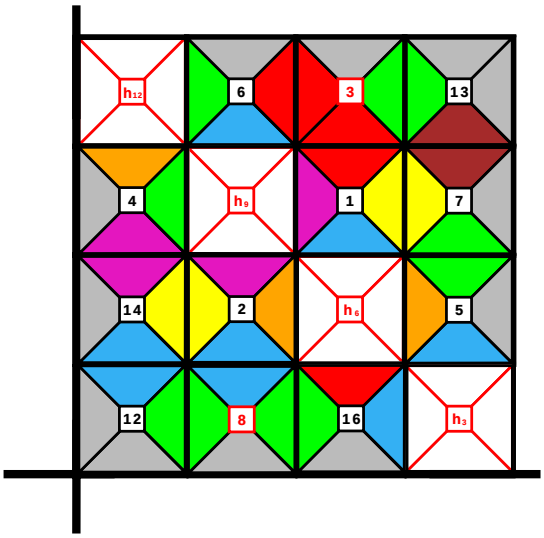


Figure 3.2: **Example execution snapshots of the diagonal-wise tile stacking algorithm in deadend state.** The algorithm selects the wrong diagonal sets in Step 2 $(t_8, t_{14}), (t_7, t_3)$. Although the placement is legal in this step, and even though Step 3 was also successful, the algorithm ultimately deadends during Step 4: there exists no 4-multiset of tiles that can be placed in (h_3, h_6, h_9, h_{12}) while satisfying all edge-matching/boundary constraints. Visual inspection of Hypatia would reveal, for example, that there is no $(bl, ng, rd, ng) \in \Theta$, which is the required orientation for a successful placement in h_6, considering the particular partial solution up to Step 3.

The algorithm must therefore *backtrack* when reaching a deadend state and try a different selection of diagonal multi-sets. Finding the solution using such an algorithm can thus be characterized as a "series of correct selection decisions of diagonal multi-sets from Step 1 to Step 4". In what follows, we outline a more robust version of the algorithm, incorporating the backtracking mechanism:

Algorithm 3.1.1:

$Let\ 0 \le s \le 15;\ 1 \le q,r,w,x,y,z \le 16;\ 1 \le f,i,j,k,l,u \le 4$
$G(Hypatia) = \phi \Rightarrow \forall h_s \in G: h_s = \phi;\ G(Hypatia) = SOLVED \Rightarrow \forall h_s \in G: h_s \neq \phi;$

$\boldsymbol{Solve}(\boldsymbol{G}, \boldsymbol{1});$
$Print\ (G);$

$\boldsymbol{Solve}(\boldsymbol{G}, \boldsymbol{step})\{$

$\boldsymbol{switch}(\boldsymbol{step})\{$

$\boldsymbol{case\ step} = \boldsymbol{1}$:

$Let\ \Pi = \Theta, \forall (t_{q/f}, t_{r/i}) \in \Pi^2$

IF $(h_0) = (t_{q/f})\ \wedge\ (h_{15}) = (t_{r/i})$ $\Rightarrow$ $t_{q/f} \in O_\pi\ \wedge\ [S(h_0)\quad W(h_0)] = [b\quad b]\ \wedge\ [N(h_{15})\quad E(h_{15})] = [b\quad b]$ (Seed) (ε_1) (φ_1)

THEN $\boldsymbol{Solve}(\boldsymbol{G}, \boldsymbol{2});$
IF $G == SOLVED\ \vee\ \Pi == \phi$
THEN return;
$\Pi = \Pi - \{(t_{q/f}, t_{r/i})\};$

$\boldsymbol{case\ step} = \boldsymbol{2}$:

$Let\ \Pi = \Theta^2,\ \ \forall (t_{q/f}, t_{r/i}), (t_{w/j}, t_{x/k}) \in \Pi^2$

IF $(h_1, h_4) = (t_{q/f}, t_{r/i})\ \wedge\ (h_{11}, h_{14}) = (t_{w/j}, t_{x/k})$ $\Rightarrow$

$\left.\begin{matrix}[S(h_1)\quad W(h_1)] = [b\quad E(h_0)]\ \wedge \\ [S(h_4)\quad W(h_4)] = [N(h_0)\quad b]\end{matrix}\right\}$ (ε_3)

$\left.\begin{matrix}[N(h_{11})\quad E(h_{11})] = [S(h_{15})\quad b]\ \wedge \\ [N(h_{14})\quad E(h_{14})] = [b\quad W(h_{15})]\end{matrix}\right\}$ (φ_3)

THEN $\boldsymbol{Solve}(\boldsymbol{G}, \boldsymbol{3});$
IF $G == SOLVED\ \vee\ \Pi == \phi;$
THEN return;
$\Pi = \Pi - \{(t_{q/f}, t_{r/i}), (t_{w/j}, t_{x/k})\};$

$\boldsymbol{case\ step} = \boldsymbol{3}$:

$Let\ \Pi = \Theta^3,\ \ \forall (t_{q/f}, t_{r/i}, t_{w/j}), (t_{x/k}, t_{y/l}, t_{z/u},) \in \Pi^2$

IF $(h_2, h_5, h_8) = (t_{q/f}, t_{r/i}, t_{r/i})\ \wedge\ (h_7, h_{10}, h_{13}) = (t_{x/k}, t_{y/l}, t_{z/u})$ $\Rightarrow$

$\left.\begin{matrix}[S(h_2)\quad W(h_2)] = [b\quad E(h_1)]\ \wedge \\ [S(h_5)\quad W(h_5)] = [N(h_1)\quad E(h_4)]\ \wedge \\ [S(h_8)\quad W(h_8)] = [N(h_4)\quad b]\end{matrix}\right\}$ (ε_5)

$\left.\begin{matrix}[N(h_7)\quad E(h_7)] = [S(h_{11})\quad b]\ \wedge \\ [N(h_{10})\quad E(h_{10})] = [S(h_{14})\quad W(h_{11})]\ \wedge \\ [N(h_{13})\quad E(h_{13})] = [b\quad W(h_{14})]\end{matrix}\right\}$ (φ_5)

THEN $\boldsymbol{Solve}(\boldsymbol{G}, \boldsymbol{4});$
IF $G == SOLVED\ \vee\ \Pi == \phi$
THEN return;
$\Pi = \Pi - \{(t_{q/f}, t_{r/i}, t_{w/j}), (t_{x/k}, t_{y/l}, t_{z/u},)\};$

case step $= \mathbf{4}$:

$Let\ \Pi = \Theta^4, \forall (t_{q/f}, t_{r/i}, t_{w/j}, t_{x/k}) \in \Pi$

$$\texttt{IF} \quad (h_3, h_6, h_9, h_{12}) = (t_{q/f}, t_{r/i}, t_{r/i}, t_{x/k}) \Rightarrow \begin{cases} [S(h_3) \quad W(h_3)] = [b \quad E(h_2)] \wedge \\ [S(h_6) \quad W(h_6)] = [N(h_2) \quad E(h_5)] \wedge \\ [S(h_9) \quad W(h_9)] = [N(h_5) \quad E(h_8)] \wedge \\ [S(h_{12}) \quad W(h_{12})] = [N(h_8) \quad b] \end{cases} (\varepsilon_7)$$

$$\begin{cases} [N(h_3) \quad E(h_3)] = [S(h_7) \quad b] \wedge \\ [N(h_6) \quad E(h_6)] = [S(h_{10}) \quad W(h_7)] \wedge \\ [N(h_9) \quad E(h_9)] = [S(h_{13}) \quad W(h_{10})] \wedge \\ [N(h_{12}) \quad E(h_{12})] = [b \quad W(h_{13})] \end{cases} (\varphi_7)$$

$\texttt{THEN} \quad G = SOLVED; \texttt{ return;}$

$\texttt{IF} \quad \Pi == \phi;$

$\texttt{THEN} \quad \texttt{return;}$

$\Pi = \Pi - \{(t_{q/f}, t_{r/i}, t_{w/j}, t_{x/k})\}$

Notice that each tile is involved in constraint validation in two separate instances: at each instance, *half* of the tile (half = a corner of two neighbouring edges within a tile) is validated for constraint-satisfaction. Consider, for example, the placement of $h_1 = t_{q/f}$ in Step 2. Only the southern and western edges are validated at this step, the constraints being that the southern edges is *gray* and the western edge matches colour with the eastern edge of the tile that has already been placed in h_0 during the previous step; hence we say $[S(h_1) \quad W(h_1)] = [b \quad E(h_0)]$. The other *half* of $t_{q/f}$ (i.e. the northern and eastern edges) are *validated-against* during Step 3.

In the next section, we capture this notion of *half-tile* and redefine the solution as a successful stacking of *lanes* of half-tiles. In the algorithm above, these lanes are marked ε_1/φ_1,.., ε_7/φ_7 and will be referenced in the subsequent discussion. Notice the special constraint (referred to as "seed") which restricts h_0 to the seed tile.

3.2 Diagonal-wise *half*-Tile Stacking:

We begin by revisiting the snapshots of a successful progression of the algorithm, but now showing only the edges that are relevant to edge-matching/boundary constraints in a given step –and only in that step. Figure 3.3 shows the four successful steps that lead to the solution of Hypatia. Each step can be described as a successful placement of one or more *half-tiles* (a corner of two neighbouring edges within a tile). The multi-sets of half-tiles that are relevant and connected at each step are referred to as "lanes".

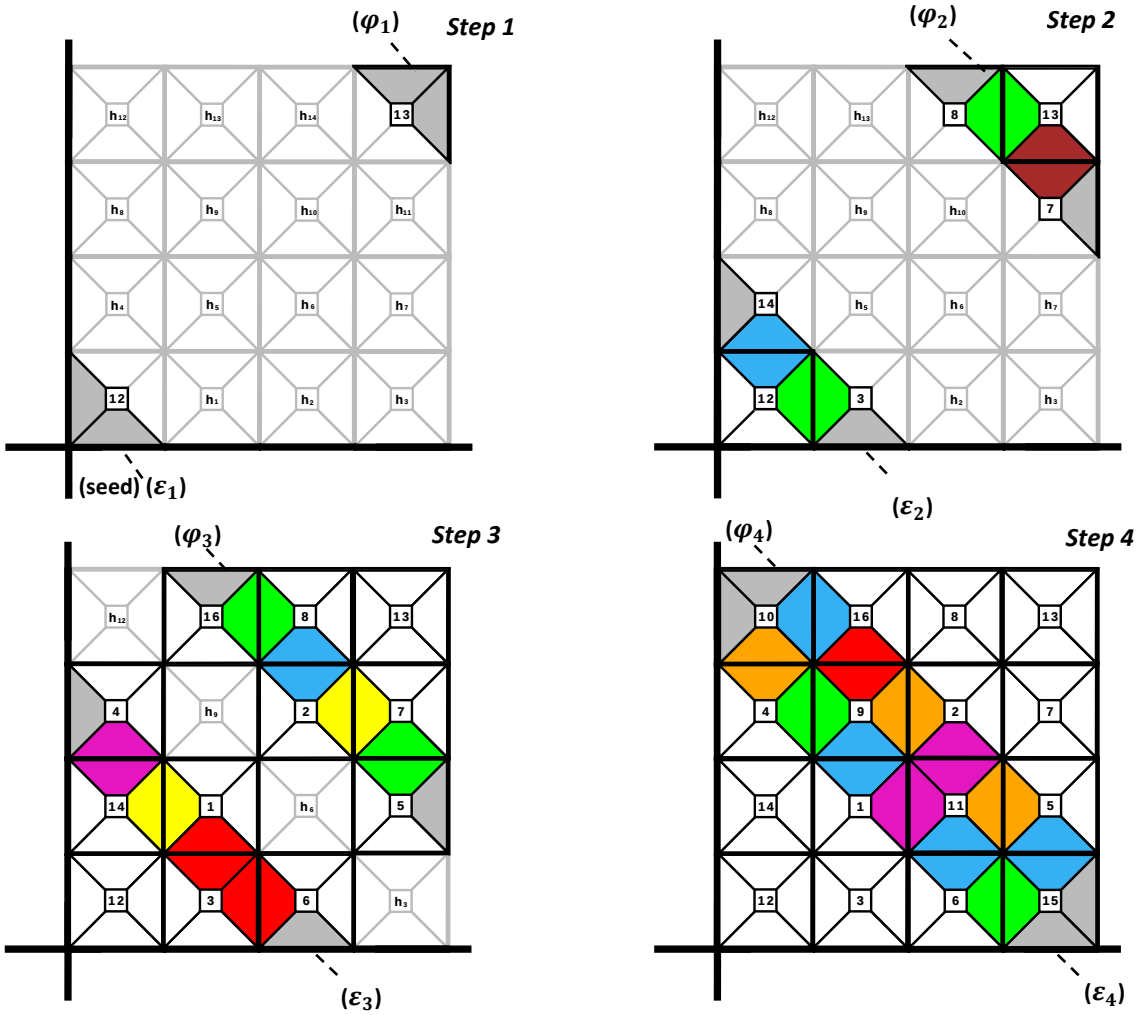


Figure 3.3: Diagonal-wise half-tile stacking algorithm. At each step, only edges that are checked for satisfaction edge-matching/boundary constraint are shown. Each step can be characterized as a successful placement of a multi-set of *half-tiles*. A connected multi-set of one or more half-tiles is referred to as a "lane", and as such there are 8 lanes $|\varepsilon_1|=|\varphi_1|=1$, $|\varepsilon_3|=|\varphi_3|=3$, $|\varepsilon_5|=|\varphi_5|=5$, $|\varepsilon_7|=|\varphi_7|=7$ half-tiles. Notice the extra constraint on the tile placed in h_0 which must be the seed tile (t_{12}).

At each step, two equal-size multi-sets (lanes) of half-tiles are selected, and if all edge-matching/boundary constraints are met, the algorithm moves to the next step. There are 8 lanes: ε_1/φ_1,.., ε_7/φ_7. At each step, constraint validation is checked for a pair of edges (see Algorithm 3.1.1). An extra constrain is imposed on ε_1 such that the half-tile placed at h_0 must belong to the seed tile (see Step 1 in Algorithm 3.1.1, and the definition of Hypatia in 3.3).

3.2.1 Representation:

A tile can therefore be represented conceptually as the union of two half-tiles along a diagonal. Consider, for example, $t_2 \in Hypatia$, and assume that t_2 is to be placed in h_i (Figure 3.4a). Clearly t_2 can be placed in one of its four orientations: $O_2 = \{o_1 = (bl, yl, pr, ng), o_2 = (yl, pr, ng, bl), o_3 = (pr, ng, bl, yl)$ and $o_4 = (ng, bl, yl, pr)\}$, as shown in Figure 3.4b.

If a tile is represented as the union of two half-tiles along the diagonal, and given that there are two diagonals in a square tile, then clearly there are two possible pairs of half-tiles the union of each produces the original tile *in one of four orientations*. We refer to such pairs as **$\boldsymbol{\alpha}$-pair** and **$\boldsymbol{\beta}$-pair**, as shown in Figure 3.4c. Notice that, in a given pair of half-tiles, each edge is destined for *either* (northern, southern) *or* (eastern, western) positions when the union operation is applied. For example, edge bl in the α-pair in Figure 3.4c assumes either the northern or southern edges when the the union operation is applied to obtain $o_1 = \alpha_1 \cup \alpha_2 = (\boldsymbol{bl}, yl, pr, ng)$ or $o_3 = \alpha_2 \cup \alpha_3 = (pr, ng, \boldsymbol{bl}, yl)$, respectively. Similarily, in β-pair, bl assumes western or eastern positions when the union operations is applied to obtain $o_2 = \beta_1 \cup \beta_2 = (yl, pr, ng, \boldsymbol{bl})$ or $o_4 = \beta_2 \cup \beta_1 = (ng, \boldsymbol{bl}, yl, pr)$, respectively.

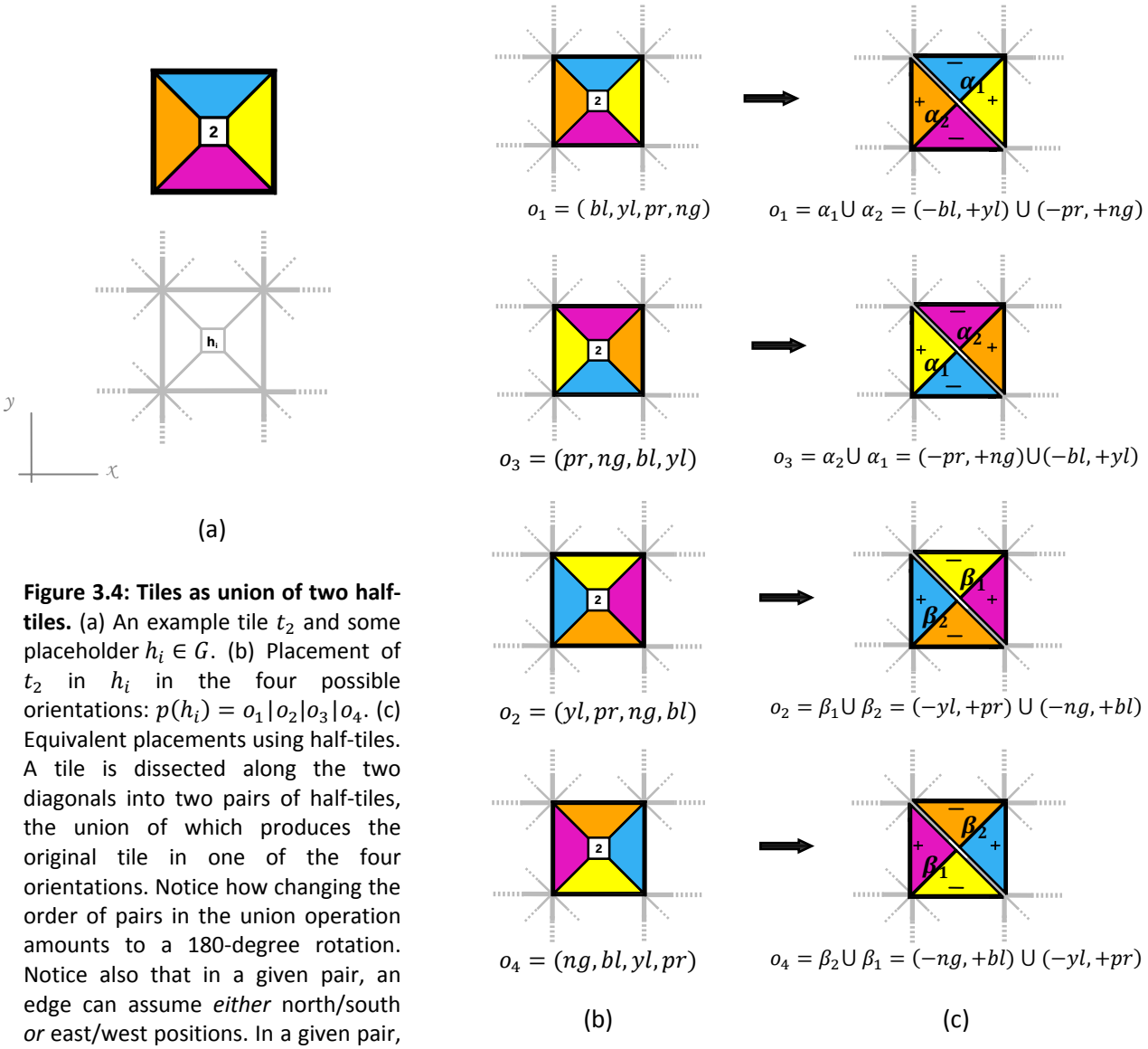


Figure 3.4: Tiles as union of two half-tiles. (a) An example tile t_2 and some placeholder $h_i \in G$. (b) Placement of t_2 in h_i in the four possible orientations: $p(h_i) = o_1|o_2|o_3|o_4$. (c) Equivalent placements using half-tiles. A tile is dissected along the two diagonals into two pairs of half-tiles, the union of which produces the original tile in one of the four orientations. Notice how changing the order of pairs in the union operation amounts to a 180-degree rotation. Notice also that in a given pair, an edge can assume *either* north/south *or* east/west positions. In a given pair, an edge is given a (-) or (+) sign if it can assume (north/south) or (east/west) respectively.

Definition 3.2.1.1

Given $t_i = \{a, b, c, d\} \in Hypatia$, define $\delta_i = \{\alpha_1^i = (-a, +b),\ \alpha_2^i = (-c, +d),\ \beta_1^i = (-b, +c),\ \beta_1^i = (-d, +a)\}$ as the **set of half-tiles** of t_i.

Given $\delta_\tau = \delta_{12} = \{\alpha_1^\tau = (-bl, +gn), \alpha_2^\tau = (-gr, +gr),\ \beta_1^\tau = (-gr, +bl),\ \beta_2^\tau = (-gn, +gr)\}$, define $\Delta = \{\delta_1, \delta_2, .., \delta_{16}\} - \{\beta_1^\tau,\ \beta_2^\tau\}$ as the **superset of all half-tiles** over Hypatia.

Definition 3.2.1.2

Given $t_i = \{a, b, c, d\} \in Hypatia$ and its set of orientations $O_i = \{o_1, o_2, o_3, o_4\}$, define $\bar{O}_i = \{\bar{o}_1 = (-a, +b, -c, +d), \bar{o}_2 = (-b, +c, -d, +a), \bar{o}_3 = (-c, +d, -a, +b), \bar{o}_4 = (-d, +a, -b, +c)\}$ as the **signed set of orientations** of t_i. Define $\bar{\Theta} = \bar{O}_{t_1} \cup \bar{O}_{t_2} \cup .., \cup \bar{O}_{t_{16}}$ as the **superset of all signed orientations** over $Hypatia$.

Definition 3.2.1.3
$(\delta_i, \cup)$ is a set of half-tiles δ_i and an irreflexive, intransitive, symmetric binary relation $\cup$: $\delta_i \times \delta_i \rightarrow \bar{O}_i$ where $\forall\big((x,y),(w,z)\big) \in {\delta_i}^2, (x,y)\cup(w,z) = (x,y,w,z)$ iff $(x,y,w,z) \in \bar{O}_i$.

In any given half-tile, an edge is given a (-) sign if it can assume northern/southern positions (in the corresponding orientation that results from applying union operation) and (+) if it can assume eastern/western positions (Figure 3.4c). The colour-sign compound attribute ensures that, were half-tiles to join *independently* to form a lane, then the union operation on each half-tile in that lane with its complement in another lane necessarily produces a valid signed orientation as defined by Definition 3.2.1.2.

The following example shows the implication of colour-sign compound attributes in half-tiles. Consider tiles Q and R, in Figure 3.5a, which have one colour pr in common and can therefore abut against it, in obviously four possible ways on the tiling grid: east/west, west/east, north/south and south/north edge of Q/R respectively (Figure 3.5b). If tiling was done using half-tiles, and half-tiles of Q and R (Figure 3.5c) are allowed to join based on the colour-sign compound attribute, then their joining (Figure 3.5d) translates into valid signed orientations that correspond to tile orientations in Figure 3.5b –and only those orientations. If colour *only* were used as an attribute for half-tiles, then "twisted" lanes would form, and the union operation on the "misbehaved" half-tile to its complement does not constitute a valid signed orientation.

In Figure 3.5e, the sign constraint is ignored allowing half-tile β_{r2} to join with β_{q1}. Clearly, if the union operation of β_{r2} and its complement is applied, β_{r1} then the resulting orientation is either $\beta_{r2} \cup \beta_{r1} = (-r_2, +r_3, -r_4, +r_1)$ or $\beta_{r1} \cup \beta_{r2} = (-r_4, +r_1, -r_2, +r_3)$, niether of which is in fact the actual (and erroneous) orientation on the grid $(+r_1, -r_2, +r_3, -r_4) \notin \bar{O}_R$. The colour-sign attribute therefore guarantees that the

orientation resulting from the union of a half-tile on one lane to its complement half-tile in another lane always results in a valid orientation, ruling out the possibility of “twisted” lanes.

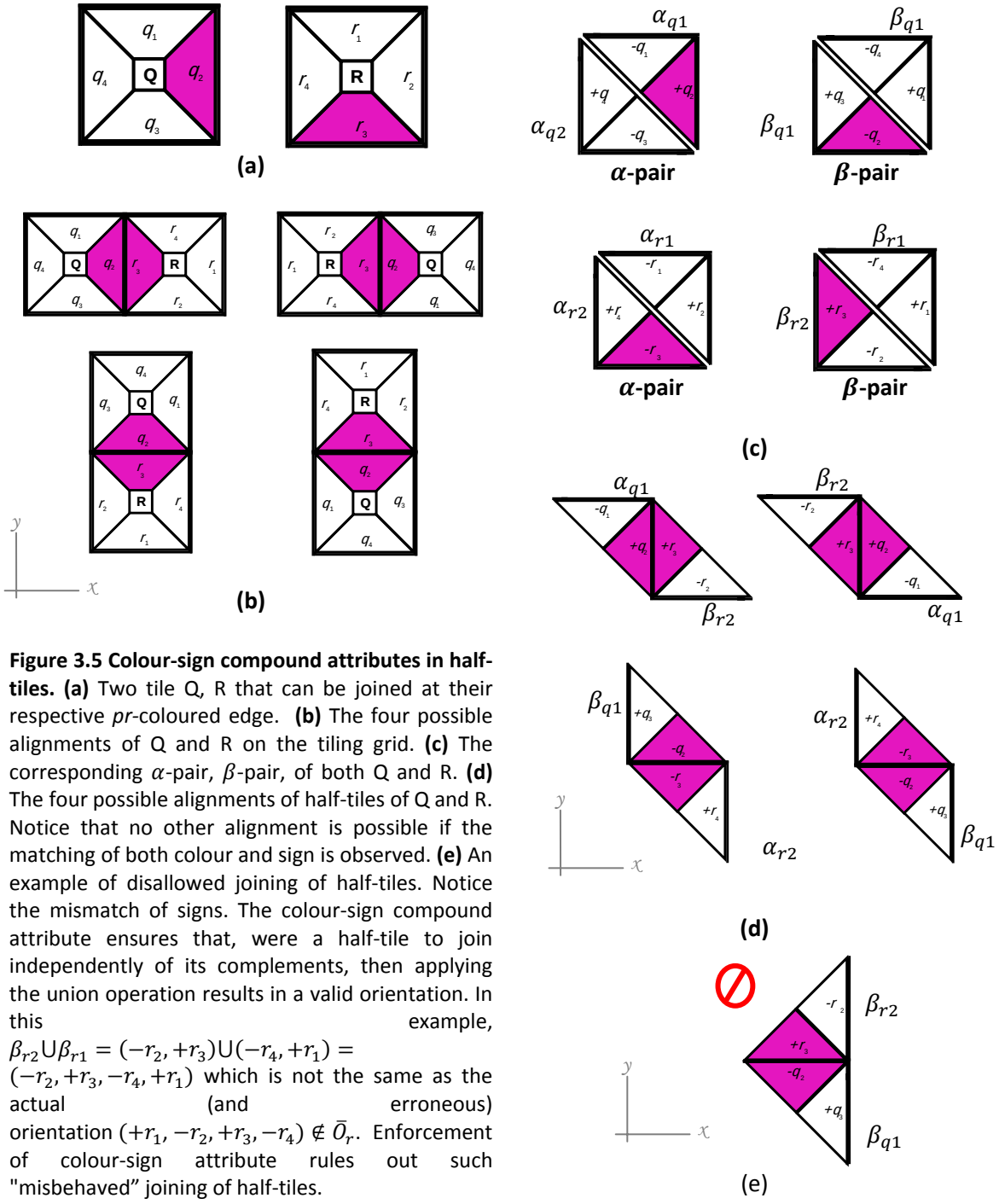


Figure 3.5 Colour-sign compound attributes in half-tiles. (a) Two tile Q, R that can be joined at their respective *pr*-coloured edge. **(b)** The four possible alignments of Q and R on the tiling grid. **(c)** The corresponding α-pair, β-pair, of both Q and R. **(d)** The four possible alignments of half-tiles of Q and R. Notice that no other alignment is possible if the matching of both colour and sign is observed. **(e)** An example of disallowed joining of half-tiles. Notice the mismatch of signs. The colour-sign compound attribute ensures that, were a half-tile to join independently of its complements, then applying the union operation results in a valid orientation. In this example, $\beta_{r2} \cup \beta_{r1} = (-r_2, +r_3) \cup (-r_4, +r_1) = (-r_2, +r_3, -r_4, +r_1)$ which is not the same as the actual (and erroneous) orientation $(+r_1, -r_2, +r_3, -r_4) \notin \bar{O}_r$. Enforcement of colour-sign attribute rules outs such "misbehaved” joining of half-tiles.

3.2.2 Stapling:

We formalize the relation between half-tiles as they join according to colour-sign compound attributes, capturing the notion of what has so far been referred to as "lanes". We refer to the joining of two half-tiles as "stapling".

Definition 3.2.2.1
$(\Delta, \perp)$ is Hyapatia's half-tile superset Δ and an irreflexive, intransitive and symmetric binary relation $\perp: \Delta \times \Delta \to \Delta$ where $\forall\big((x,y),(w,z)\big) \in \Delta^2, \quad (x,y) \perp (w,z) = (y,z)$ ***iff*** $x = w \neq gr$. We say that (x,y) and (w,z) are **vertically stapled.**

Definition 3.2.2.2
$(\Delta, \vdash)$ is Hyapatia's half-tile superset Δ and an irreflexive, intransitive and symmetric binary relation $\vdash: \Delta \times \Delta \to \Delta$ where $\forall\big((x,y),(w,z)\big) \in \Delta^2, \quad (x,y) \vdash (w,z) = (\boldsymbol{w},x)$ ***iff*** $y = z \neq gr$. We say that (y,x) and (w,z) are **horizontally stapled.**

Definition 3.2.2.3
A multi-set $\iota = \{\gamma_1, \gamma_2, .. \gamma_j\} \subset \Delta^j, \gamma = \{\alpha|\beta\},$ is called a **valid lane** over Δ ***iff:***

1) $j = |\iota| \in \{1,3,5,7\} \quad \wedge$
2) $(..((\gamma_1 \vdash \gamma_2) \perp \gamma_3) \ldots . \gamma_j) = (-gr, +gr).$

Define $\boldsymbol{\ell_j}$ as the **set of all valid lanes of length *j*** over Hypatia and $\boldsymbol{\mathcal{L}} = \bigcup_{j=1}^{7} \boldsymbol{\ell_j}$ as the **superset of all valid lanes over** Δ.

As the first two definitions capture the notion of (strictly) north-to-south/east-to-west joining of two half-tiles, the third definition identifies which assemblies of half-tiles constitute valid lanes. A valid lane therefore, according to Definition 3.2.2.3, is one that 1) has the right length and 2) begins with a "$-gr$", ends with a "$+gr$", alternating between valid horizontal and vertical stapling. Notice the convention implied in Definition 3.2.2.3 that lanes are read from the (-) to the (+) end. In the second condition, that valid lanes must algebraically collapse into $(-gr, +gr)$, which obviously necessitates that the extremities γ_1 and γ_j be $(-gr, *)$ and $(*, +gr)$. Clearly neither vertical nor horizontal stapling is defined on the boundary colour "grey"= gr.

For illustration of lane validity, consider the signed grid and the two sample lanes on Hypatia's set of half-tiles, $\varepsilon_3 = \{(-gr, +gn), (-bl, +gn), (-bl, +gr)\}$ and $\varphi_5 = \{(-gr, +gn), (-bl, +gn), (-bl, +yl), (-gn, +yl), (-gn, +gr)\}$ shown in Figure 3.6. Clearly the first condition is met since $|\varepsilon_2| = 3, |\varphi_3| = 5$. To check the second condition, according to Definition 3.2.2.3 we have:

$$\begin{gathered}\big((-gr, +\boldsymbol{gn}) \vdash (-bl, +\boldsymbol{gn})\big) \perp (-bl, +gr) = \\ (-\boldsymbol{bl}, -gr) \perp (-\boldsymbol{bl}, +gr) = \\ (-\boldsymbol{gr}, +\boldsymbol{gr})\end{gathered}$$

Similarly for L_6:

$$\begin{gathered}\Big(\big((-gr, +\boldsymbol{gn}) \vdash (-bl, +\boldsymbol{gn})\big) \perp (-bl, +yl)\Big) \vdash (-gn, +yl)\Big) \perp (-gn, +gr) = \\ \Big(\big((-\boldsymbol{bl}, -gr) \perp (-\boldsymbol{bl}, +yl)\big) \vdash (-gn, +yl)\Big) \perp (-gn, +gr) = \\ \big((-gr, +\boldsymbol{yl}) \vdash (-gn, +\boldsymbol{yl})\big) \perp (-gn, +gr) = \\ (-\boldsymbol{gn}, -gr) \perp (-\boldsymbol{gn}, +gr) = \\ (-\boldsymbol{gr}, +\boldsymbol{gr})\end{gathered}$$

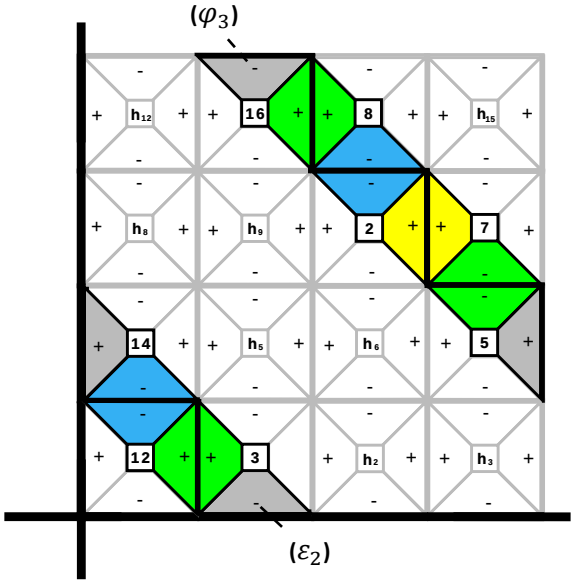


Figure 3.6 Examples of valid lanes. Two examples of valid lanes over the set of half-tiles of Hypatia, $|\varepsilon_2| = 3, |\varphi_5| = 5$ half-tiles. Both lanes conform to the length (both of odd length $3,5 \in \{1,3,5,7\}$ and extremety conditions (beginning and ending with with a negative and positive boundary colour, respectively).

3.2.3 Bridging:

Having defined the notion of a half-tile and the relation between half-tiles (stapling), we proceed to establishing the relation between *lanes of half-tiles*, which we refer to as "bridging". This is the last piece needed to formally describe the solution to Hypatia using the diagonal-wise half-tile stacking algorithm which was outlined informally at the beginning of 3.2.

Definition 3.2.3.1:
$(\boldsymbol{\mathcal{L}},\div)$ is Hypatia's superset of all valid lanes and an irreflexive, intransitive and symmetric binary relation $\div : \boldsymbol{\ell_i} \times \boldsymbol{\ell_j} \to \bar{\Theta}^r,\ r = ((i < j)?: \lfloor i/2 \rfloor, \lfloor j/2 \rfloor),$ where $\forall \iota_i \in \boldsymbol{\ell_i}, \iota_j \in \boldsymbol{\ell_j},\ (\iota_i \div \iota_j)\ \boldsymbol{iff}$:

- $(|j - i| = 2) \vee (i = j = 7) \quad \wedge$
  - $(i \leq j) \Rightarrow \forall k, (1 \leq k \leq i) \wedge (k\ mod\ 2 = 1) \Rightarrow (\acute{\gamma}_{k+1} \cup \gamma_k) \in \bar{\Theta} \Leftrightarrow \boldsymbol{\iota_i} \div \boldsymbol{\iota_j} = \{(\boldsymbol{\acute{\gamma}_2} \cup \boldsymbol{\gamma_1}), \dots, (\boldsymbol{\acute{\gamma}_{k+1}} \cup \boldsymbol{\gamma_k})\}$
  - $(i > j) \Rightarrow \forall k, (1 \leq k \leq j) \wedge (k\ mod\ 2 = 1) \Rightarrow (\gamma_{k+1} \cup \bar{\gamma}_k) \in \bar{\Theta} \Leftrightarrow \boldsymbol{\iota_i} \div \boldsymbol{\iota_j} = \{(\boldsymbol{\gamma_1} \cup \boldsymbol{\acute{\gamma}_2}), \dots, (\boldsymbol{\gamma_{k+1}} \cup \boldsymbol{\acute{\gamma}_k})\}$

where γ_k is the k^{th} half-tile of ι_i, and $\gamma, \acute{\gamma} = \{\alpha | \beta\}$. We say that ι_i **bridges to** ι_j**.**

Definition 3.2.3.1 captures the "stacking" notion of the half-tile stacking algorithm that was informally described at the beginning of this section. The stacking of two lanes is therefore the equivalent of placing a diagonal set of tiles (see 3.1). The solution to Hypatia, which has previously been described as a mapping of placeholders to tiles/orientations (Definition 3.2.1.16), can equivalently be described now as a mapping between placeholders and lane-to-lane bridging.

3.3 Solution of Hypatia Using Diagonal-wise Half-tile Stacking

Definition 3.2.3.2:
Given $\mathcal{B} = \{\varepsilon_1, \varphi_1, \varepsilon_3, \varphi_3, \varepsilon_5, \varphi_5, \varepsilon_7, \varphi_7\}, \varepsilon_i, \varphi_i \in \Delta^i$ and $|\varepsilon_i| = |\varphi_i| = i$, where:
$[(\varepsilon_1 \div \varepsilon_3) = (\alpha_1^{\tau} \cup \alpha_2^{\tau})] \wedge (\varphi_1 \div \varphi_3) \wedge (\varepsilon_3 \div \varepsilon_5) \wedge (\varphi_3 \div \varphi_5) \wedge (\varepsilon_5 \div \varepsilon_7) \wedge (\varphi_5 \div \varphi_7) \wedge (\varepsilon_7 \div \varphi_7)$

Define the injective function $\overline{\boldsymbol{p}}: H \to \bar{\Theta}$:

$$h_0 \longmapsto (\varepsilon_1 \div \varepsilon_3)$$
$$h_{15} \longmapsto (\varphi_1 \div \varphi_3)$$
$$h_1, h_4 \longmapsto (\varepsilon_3 \div \varepsilon_5)$$
$$h_{11}, h_{14} \longmapsto (\varphi_3 \div \varphi_5)$$
$$h_2, h_5, h_8 \longmapsto (\varepsilon_5 \div \varepsilon_7)$$
$$h_7, h_{10} h_{13} \longmapsto (\varphi_5 \div \varphi_7)$$
$$h_3 h_6, h_4, h_{12} \longmapsto (\varepsilon_7 \div \varphi_7)$$

Theorem 3.1:
If $p(H)$ exists, then so does $\overline{p}(H)$.

Proof:
If $p(H)$ exists, then there is a a set of diagonal tiles:
$\mathrm{E}_i = \{t_{x_1/j}, \dots, t_{x_i/k}\}$, $\mathrm{E}_{i+1} = \{t_{y_1/u}, \dots, t_{y_{i+1}/v}\}$; $1 \le |\mathrm{E}_i| = i \le 3; 1 \le j, k, u, v \le 4; t_{x_i}, t_{y_i} \in Hypatia.$
Let $p(h_r) = t_{x_i/k} \in \mathrm{E_i}$. Let $t_{x_1/j} = (a_1^1,\ a_2^1,\ a_3^1,\ a_4^1) \dots t_{x_i/j} = \left(a_1^i,\ a_2^i,\ a_3^i,\ a_4^i\right)$. Let $t_{y_1/j} = (b_1^1,\ b_2^1,\ b_3^1,\ b_4^1) \dots t_{y_{i+1}/j} = \left(b_1^{i+1},\ b_2^{i+1},\ b_3^{i+1},\ b_4^{i+1}\right)$

By Definition 3.2.1.16:
$b_3^1 = gr, b_4^1 = a_2^1,\ a_1^1 = b_3^2, \dots\dots\ b_3^{i+1} = a_1^i,\ b_4^{i+1} = gr$
By Definition 3.2.1.1:
$\exists(-a_3^1, +a_4^1) \in \delta_{x_1}, \exists\left(-a_3^i, +a_4^i\right) \in \delta_{x_i}$
And
$\exists(-b_3^1 = gr, +b_4^1) \in \delta_{y_1}, \exists(-a_1^1, +a_2^1) \in \delta_{x_1},\ \exists(-b_3^2, +b_4^2) \in \delta_{y_2} \dots \exists\left(-a_1^i, +a_2^i\right) \in \delta_{x_i}, \exists\left(-b_3^{i+1}, +b_4^{i+1} = +gr\right) \in \delta_{y_{i+1}}$

Let $\iota_q = \left\{(-gr, +b_4^1), (-a_1^1, +a_2^1), (-b_3^2, +b_4^2) \dots \left(-a_1^i, +a_2^i\right), \left(-b_3^{i+1}, +gr\right)\right\}$
Clearly, $q = i + (i+1) \in \{3,5,7\}$ and $(-gr, +b_4^1), \vdash (-a_1^1, +a_2^1) \perp (-b_3^2, +b_4^2) \dots \left(-a_1^i, +a_2^i\right) \perp \left(-b_3^{i+1}, +gr\right) = (-gr, +gr)$

$\therefore \iota_q$ is a valid lane by Definition 3.2.2.3

For $i = 1 \Rightarrow |\iota_q| = 3 \Rightarrow \iota_s = (-a_3^1, +a_4^1) \in \delta_{x_1}$ by Definition 3.2.1.16.
Since $p(H)$ exists, then by Definition 3.2.1.16, $a_3^1 = a_4^1 = gr$, and $|s| = 1 \in \{1,3,5,7\} \Rightarrow \iota_s$ is a valid lane by Definition 3.2.2.3, and $(\iota_s, \iota_q) \in \div \Rightarrow \iota_s \div \iota_q = \{(-a_1^1, +a_2^1) \cup (-a_3^1, +a_4^1)\} = \{(-a_1^1, +a_2^1, -a_3^1, +a_4^1)\} \subset \bar{O}_{x_1}$

$$\therefore p(h_0) = t_{x_1/j} \in O_{x_1} \Leftrightarrow \overline{p}(h_0) = \iota_q \div \iota_s = \bar{t}_{x_1/j} \in \bar{O}_{x_1}$$

By trivial extension, the same argument follows for $i = 3, 5, 7$. By virtue of symmetry, the same procedure can be repeated for the diagonal set of tiles Φ_i, $1 \le i \le 3$.

$\therefore$ for every diagonal set $X \in Range(p)$, there exists two valid lanes the bridging of which results in the corresponding signed diagonal set $\overline{X} \in Range(\bar{p})$. ■

Theorem 3.2:
If $p(H)$ is unique, then so is $\bar{p}(H)$.

Proof:
Let $p(h_r) = t_{x_i/k} = (a, b, c, d)$.
By Theorem 3.1, $\exists \bar{o}_j^x = (-a, +b, -c, +d)$: $\bar{p}(h_r) = \bar{o}_j^x$.

Assume $\exists \bar{p}_2 : H \to \bar{\Theta}$, so $\exists \bar{o}_k^y \neq \bar{o}_j^x : \bar{p}_2(h_r) = \bar{o}_k^y$ (1)
By Definition 3.2.1.16, $\bar{o}_k^y = (\pm a, \pm b, \pm c, \pm d)$.
Since $\{(+a, \pm b, \pm c, \pm d), (\pm a, -b, \pm c, \pm d), (\pm a, \pm b, +c, \pm d), (\pm a, \pm b, \pm c, -d)\} \notin \bar{\Theta}$
$\therefore \bar{o}_k^y = (-a, +b, -c, +d) = \bar{o}_j^x \Rightarrow$ Contradiction with (1). ■

3.4 Motivation:

The diagonal-wise half-tile stacking algorithm is designed with DNA implementation in mind. The mapping of the three major stages of the algorithm (representation, stapling and bridging) onto a DNA-Computing protocol (see Chapter 4) allows for the use of a supremely powerful heuristics in DNA-Computing: polymerase chain reaction (PCR).

Given a mixture containing thousands of billions of DNA strands, PCR can selectively amplify (literally, produce exponentially many copies of) a certain strand that we consider *correct* or at least *very likely to be correct*. A strand can be subjected to exponential amplification of PCR if a partial sequence of bases at the two ends of the strand are known (usually 10 or more bases should suffice). In the developed half-tile stacking algorithm, and given the DNA sequence encoding a lane, we have straight-forward criteria for PCR-ing lanes: amplify those beginning and ending with a sequence that encodes a boundary edge (grey).

As will be laid out in details in the next chapter, each half-tile will be encoded by a single-stranded DNA sequence, and half-tiles will be stapled together by other single-stranded *complementary* sequences that bring half-tiles together by the rule of colour-sign compound attribute. As the process is of course fundamentally non-deterministic, all sorts of erroneous

lanes could result (e.g. a lane of an even length of half-tiles or a lane not beginning or ending with a grey edge). Selection goes against those lanes, by amplifying only the sequences that are of the right *length* (which we can harvest using gel electrophoresis) and the right extremities (grey).

Since the ultimate goal is to find the set of lanes that will make a perfect bridging, i.e. the solution to the full puzzle, we now have a better chance of getting that solution since each correct lane was *favoured* and now exists in abundance. Furthermore, the half-tile stacking algorithm allows for PCR to be inserted at any step. For example, amplified lanes of length 3 that succeed in bridging to lanes of length 5 can be subjected to PCR once again, so as to increase their numbers against those who failed to bridge. Successive PCR insertion throughout the DNA-based implementation of the algorithm is indeed the molecular equivalent of a very powerful heuristics in optimization algorithms.

Chapter 4 DNA Implementation and Results

In light of the discussion of the previous chapter, the DNA-based implementation of the diagonal-wise half-tile stacking algorithm can generally be described as a successive execution of three processes: (1) Encoding $\boldsymbol{\Delta}$, (2) Enumerating $\boldsymbol{\mathcal{L}}$, and (3) Finding $\boldsymbol{\mathcal{B}}$. Each half-tile is encoded as a ssDNA sequence (representation); such sequences are joined together —observing the constraint of colour-sign matching— by *complementary* single-stranded DNA sequences (stapling); and finally the resulting lanes are stacked in a step-wise fashion (bridging) using, again, other complementary sequences of ssDNA. There is of course a considerable amount of wet lab details that goes into each of these three main processes, as will be presented in this chapter's demonstrations. The stapling and bridging protocols are demonstrated using a sample set of DNA oligonucleotides and the results are shown.

4.1 Representation:

We present here the DNA encoding of the Hypatia puzzle with 1) strands for half-tiles, 2) stapling strands that join half-tiles based on matching of colour-sign attribute (effectively implementing the horizontal ($\vdash$) and vertical ($\perp$) relations), and 3) bridging strands that stack-up lanes such that a half-tile residing on one lane is bridged to its α or β pair complement on another lane (effectively implementing the bridging ($\div$) relation). By the end of this subsection, we provide a high-level summary relating the DNA encoding to the problem definition in Chapter 2 and the solution approach of Chapter 3, and conclude with a summary of the total number of DNA strands involved throughout the DNA-based computation of the solution to Hypatia.

4.1.1 Encoding of DNA half-tiles:

We begin by showing the DNA encoding of the seed tile. Recall from 2.3 that $\tau = t_{12}$ is the designated seed tile, and as such must appear in the orientation $\alpha_1^\tau \cup \alpha_2^\tau = (-bl, +gn) \cup (-gr, +gr) = (-bl, +gn, -gr, +gr)$. The β-pair is therefore not encoded (Figure 4.1a, see also Definition 3.2.1.1). The two half-tiles in the α-pair are each encoded with a random 26-bp (bp) ssDNA sequence (Figure 4.1b; see 4.1.3 for the rationale of choosing 26-bp specifically). The encoding of the seed tile readily determines the directionality (5'-3' or 3'-5') of lanes on the DNA grid as shown in Figure 4.1c (bridges are not shown on the grid, see 4.1.3). Notice that lanes alternate between 5'-3' and 3'-5' orientations — a fundamental requirement of any hybridizing DNA strands.

For the rest of tiles in Hypatia, each half-tile of each α/β pair is encoded in such a way that every tile can assume every possible *position* and every possible *orientation* on the DNA grid. Given a 5'-3' DNA sequence $s = 5'\text{›}N_1, N_2 \ldots N_k\text{›}3'$, where $N_k \in \{A, C, G, T\}$, we refer to $R(s) = 5'\text{›}N_k, \ldots N_2, N_1\text{›}3'$ as the *reverse* sequence of s. A half-tile $(-c_1, +c_2)$ is encoded with a 26-bp 5'›[-c_1][CORE][+c_1]›3' sequence: 8-bp sub-sequence encoding c_1, 8-bp sub-sequence encoding $+c_2$, and 10-bp "core" sub-sequence that is unique to each half-tile. The reverse of such sequence, R(5'›[-c_1][CORE][+c_1]›3') = 5'›R [+c_1]R[CORE]R[-c_1]›3' is also encoded, resulting in two ssDNA per half-tile, and so each colour-sign attribute of a half-tile appears at the 5' end in one sequence, and at the 3' in the other.

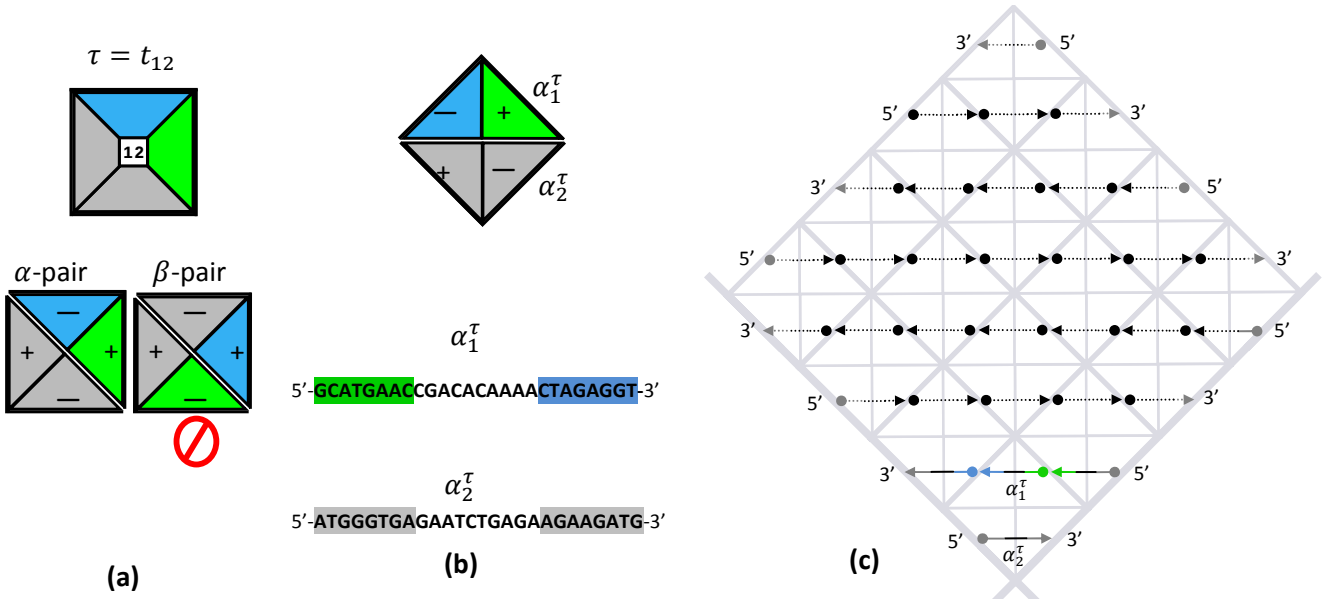
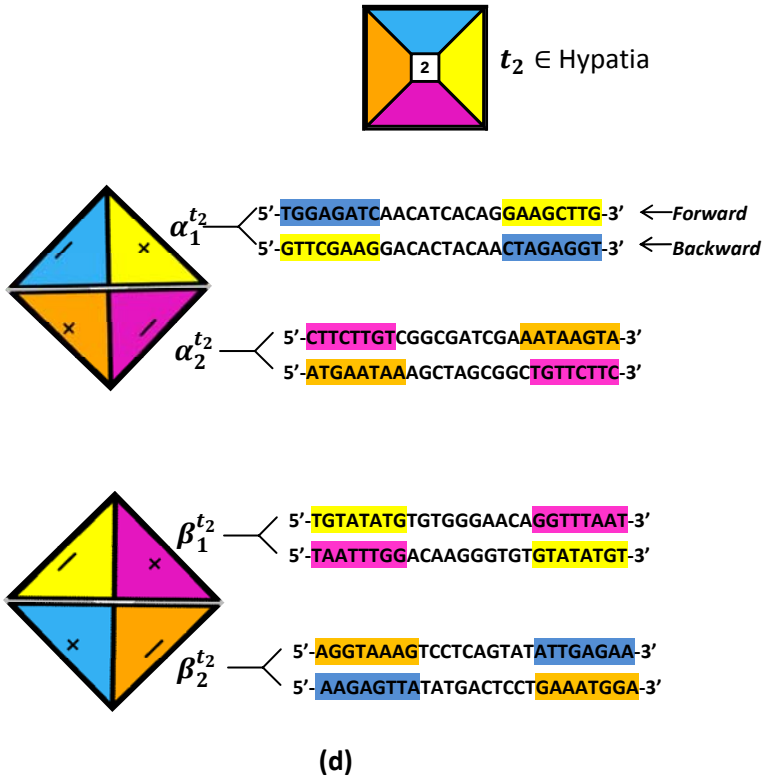


Figure 4.1 Encoding of half- tiles. a) The seed tile and the corresponding pairs of half-tiles. As the position and orientation of the seed tile is pre-determined, the β-pair is not encoded. **b)** The encoding of the seed tile: each half-tile of the α-pair is encoded with a random 26-bp ssDNA. **c)** The layout of the DNA grid: the encoding of the seed tile predetermines the directionality (5′-3′ or 3′-5′) of each lane in the final solution. Successive lanes of DNA must have reverse directionality as bridging takes place (see 4.1.3). A 5′ end is denoted with a filled circle while an arrow denotes a 3′ end. **d)** An example tile t_2 from Hypatia puzzle set and the corresponding encoding of its α and β pairs. 26-bp ssDNA sequences, each encoding a half-tile once in 5′-3′ and another in 3′-5′ directionality (the two being the reverse of each other). Each colour-sign compound attribute is associated with an 8-bp sub-sequence (see 4.1.2 on stapling), appearing at 5′ end in one sequence, and at 3′ end in the other (the reverse). Notice that the 10-bp core area is unique across half-tiles and that a (+)-colour sequence is distinct from (-)-colour's. Being encoded with forward and backward sequences, a half-tile can in effect assume a position on both 5′-3′ and 3′-5′ lanes.



For each half-tile, we will refer to 5'›[-c_1][CORE][+c_1]›3' as the ***forward*** sequence and to R(5'›[-c_1][CORE][+c_1]›3') = = 5'›R [+c_1]R[CORE]R[-c_1]›3' as the ***backward*** sequence*. As such, every half-tile can assume a position in both 5'-3' and 3'-5' lanes on the grid. There is therefore a total of 8 26-bp ssDNA per each tile in Hypatia (except the seed tile where there are only 2 strands). Note that 8-bp subsequence encoding a colour-sign attribute is the same across all half-tiles, except for $\mp grey$ which is made distinct in corner tiles from boundary tiles (those with one grey edge). This distinction provides a further useful heuristic for DNA stapling protocol (see Step 2 in 4.2), as valid lanes of length 1 and 7 half-tiles are distinguishable from those of length 3 and 5 half-tiles by their beginning and ending sequences.

4.1.2 Encoding of DNA staples:

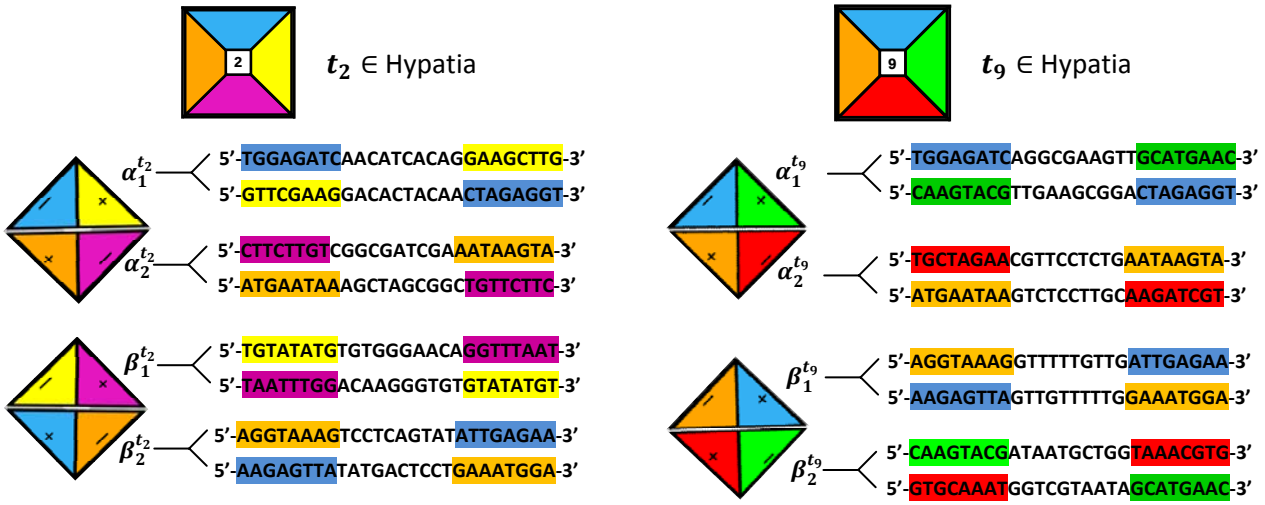
It is very important to point out that, given any 26-bp half-tile sequence, the 8-bp sub-sequence encoding a colour-sign attribute is the same across all half-tiles in $\boldsymbol{\Delta}$ (except $\pm grey$ which are encoded differently in corner and boundary tiles). As a result, a stapler ssDNA is a 16-bp sequence whose 8-bp subsequence at the 5' end is complementary to the 8-bp subsequence encoding a colour-sign attribute at the 3' end of a half-tile's 26-bp sequence, while its 8-bp subsequence at the 3' end is complementary to the 8-bp subsequence encoding the *same* colour-sign attribute at the 5' end of the 26-bp sequence encoding another half-tile.

Consider t_2 and t_9 from Hypatia's puzzle set. Figure 4.2a shows the encoding of their respective pairs of half-tiles, following the presentation of 4.1.1. The two tiles have blue (bl) and orange (ng) edges in common. There are two signs +/- and two directionalities (forward/backward), hence the two tiles can be stapled in four different manners for each common colour (Figure 4.2b for bl stapler and Figure 4.2c for ng stapler). Notice, however, that

* Corner tiles (other than the seed tile) can in fact be encoded using 6 sequences in total, since the directionality of the pair (-gr, +gr)U(*,*), which is the tile in placeholder h_{15}, is already predetermined and so the forward sequences of this pair need not be encoded (see the DNA grid above).

there is one stapler strand per colour-sign attribute. For example, the vertical stapling of half-tile $\alpha_1^{t_2}$ to half-tile $\alpha_1^{t_9}$ on colour-sign attribute $(-bl)$ is carried out by the same strand regardless of which strand carries $(-bl)$ at the 5'- and which at the 3'-end. This is clearly a direct result of our choice of making the 8-bp subsequence at a 3'-end the reverse of that at the 5'-end. Notice that stapling takes place always between a forward and a backward sequence.

Since there is a total of 7 colours × 2 signs (+/-) in Hypatia's superset of half-tiles Δ (the eighth colour being grey, but it is not involved in stapling of course), there are therefore 14 stapler strands in total. Identical 8-bp subsequence of each colour-sign attribute across all half-tiles in Δ has a very clear implication: a half-tile could be stapled to itself. Why, then, not make each colour-sign attribute unique from one half-tile to another, so as to avoid such erroneous stapling (and in fact comply with the irreflexivity property of the vertical/horizontal stapling as per Definition 3.2.2.1 and Definition 3.2.2.2)? Recall from 3.1 that duplication cannot result in full tiling of the full 4×4 tiling grid, and as such the devised diagonal-wise stacking algorithm



(a)

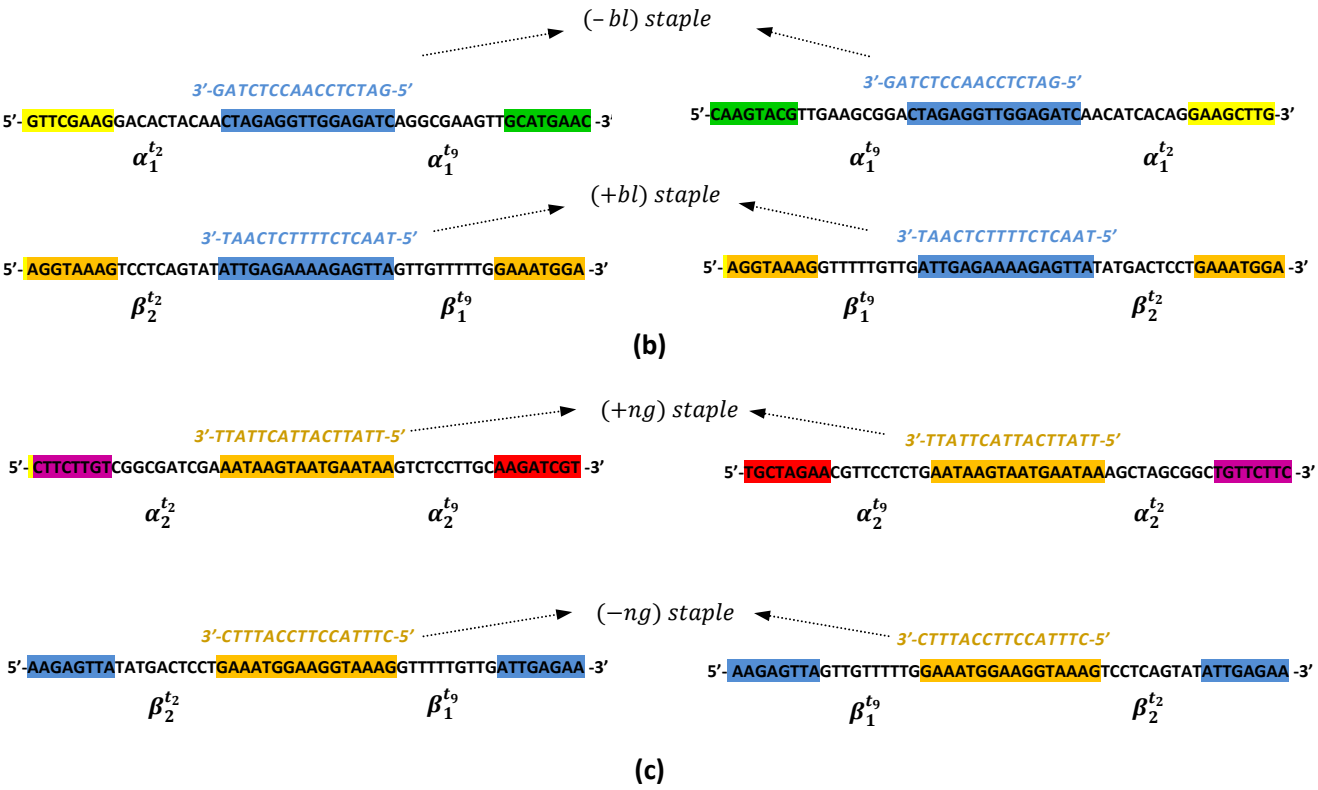


Figure 4.2: Encoding of Staples. a) t_2 and t_9 ∈ Hypatia, their corressponding pairs of half-tiles and ssDNA encoding of each half-tile with a 26-bp sequence as in 4.1.1. Notice that an 8-bp subsequence encoding a colour-sign attribute is the same across all half-tiles (in this case, $\pm bl$ and $\pm ng$ have the same 8-bp subsequence in the encoding of t_2 and t_9's half-tiles). **b)** Stapling of $\alpha_1^{t_2}$ - $\alpha_1^{t_{11}}$ on $(-bl)$ and $\beta_2^{t_2}$ - $\beta_1^{t_9}$ on the $(+bl)$ colour-signed attribute: 2 signs X 2 directionalities = 4 ways of stapling. Notice, however, that the same stapler strand is used regardless of which half-tile holds the 8-bp subsequence at the 5′ and which at the 3′ (a direct result of our choosing of the same 8-bp encoding of a colour-sign attribute across all half-tiles; sequences are always read from 5′ to 3′ by convention). **c)** Stapling of $\alpha_2^{t_2}$ - $\alpha_2^{t_9}$ on $(+ng)$ and $\beta_2^{t_2}$ - $\beta_1^{t_9}$ on the $(-ng)$ colour-signed attribute. As in (b), there is one 16-bp stapler strand per colour-sign attribute.

needs not rule out duplication. Secondly, enforcing uniqueness results in a much larger set of stapler strands, namely 368 stapler strands in total, which can be obtained from:

$$\sum c_x(c_x - 1)$$

where $x \in \{rd, gn, yl, bl, ng, pr, br\}$ and c_x is the total number of x-coloured edges in Hypatia. We have therefore (ab)used a property of the puzzle set to minimize the number of stapler strands (the synthesis of which costs ~$0.08-0.15 per nucleotide at the time of this writing).

4.1.3 Encoding of DNA bridges:

For the time being, let us assume that 26-bp half-tile strands have been joined together by stapler strands to form lanes of ssDNA, and that they (stapler strands) have disappeared thereafter (in 4.2, we will see the details of stapling and how stapler strands are removed after accomplishing their purpose). The bridging of ssDNA lanes is accomplished using, once again, ssDNA strands that uniquely bring pairs of half-tiles together via hybridization. Figure 4.3 shows the DNA grid decorated with a schematic representation of DNA bridges (in red), where consecutive 26-bp subsequences of each lane alternate between being bridged *up* and bridged *down*.

At each bridging area between two ssDNA lane sequences, a pair of ssDNA bridges, each of length 26-bp, each hybridizes to 13-bp sequence on one lane and to another 13-bp sequence in the other lane (starting at a 5' (3') on one lane and ending at 3' (5') of the other lane). A 13-bp subsequence of a 26-bp ssDNA bridge complements, in one lane, the 8-bp encoding a colour-sign attribute plus 5-bp off of the CORE subsequence; and does the same on the second lane with the remaining 13-bp subsequence. The meeting point of the two ssDNA bridges is called the *crossover* point, as each bridge hybridizes to one lane then *crosses over* to the other lane at that point. This can be seen in Figure 4.3 with the caveat that, in reality, there are no gaps between lanes and/or pairs of bridges at the crossover point (gaps are included in the figure to in order to portray the correspondence between the DNA grid and tiling grid). Therefore, the full DNA grid exists in double-stranded DNA (dsDNA) form.

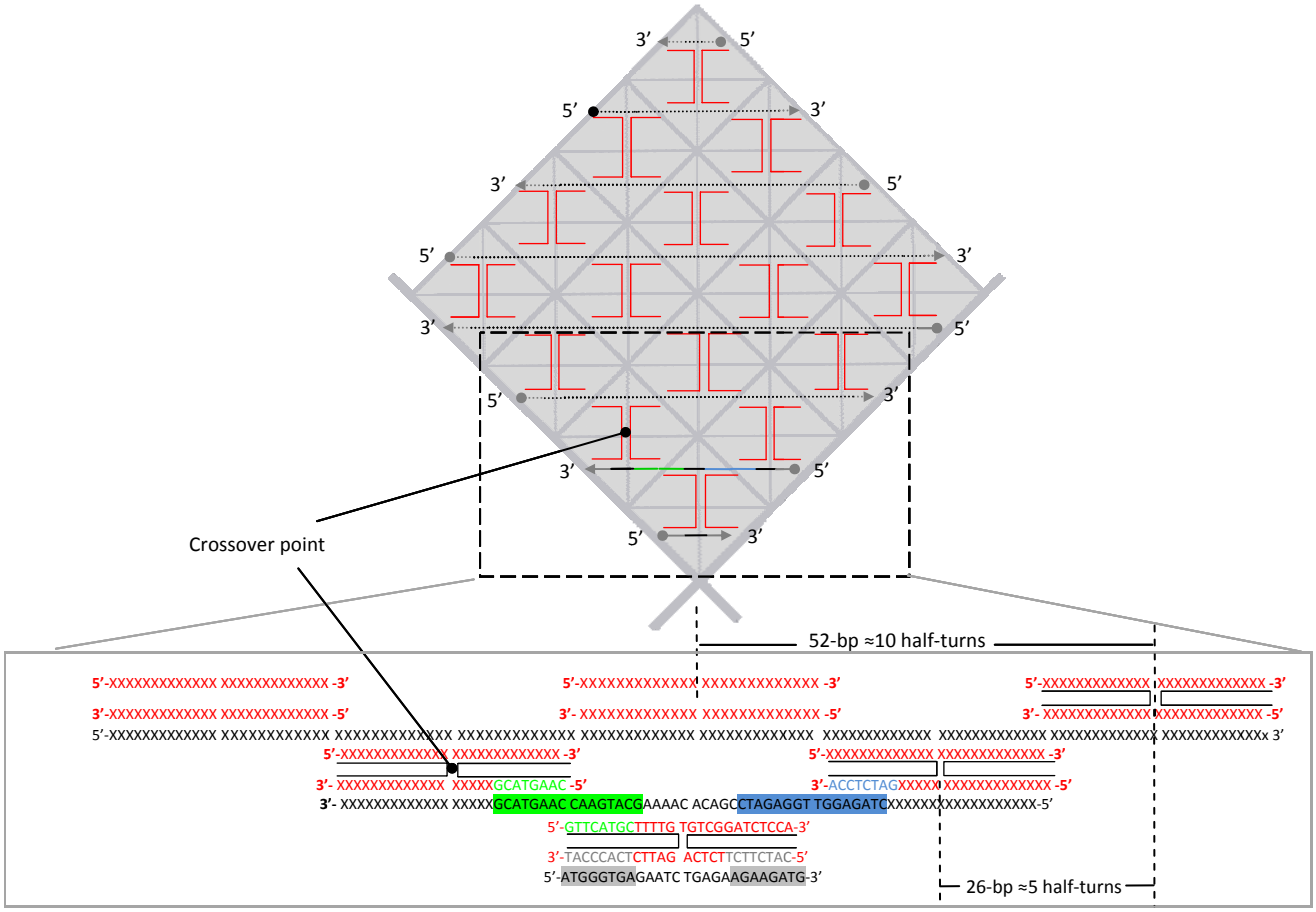


Figure 4.3 Anatomy of DNA bridging. A pair of 26-bp ssDNA strands is used to bridge half-tiles from one lane to another. Each ssDNA bridge hybridizes over 13-bp sequence on one lane (8-bp sequence encoding a colour-sign attribute + 5-bp sequence off of the core sequence) then *crosses over* and hybridizes, on the opposite direction, to 13-bp sequence on the other lane. The meeting point of a pair of bridges is called *crossover point,* as both bridges cross over from one lane to another at that point. Both the DNA grid and the inset should be viewed with a caveat that, in reality, there are no gaps between lanes: the 13th base on one ssDNA bridge hybridizes on one lane and the immediate 14th base hybridizes to the next lane. This U-turn behaviour of a ssDNA bridge is indicated in the inset with a line from the 5′ to the 3′ end of each bridge. The distance between two consecutive crossover points amounts to an odd number of half-turns (a strand on the double helix makes full turn in 3D every 10.67 bases), such that the up/down crossovers of bridges result in minimal twist strain on the double-helix formation. Hence the rationale behind designating 26-bp for encoding half-tiles in 4.1.1.

Using double-crossover (DX) DNA strands to bring together two ssDNA strands was first demonstrated by Seeman [49] and has been the underlying principal in structural DNA nanotechnology (see [43][63][42] for examples and variants of DX arrangements). A strand in the DNA double-helix makes a *full turn* in 3-Dimensional space every 10.67 bases [42], and so the distance in base pairs between two consecutive crossover points must be an **odd** number of

half turns, in order to suit the *up/down* alternation of bridging while creating minimal twist strain on the double-helix. In our design, the distance between two crossover points is 5 half-turns:

$$(5\ half-turns) \times \left(5.3\ \frac{bp}{half-turns}\right) \approx 26\ bp$$

It is clear therefore why, in 4.1.1, we have chosen to encode half-tiles using specifically 26-bp sequences. Notice that the distance between every other crossover points is an **even** number of half-turns, which in our design is ~10 half-turns, given by:

$$(10\ half-turns) \times \left(5.3\ \frac{bp}{half-turns}\right) \approx 53\ bp$$

There is 1 bp difference between the ideal distance (~52 bp) and the distance we have in our design, which clearly results from having a non-integer length of a full turn in the DNA double helix. The design can of course be optimized further by, for example, adding extra bases to sequences encoding boundary half-tiles (grey), which would reduce the strain on the DNA grid (resulting from "twisting" of the double-helix in order to make up for the missing base [42]). However, since the longest series of DX points in our problem is 7, which occurs at the main diagonal of the DNA grid at lanes ε_4 and φ_4 (see 3.2), we can assume that the resulting twist strain has negligible effect.

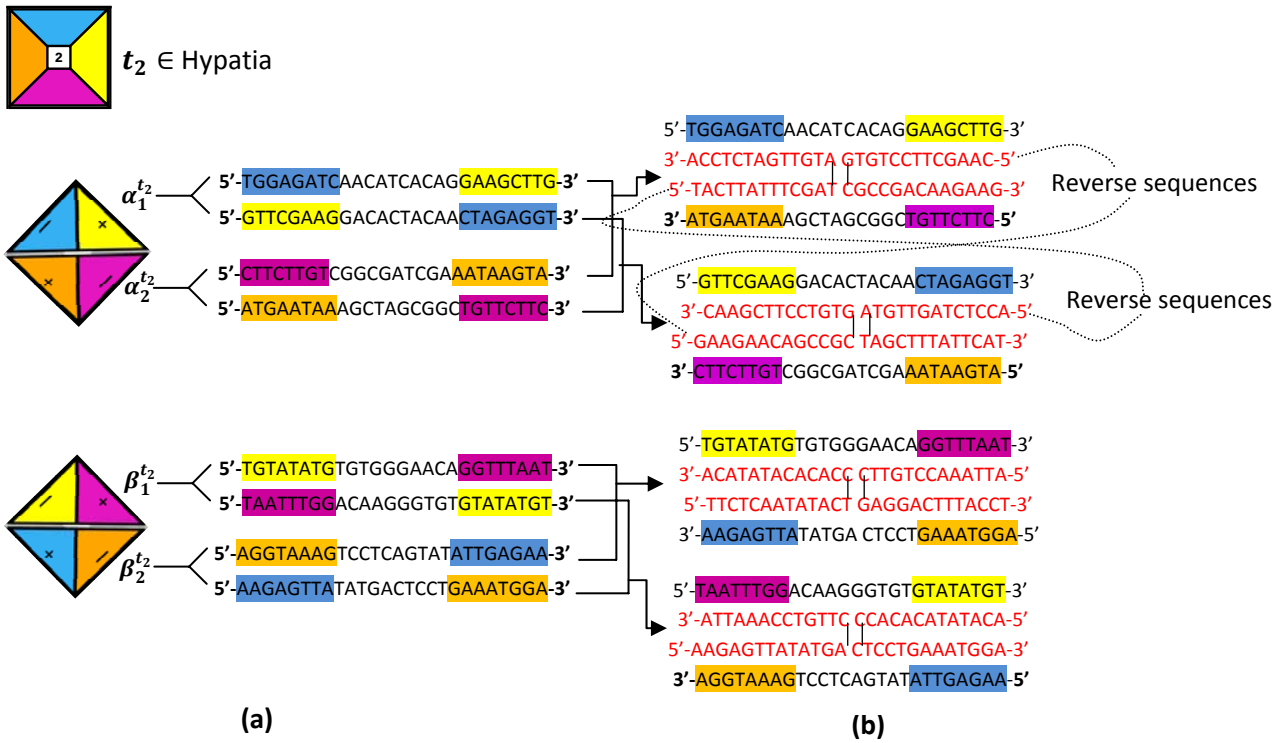


Figure 4.4: Encoding of Bridges. a) An example tile t_2 ∈ Hypatia, its pairs of half-tiles and their corresponding encoding following 4.1.1. **b)** For each α/β pair of half-tiles, the forward (backward) sequence of a half-tile is bridged to the forward (backward) sequence of the other tile in the same α/β pair. Notice that the pair of 26-bp ssDNA bridges in the forward-to-forward bridging are the reverse of those in the backward-to-backward bridging (naturally, since forward and backward sequences of a half-tile are the reverse of each other). There are therefore a total of 8 26-bp ssDNA bridging sequences per tile in Hypatia (except for the seed tile where there is only 2 26-ssDNA bridging sequences).

Over the sequences encoding each α/β pair of half-tiles as per 4.1.1, the bridging takes place from a forward (backward) sequence in one half-tile to the forward (backward) sequence in the other half-tile (within the same α/β pair), using two 26-bp ssDNA bridges in each instance. Figure 4.4 illustrates the set of ssDNA bridges for an example tile t_2 ∈ Hypatia. Since forward and backward sequences encoding a half-tile are the reverse of each other, the pair of ssDNA bridges of the forward sequences and those of the backward sequences are also such. There are therefore 8 26-bp ssDNA bridges per tile (except, the seed tile, where there is only one pair of ssDNA bridges).

4.1.4 Summary

ssDNA sequences presented in the previous three subsections (half-tiles, staples, and bridges) are manipulated –using standard molecular-biology protocols– to carry out a DNA-based computation of the solution to Hypatia. The DNA-based computation (explained in 4.2 and 4.3) follows the half-tile stacking algorithm presented in Chapter 3, itself formulated based on the foundational formalizations of Chapter 2. Therefore, in the interest of high-level clarity of our approach from problem definition to abstract solution outline to DNA implementation, we highlight here the big picture of the whole approach. Figure 4.5a shows an example tile $t_2 \in$ Hypatia with some placeholder on the tiling grid. t_2 can be assigned to a placeholder on the tiling grid in one of its four orientations, as shown in Figure 4.5b. Our solution approach has been to represent a tile as a union of two half-tiles resulting from the dissection along the two diagonals (see 3.2). The two resulting pairs of half-tiles can re-produce the mother tile in one of its four orientations (Figure 4.5c). Finally, the DNA encoding of half-tiles allows t_2 to assume all positions and orientations on the DNA grid. Notice that, as a consequence of DNA having directionality[1], a half-tile can assume both a 5'-3' or a 3'-5' position on the DNA grid, hence the *forward* and *backward* encoding of half-tiles (see the outline of the DNA grid in 4.1.1).

[1] We use "directionality" instead of the widely-used "orientation" in referring to a DNA sequence's 5'-3' and 3'-5' conformations to avoid ambiguity with "orientations" in the context of tiling.

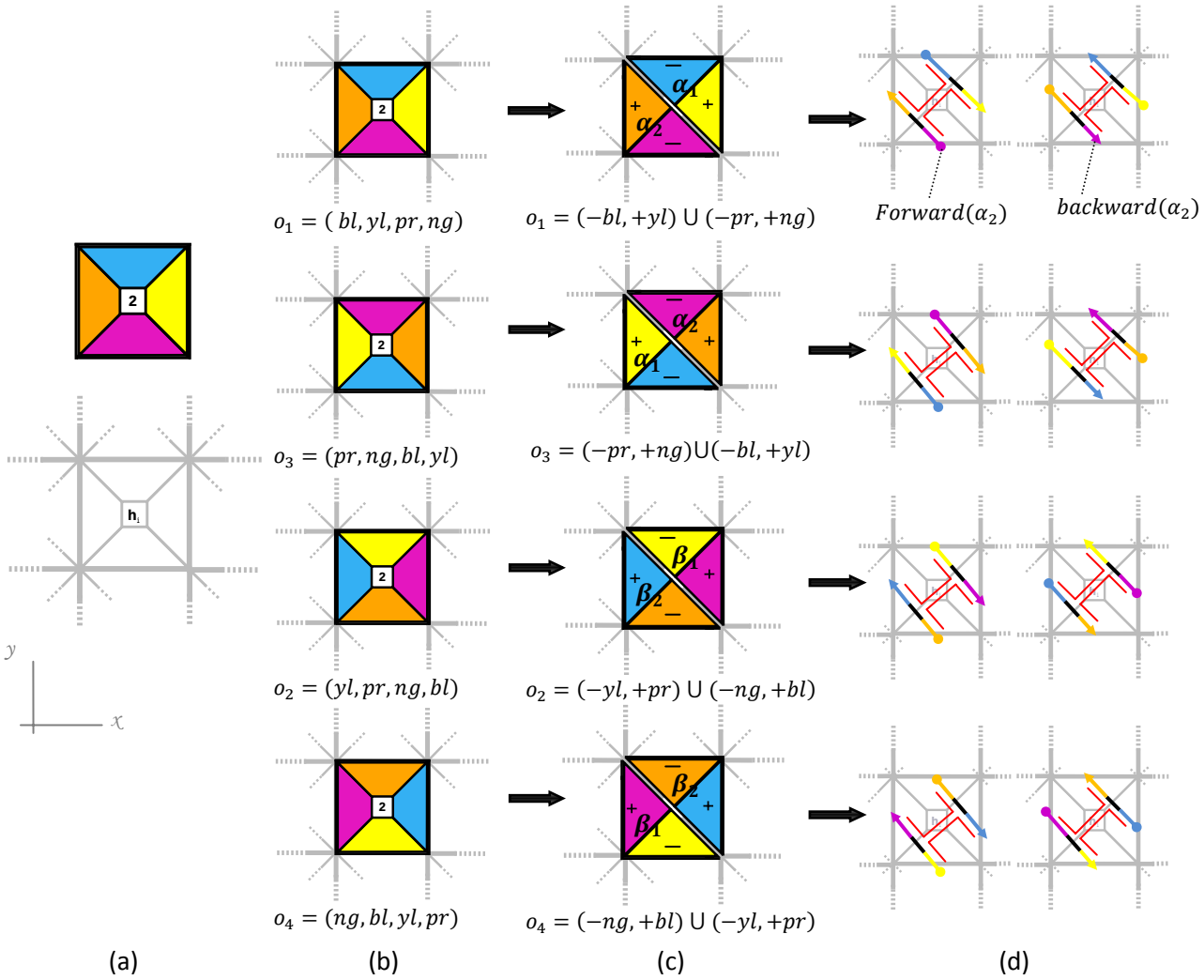


Figure 4.5: The big picture. a) An example tile $t_2 \in$ Hypatia, with some placeholder on the tiling grid. **b)** The corresponding four orientations in t_2 which can be placed on the tiling grid (see 2.2). **c)** The α and β pairs of half-tiles of t_2, the union of which can re-produce one of four orientations (see 3.2). **d)** For each α/β pair of half-tiles, the forward (backward) sequence of a half-tile is bridged to the forward (backward) sequence to reproduce the mohter tile in an orientation. Because DNA has two directionalities, 5'-3' and 3'-5', each half-tile must be able to assume both directionalities (see the DNA grid in 4.1.1), hence the encoding of both forward and backward sequences for each half-tile (2 directionalities x 4 orientations = 8 possible placements of a tile on the DNA grid). Notice that a 5' and 3' ends of a sequence are denoted by a solid circle and an arrow, respectively.

The table below is a summary of the DNA encoding of Hypatia:

	Length of sequence	Number of sequences
Half-tiles	26-bp	122 [=2 for seed + (15x8=120) for others]
Staples	16-bp	14
Bridges	26-bp	122
		Total =258 sequences

Table 4.1: A summary of DNA sequences required for a DNA-based solution of Hypatia.

4.2 Stapling Protocol

In the stapling phase, sequences encoding half-tiles, which we henceforth refer to as DNA(Δ), are mixed with the set of all stapler strands, which we henceforth refer to as DNA(S). The protocol follows three basic steps:

Step 1: Generate random stapling by mixing DNA(Δ) and DNA(S).
Step 2: Keep only lanes that begin and end with "grey".
Step 3: Keep only lanes of length 1, 3, 5, and 7 half-tiles.

The first step is in effect an exhaustive search to generate every possible lane. Clearly, the generation is bounded by the concentration of sequences of DNA (S) and DNA (Δ). The question then arises: what concentration of each distinct species must necessarily be present for a full exhaustive search of a lane of length i? Let $S = \{s_1, \dots, s_r\}$ be the set of all distinct species, and $K_i \subset S$ bet the set of species that staple to s_i on one end. Consider K_u: $\forall K_i, |K_u| \geq |K_i|$, then an upper-bound on the number of possible lanes of length i is given by:

$$\binom{|S|}{1} \cdot \binom{|K_u|}{1}^{i-1}$$

An upper-bound for an exhaustive search of all lanes of length 1, 3, 5,.. i is therefore given by:

$$\binom{|S|}{1} \cdot \left[1 + \binom{|K_u|}{1}^{2} + \binom{|K_u|}{1}^{4} \dots + \binom{|K_u|}{1}^{i-1}\right] = O(|K_u|^{i-1})$$

For Hypatia, $|S| = 122$, $|K_u| = 12$ and $i = 7$, and therefore the concentration of each distinct species should be at least $O(12^6) \approx 3 \times 10^6$ copies of each distinct species in total, which is equivalent to 2×10^{-6} picomoles. Vendors provide standard DNA synthesis at nanomole scales, and so such upper-bound is considered low.

As the previous discussion relates to the question of *how much* computation is needed, another relevant question is *how long* the computation takes. The progress of the random stapling can be modelled by the differential equation:

$$\frac{dQ}{dt} = -\lambda Q$$

where Q is the quantity of free (non-stapled) ssDNA and $\lambda > 0$ is the reaction forward constant[2]. Solving the equation we have:

$$\frac{dQ}{Q(t)} = -\lambda dt \quad \Rightarrow \quad \ln Q(t) = -\lambda t + C \quad \Rightarrow \quad Q(t) = e^{C} e^{-\lambda t} \text{ (by integration)}$$

Since $Q(0) = e^{C}$, we have:

$$Q(t) = Q_0 e^{-\lambda t}$$

The computation proceeds towards a halt state as $Q(t)$ approaches zero. In practice, we have experimentally determined that the course of several hours is considered sufficient (though different buffers and reaction conditions may present different outcomes, see the experimental results below).

In Step 2 and 3, lanes of the right begin/end sequences (i.e. $\pm grey$) and of the right length (i.e. lanes of length 3x(26)=78, 5x(26)=130 and 7(26)=182 base-pairs) are selected. Lanes of length 1 (which encode half-tiles of corner tiles that have $\pm grey$ at both 5' and 3' ends) need not be involved in stapling. Notice that any lane of odd length with $\pm grey$ extremeties necessarily has a $-grey$ in one end and a $+grey$ on the other (see 3.2.2).

[2] In a sense, λ is not really a constant since it encompasses many factors (temperature, pressure, acidity etc.) which are the focus of chemical kinetics –the details of which are beyond the scope of this thesis.

4.2.1 Implementation:

Denote the salt-adjusted melting temperature of ssDNA staples DNA(S) as $T_m(S)$[3].

Step 1:

Phosphorylation of DNA (Δ): As a prerequisite to ligation, oligonucleotides (BioCorp, Montreal) are phosphorylated using T4 Polynucleotide Kinase[4] enzyme (PNK), which catalyzes the transfer of γ-phosphate from ATP to the free hydroxyl of the 5'-terminus. PNK can be used with a T4 Ligase reaction buffer instead of the supplied buffer since ligation follows immediately. The T4 Ligase buffer already contains ATP (which is missing from the PNK reaction buffer). 10 units of PNK is used to phosphorylated 200 pmol of ssDNA, with 5 µl of 10x T4 Ligase buffer (500 mM Tris-HCl, 100 mM MgCl2, 10 mM ATP, 100 mM Dithiothreitol, pH 7.5), bringing the total volume to 50 µl by adding ddH_2O. The mix is incubated at $T_m(S)$–5 for 30 minutes. The final molarity of the oligonucleotides must be re-calculated and if it is severely lower that what is desired for the hybridization/ligation step, it can be increased by various methods (which vary in efficiency and % product loss [44][16]). Alternatively, oligonucleotides can be ordered already phosphorylated, but that adds considerably to the cost of synthesis.

Hybridization/Ligation: Stoichiometric amounts of DNA (S) and phosphorylated DNA (Δ) are mixed. The DNA (S) should be diluted to a very high molarity so that its contribution to the total volume of the hybridization/ligation reaction is minimal. This eliminates the need to adjust the buffering capacity of the mix as DNA (Δ) is already in the right buffer. In addition, 10 units of T4 DNA Ligase

[3] Salt-adjusted T_m of stapler strands should ideally be the same or within a very small range (in which case their average is taken).

[4] Unless stated otherwise, all materials are supplied by New England BioLabs, MA.

enzyme per 200 pmol of DNA (Δ) is added (the enzyme closes DNA "nicks" at the 5' to 3' join points of oligonucleotides) and the total volume is brought to ≤100µl by adding ddH_2O. It is important that the pH level is measured and adjusted to 7.5-8.0 by the addition of a base (e.g. NaOH) or an acid (e.g. HCl) to increase or decrease the pH, respectively. The mix is incubated at room temperature for 0.5-3 hours. Note that the original sequence design must be optimized to minimize hairpins, self- hybridization and un-intended partial hybridization [5].

Step 2:

Polymerase Chain Reaction (PCR): A single PCR reaction mix consists of: up to 200 pmol of the ligation product, 200 pmol of primers (forward and reverse primers for both 5'-($\pm grey$)-3' and 3'-($\pm grey$)-5' 8-bp subsequences[6]), 5 units of Taq DNA polymerase, 5 µl of 10x dNTP, 5 µl of 10X ThermoPol Reaction Buffer, bringing the total volume to 50 µl by adding ddH_2O. The mix is brought to 95 °C for 2 minutes, followed by 32 cycles of: 94 °C (1 min), average[7] salt-adjusted primer's Tm−5 °C (1 min), 72 °C (30 sec). A final elongation step is added by bringing the mix to 72 °C (5 mins)[8], then the mix is stored at ≤4 °C.

Step 3:

Gel Electrophoresis: The PCR product is loaded into an ethidium-bromide (EtBr)-stained 0.5%-1% agarose gel, along with appropriate amount of loading dye. The gel is subjected to 80-95V electrical field in an electrophoresis apparatus, for 45-

[5] The so-called "negative design problem for DNA" can be stated as finding "sequences that maximize the free energy difference between the desired conformation and all other possible conformations"[61]. See also [49][51]. Also see 5.1 on how buffering conditions can reduce the the constraints of this optimization problem.

[6] $\pm grey$ primers are different in lane 1, 7 from those in lane 3 and 5 (see 5.1.1).

[7] Ideally, primers should have the same salt-adjusted T_m (melting temperature).

[8] 72° is the widely-accepted temperature at which the Taq polymerase enzyme starts replicating a DNA strand.

60 minutes. Better band separation can be achieved with lower voltage, but longer running time is then required. In general, the electrophoresis should be terminated when the dye has visibly migrated 2/3 of the gel's length. In addition to the PCR products, a DNA ladder (Fermentas, ON) of known band lengths (with band resolution close to that of the expected bands from the PCR product) is loaded in one or more wells of the gel.

Excision and Purification: The agarose gel is exposed to ultraviolet light and bands of various lengths in each well containing the PCR product can be seen and compared across to bands from the DNA ladder's well (EtBr, which binds to dsDNA as it migrates through agarose gel, fluoresces under ultraviolet light). Bands with the right length are marked and excised using a sterile razor. Subsequently, the dsDNA product in each excised band is extracted using an agarose gel purification kit[9] (Bio Basic Inc., ON).

We apply the presented protocol above on a sample set of ssDNA, schematically depicted in Figure 4.6a, where three and five ssDNA oligonucleotides (coloured blue and red, respectively) are joined with two and four complementary stapler strands, respectively, to produce two lanes of length 91 and 151-bp[10]. In this schematic representation, bullets and arrows indicate 5' and 3' ends, respectively. Figure 4.6b shows a schematic representation of two lanes after the hybridization and ligation step. Notice that the oligonucleotides are ligated at the meeting points (called DNA nicks) after having been complemented by stapler strands, effectively transforming them into one continuous strand of DNA. The exponential replication of

[9] The efficiency of the purification method/kit must be examined carefully to avoid loss of dsDNA.
[10] The exact DNA sequences of these strands can be found in Appendix A. Notice that the length of these oligos does not conform to the 26-bp length specified in our design. However, since bp length is irrelevant in stapling, we chose to use these oligos (which are available at our labs) and avoid further ordering.

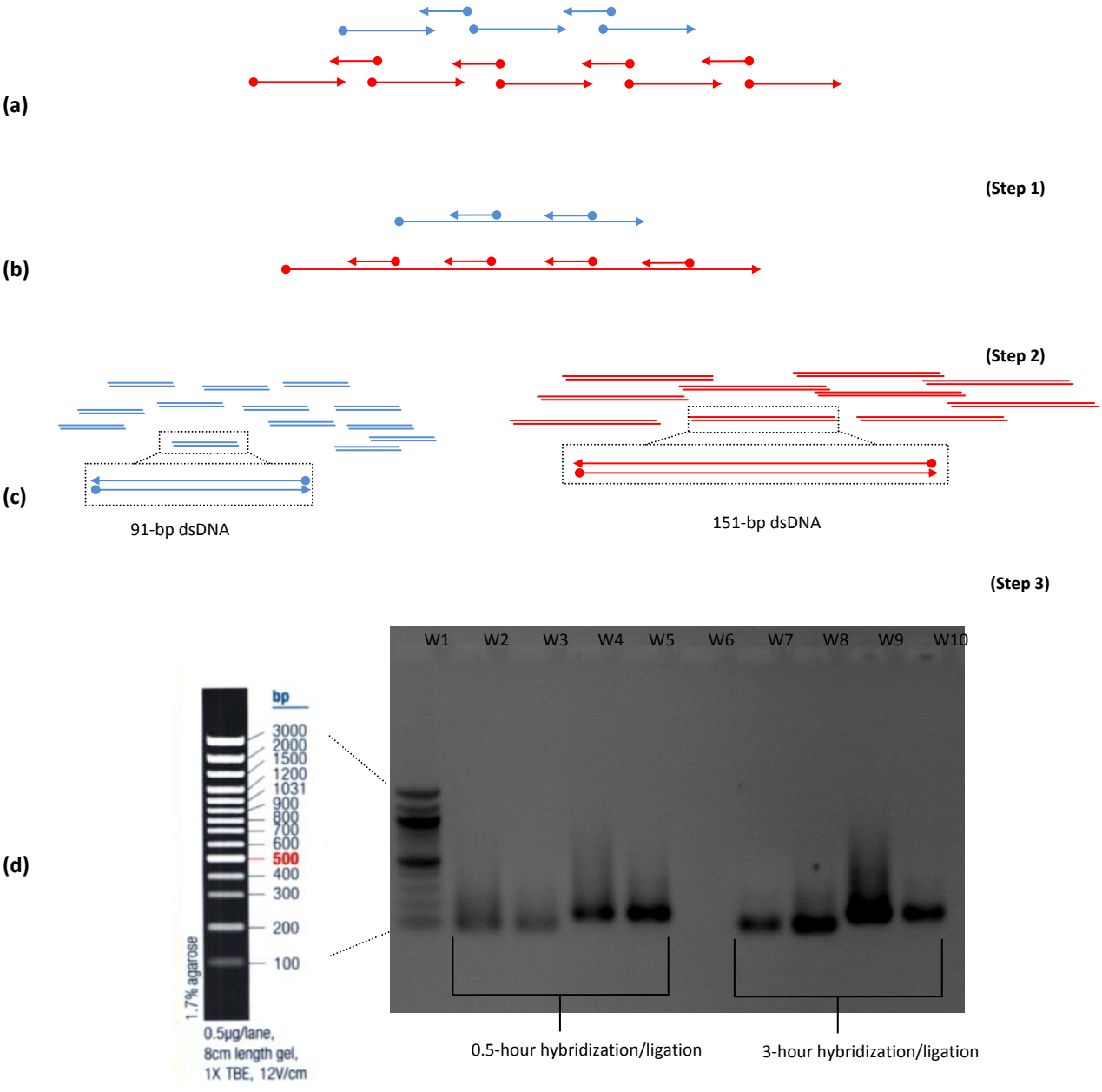


Figure 4.6: Results of stapling protocol. a) Two sets of ssDNA along with stapler strands, bullets and arrows indicate 5' and 3' ends respectively**. b)** Schematic representation of post-hybridization/ligation step. Stapler strands join ssDNA oligonucleotides through Watson-Crick complementarity, while the ligase enzyme closes the "nicks" at the join points, effectively transforming the 3 and 5 individual strands of each lane into one continuous ssDNA of length 91 and 151 respectively (individual stapled strands are of length 30-31 bp each, see Appendix A.1 for the exact sequences). **c)** Schematic representation of post-PCR product. The exponential replication of each lane produces perfectly dsDNA. **d)** Gel electrophoresis results of the PCR product. Well 1 contains a standard DNA ladder (left image) against which the length of resulting bands of PCR product are compared. PCR using templates from two different hybridization/ligation reactions (a 0.5 and 3-hour hybridization/ligation incubation time, respectively). Each lane is amplified separately two times per reaction. Well 2-3, 4-5 show the result of amplifying the 91 and 151-bp lanes, respectively, from a 0.5-min hybridization/ligation. Well 6 is intentionally empty. W7-W8, W9-W10 show the result of amplifying the 91 and 151-bp lanes, respectively, from the 3-hr hybridization/ligation.

lanes (using a set of forward and backward primers for each lane) results in perfectly double-stranded DNA (dsDNA) strands, depicted schematically in Figure 4.6c. The inset shows a magnified view of a dsDNA lane.

In Figure 4.6d, we show the results of the PCR products whose templates come from two separate hybridizations/ligations reactions that were carried for 0.5 and 3 hour incubation times, respectively. From each reaction, each lane is PCR-amplified twice (hence two wells per lane per incubation method). Notice that the 3-hour ligation clearly shows better efficiency, measured by the % Q_0 that has in fact been involved in stapling (given the original molarity of oligonucleotides in a reaction, and the measured molarity of the PCR product which can spectrophotometerically-quantified). Longer hybridization/ligation incubation times did not show greater efficiency (data not shown). The final molarity should meet upper-bound requirements for bridging (see next section).

4.2.2 Verification:

The results presented above were confirmed by sequencing. Two agarose gel bands, corresponding to the two PCR products of the trial experiment above (Wells 7 and 9 for the 91 and 151-bp products respectively), were excised and purified (see step 3 above). The two products were transformed and cloned (Sticky-End Cloning Protocol, CloneJET™ PCR Cloning Kit, Fermentas, ON) into a plasmid (GeneJET™ Plasmid Miniprep Kit, Fermentas, ON) and sent for sequencing. The two products were sequenced (McGill University-Genome Quebec, Montreal). The sequencing results, which provide positive confirmation of the success of the stapling protocol, are presented in Appendix A.1.

4.3 Bridging Protocol

Let l_i the the multi-set of lanes of length i resulting from the stapling phase. All lanes begin and end with a *grey* sequence by now and $i \in \{1,3,5,7\}$. The bridging protocol proceeds as follows:

Step 1: Generate random bridging between l_1 and l_3, assign the result to multi-set $l_{1,3}$

Step 2: Generate random bridging between $l_{1,3}$ and l_5, assign the result to multi-set $l_{1,3,5}$

Step 3: Generate random bridging between $l_{1,3,5}$ and l_7, assign the result to multi-set $l_{1,3,5,7}$

Step 4: Generate random bridging over $l_{1,3,5,7}$, assign the result to multi-set l_{final}

The bridging protocol above is in effect a successive exhaustive search over lanes. There are 7 distinct l_1 species[11] in total in Hypatia (1 from seed, 2 per each other corner tile). Furthermore, there are 44 distinct sequences that a lane can begin/end with[12]. Given that a lane of length , $j \in \{3,5,7\}$, is at this stage made up of two beginning/ending sequences encoding a grey attribute and $j-2$ "internal" (containing no $\pm grey$ subsequences) sequences, an upper bound of the post-stapling concentration of lane l_j is therefore given by:

$$\binom{44}{2}\binom{71}{j-2}$$

where 71 is the sum of $1(1)+3(2)+4(8)+8(4)$ distinct internal sequences: 1 seed, 3 corner, 4 inner and 8 boundary tiles contribute 1, 2, 8 and 4 sequences, respectively[13]. An upper-bound of the concentration of each lane for an exhaustive bridging is therefore given by:

[11] Recall that a valid l_1 is simply a 26-bp sequence encoding "$\mp$grey" at both extremeties. These sequences are not in fact involved in stapling, and are used in bridging directly from synthesis (see the stapling protocol above).

[12] In general, as per 5.1.1 encoding, there are $4(n-5)$ of such sequences for a puzzle of n tiles. Recall that the seed tile does not contribute to the sum of these sequences (because its β-pair is not encoded). The other $n-5$ tiles (possessing one or two grey edges) each contributes 4 sequences.

[13] Notice that 71+44+7 = 122 the total number of half-tile sequences in Hypatia.

$$7 \cdot \binom{44}{2}^3 \cdot \binom{71}{1} \cdot \binom{71}{3} \cdot \binom{71}{5} = 3.1 \times 10^{23} \approx 2 \times 10^{12}\ pmol$$

If the concentration of a post-stapling lane is at a lower molarity p-picomole, its concentration must be brought to the above upper-bound by:

$$(2^c)p = 2 \times 10^{12}\ pmol$$

where c denotes the number of PCR cycles, so:

$$c = \log_2\left(\frac{2\times10^{12}}{p}\right) \approx 41 - \log_2(p)$$

Hence, if the stapling products exist in say, 10^4 pmol concentration, then PCR must be carried out for $c = 41 - \log_2(10^4) \approx 28$ cycles before starting the bridging protocol (each cycle takes 2.5-3 minutes). Notice that aiming for upper-bound concentrations makes safe the assumptions of an error-free and fully efficient PCR (which, in practice, is not the case). Moreover, upper-bounds do serve as a reminder of the exponential growth of required computational resources when dealing with inherently intractable problems. Upper-bounds can however hugely overstate what is actually needed, and in practice lower or average bounds should also be considered.

4.3.1 Implementation:

See 4.2.1 for the **phosphorylation of ssDNA bridges** DNA (*B*).
The following lab operations will be referred to throughout the protocol:

Hybridize $(X, Y, .. Z, T_m)$: Add stoichiometric amounts of products $X, Y, .. Z$ in a 10x T4 Ligase buffer (500 mM Tris-HCl, 100 mM $MgCl_2$, 10 mM ATP, 10 mM Dithiothreitol, pH 7.5), bringing the total volume to 50-100µl by adding ddH_2O. Bring the mix to 95 °C for 5-10 minutes[14] then

[14] This is when the dsDNA lanes are denatured into two ssDNA (only one of which is the subject of bridging). Rapid decrease in temperature to T_m–5 °C prevents renaturation of complementary strand from interfering with bridging.

immediately to T_m–5 °C for 8 hours, then store at ≤ -4 °C until later use.

Ligate (X): Add 10 units of T4 Ligase enzyme per 200 pmol of oligonucleotides in X and incubate at room temperature for 1 hour.

Load $(X, Y, .. Z, \equiv)$: Load products $X, Y, .. Z$, along with ≥50 ng of 50/100-bp DNA ladder (Fermentas, ON) denoted here by " ≡ ", into a 5% polyacrylamide gel. Glycerol (Sigma-Aldrich, MO) and loading dye (max 1-2 µl) should be added to each load[15]. Run the gel for 2.5 hour at 40V using a 1x TBE running buffer (10.8g Tris base, 5.5g Boric acid, 0.93g EDTA, pH 8.0), keeping the electrophoresis apparatus (BIO.RAD PROTEAN®II xi CELL) at ≤ 4 °C throughout the run[16]. The running buffer may (optionally) be changed every hour. The gel is ideally pre-prepared, stored at 4 °C overnight and pre-electrophoresed (to remove possible charged contaminants that may interfere with DNA migration).

Excise (D, bp) : Immerse the gel in 100mL ddH_2O and stain the polyacrylamide gel with 10-20 µl of Ethidium bromide (EtBr)[17], in as low-light room conditions as possible, for 20-30 minutes. Destain the gel in ddH_2O for 20-30 minutes. Cut the polyacrylamide band D which

[15] The dye helps visual tracking of product migration, and viscous glycerol holds products down the wells.
[16] So as to avoid heat generated by electrophoresis from affecting the resistance of the gel unevenly bands travel slightly faster the closer they are from the center).
[17] EtBr is a known mutagen and must be handled with extreme caution.

contains bp dsDNA under ultraviolet exposure, using a sterile razor blade.

Purify (D) : Purify the excised band *D* (QIAquick® Gel Extraction Kit (50)). **PCR-amplify** if deemed necessary (e.g. using spectrophotometric quantification) using $\pm grey$ primers.

Denote the salt-adjusted melting temperature of ssDNA bridges (or DNA(B)) as $T_m(B)$[18], the bridging protocol proceeds as follows:

Step 1:

$$\overline{l_1} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(l_1, \mathrm{DNA}(B), T_m(B))\right), \equiv\right), 26\right)\right)$$ [19]

$$l_{1,3} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(\overline{l_1}, l_3, T_m(B))\right), \equiv\right), 52\right)\right)$$

Step 2:

$$\overline{l_{1,3}} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(l_{1,3}, \mathrm{DNA}(B), T_m(B))\right), \equiv\right), 104\right)\right)$$

$$l_{1,3,5} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(\overline{l_{1,3}}, l_5, T_m(B))\right), \equiv\right), 156\right)\right)$$

Step 3:

$$\overline{l_{1,3,5}} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(l_{1,3,5}, \mathrm{DNA}(B), T_m(B))\right), \equiv\right), 234\right)\right)$$

$$l_{1,3,5,7} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(\overline{l_{1,3,5}}, l_7, T_m(B))\right), \equiv\right), 312\right)\right)$$

Step 4:

$$l_{final} = \textbf{Purify}\left(\textbf{Excise}\left(\textbf{Load}\left(\textbf{Ligate}\left(\textbf{Hybridize}(l_{1,3,5,7}, \mathrm{DNA}(B), T_m(B))\right), \equiv, c_{1,3,5,7}\right), 832\right)\right)$$

[18] Salt-adjusted T_m of ssDNA bridges should ideally be the same or within a very small range (in which case their average is taken).

[19] $l_x/\overline{l_x}$ is to be interpreted in this context as "tube $l_x/\overline{l_x}$" (an actual physical lab tube!). 26 is the base-pair length of the expected right band at this step.

Figure 4.7 shows a step-wise schematic representation of the bridging protocol. Notice that the next lane to be bridged is mixed *only* with lanes that have succeeded to bridge in the previous step (because those that failed have been excluded by virtue of Excise()/Purify() operation that follow each Hybridization()). For example, if an l_3 species failed to bridge with any l_1 during the second operation of Step 1, that l_3 will have been excluded before the first operations of Step 2 is begun. Furthermore, species which have in fact succeeded to bridge at any given step can be further PCR-amplified.

The protocol in fact allows for the insertion of PCR-amplification at any stage, although that increases the number of excision and purification operations. Another challenge in the amplification of constituent lanes of a partial assembly is that, due to ligation, bridges that have a 5' 3' meeting are now one continuous strand (in Figure 4.7, for example, the trail of bridges starting at the 5' end of l_1 to and ending at one of the middle bridges of l_5 is –after denaturation of the assembly– a ssDNA of length 78 bp. Since such sequences can only have one grey end at the most, they cannot produce exponential replication (linear amplification may however take place). Moreover, if a partial assembly, say $l_{1,3,5}$, is amplified, then the constituent lanes will all be within the same PCR reaction. Hence, and in order to enforce the step-wise bridging, those PCR-ed lanes must be separated once more (by Load(), Excise(), and Purify() operations, consecutively) and step-wise bridging is restarted.

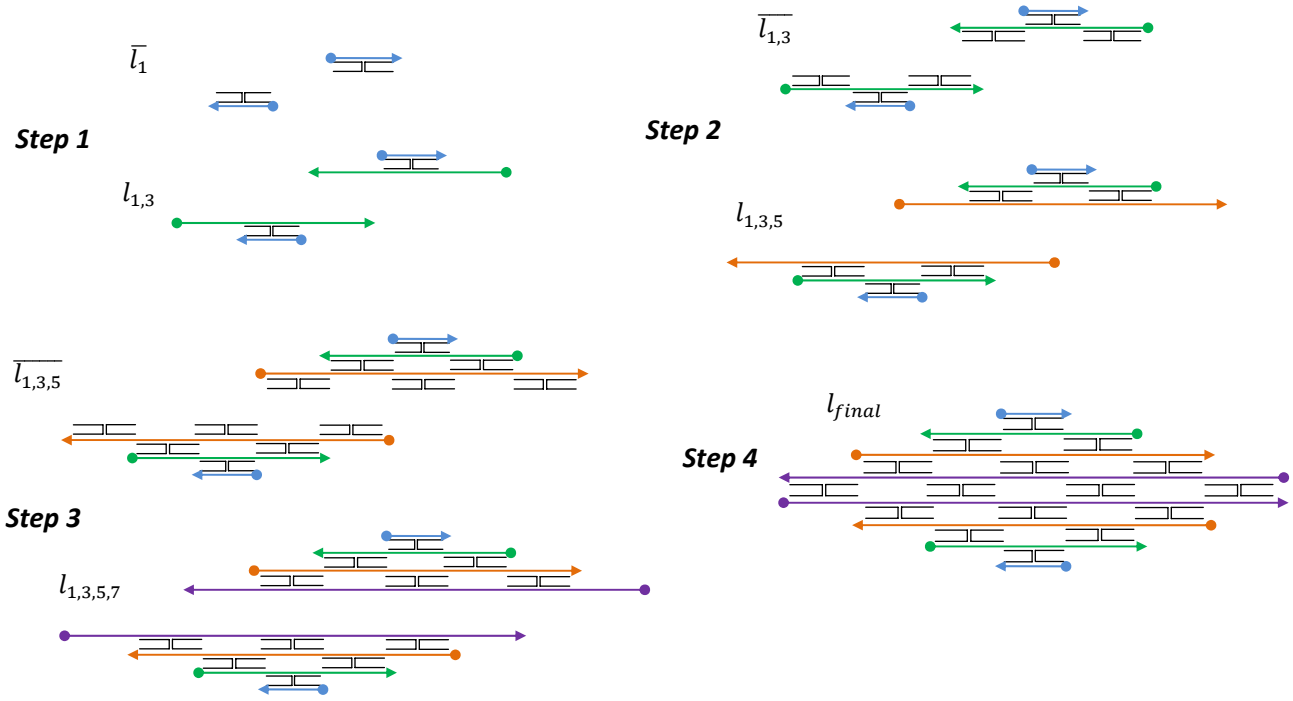


Figure 4.7: Progression of the bridging protocol: schematic representation of the resulting DNA scaffolding assemblies at each step. The transition from one set to another within a single step entails an excision and a purification of the correct assembly (*correct* as judged by the level at which a band appears on the ultraviolet-exposed polyacrylamide gel, measured against a DNA ladder while taking into account mobility issues of DNA junctions in general, see discussion). Furthermore, a PCR-amplification operation can be inserted at any stage of the protocol. Notice that in Step 4, the set of bridges DNA(B) is added to $l_{1,3,5,7}$ which may result in two half-assemblies (corresponding to half of the DNA grid), which should have been bridged to form one assembly, instead each hybridizes independently to bridges, in effect preventing the crystallization of a full grid assembly. Hence, l_{final} may be preceded with intensive amplification followed by the addition of DNA(B) in smaller stoichiometric quantities. The detection of l_{final} however is unambiguous since its base-pair size is double that of any other assembly at this stage.

What does an insertion of a PCR-amplification operation really mean? Recall from Chapter 3 that the solution to Hypatia can be stated as "finding a set of lanes that can successfully stack up and cover the entire tiling grid". Hence, the amplification is in a sense an exclusion heuristic: if a lane failed to bridge at Step *s*, it is definitely not part of *the* ultimate solution set of lanes. If a lane succeeded to contribute to a partial assembly, then it's *more likely* to be part of the solution set (but its likelihood of being such will be tested once more in the next bridging step).

The excision/purification operations deserve special attention as the risk of losing products during these operations poses a real challenge. Furthermore, to determine exactly which band to excise is complicated by the fact that the gel mobility of DNA assemblies is far from being fully understood. Many factors, including running voltage, salt concentration and polyacrylamide percentage do affect mobility [50][22]. Hence, in addition to careful optimization of such factors, other controls can also be loaded to help identify the right bands with more precision.

Step 4 in the protocol above is particularly challenging. When partial assemblies in $l_{1,3,5,7}$ are hybridized with bridges, two assemblies which should ideally be bridged to form the full DNA grid, may instead each hybridize to bridges independently, rather than being brought together with the same set of bridges. One way to reduce such outcome is to PCR-amplify $l_{1,3,5,7}$ to a very high molarity (after all, $l_{1,3,5,7}$ contains lanes which by now have a very high probability of being the correct solution set). Next, the amplified lanes are bridged step-wise again, only this time the ssDNA bridges in Step 4 are added at smaller stoichiometric amounts vis-à-vis $l_{1,3,5,7}$. Another technique would be to use a magnetic bead system [4] separate assemblies in $l_{1,3,5,7}$ which contain the seed tile from other assemblies. ssDNA bridges are then added solely to this set and –once hybridized, ligated and purified (to remove floating ssDNA bridges)– is then mixed with other remnant assemblies.

What about erroneous bridging? Consider the snapshot of a partial assembly in Figure 4.8. At some step, a lane (coloured green) stacked erroneously with a shift of one position to the right. Recall from 4.1.3 that an underlying principal of stable DNA assembly states that there must be an odd number of half-turns between crossover points, which necessarily means that the conformations of the double-helix in these two points are at a 180 °C difference in the 3-dimensional space. Hence, the two crossover points p_1 and p_2 in Figure 4.8 must necessarily

attach *up* and *down*, respectively. Now, assuming that this assembly survived the excision operation, four scenarios can take place: 1) Both bridge to the same lane, 2) Both bridge but to different lanes, 3) Both remain free, and 4) one bridges to a lane and the other doesn't. The first and second cases can immediately be ruled out as both unstable assemblies (see also Winfree's discussion on an assembly's "temperature"[63]). Such assemblies would also be at a great contrast in terms of migration mobility vis-à-vis the right assemblies at this step [50][22]. Hence, the instability and/or exclusion by excision provide for a safe assumption that such erroneous assemblies are eliminated. In the third and fourth scenarios, consider the worst case in which such assemblies survived till last step (again, an unlikely outcome considering migration mobility). In such case, and as a final exclusion heuristic, l_{final} can be stained with ssDNA-binding dye (OliGreen® ssDNA Quantitation Assay and Kit, Invitrogen, ON) and any fluorescent bands are excluded (the desired assemblies in l_{final} contain absolutely no ssDNA segments; the final DNA grid is fully dsDNA).

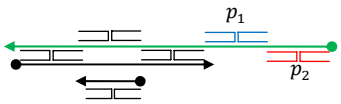


Figure 4.8: Erroneous bridging results when a lane stacks with one or more shifts left or right. Here, the green lane stacks with one shift to the right. Considering the requirements of the DNA assemblies, bridging at the two crossover points must be at a 180° difference in the 3-dimensional space. Furthermore, any subsequent scenarios involving such erroneous assembly leads to a either 1) a migration mobility contrast to the correct assemblies, and hence such assemblies get ruled out after the excision operation, or 2) the existence of ssDNA in the assembly which can then be detected and excluded using a ssDNA-binding dye (see discussion).

In what follows we present the results of applying the above protocol on a subset of ssDNA oligonucleotides (Integrated DNA Technologies, Inc., Illinois) emulating a two- and three-lane assembly. Figure 4.9a shows a schematic representation of the two assemblies (see Appendix A.2 for exact DNA sequences). The three lanes are coloured blue, green and red; the pair of bridges in black; and the dashed-lines denote ssDNA "fillings" which complete the

double-helix between two bridging areas (bridges and fillings are all 16-bp sequences). The three lanes involved are of length 30, 60 and 60 bp and the DX points are at a distance of 3 half-turns (recall the requirement of odd number of half-turns between crossover points, see 4.1.3). Since the bridging protocol assumes post-stapling products in dsDNA form, the lanes used in this demonstration are also dsNDA.

This is crucial, since the assumption that when dsDNA is denatured and the reaction conditions are set for bridging to take place (see Hybridize() operation above), the complementary strands of the denatured dsDNA will not be interfering with bridges –by hybridizing (fully or partially) to their complementary strands (which are our lane of interest). We hypothesized, and the presented results confirm, that after denaturation, if 1) the reactions conditions (mainly temperature) are rapidly set for bridging, and 2) the molarity of bridges is vastly higher than that of lanes (we used 50x or 100x bridge to lane ratio) then the complementary strand would pose no interference.

Figure 4.9b shows a picture of the ultraviolet-exposed EtBr-stained polyacrylamide gel. From left to right, wells 1, 3, 7, and 9 contain a 100-bp DNA ladder, while well 5 contains a 50-bp DNA ladder. The scale of each ladder is shown. Notice that, despite the fact the electrophoresis was done at 4 °C, the “smiley face” effect can still be seen: a band travels slightly faster the closer it is to the centre of the gel due to generated heat. Further investigation into the possibility of optimizing the electrophoresis, by altering the salt concentration and/or the % polyacrylamide for example, is clearly needed. Wells 2 and 4 contain the 3- and 2-lane assemblies, respectively. The trail in these wells reflects the left-over ssDNA bridges, which were added in the hybridization operation to a 100x ratio to lanes. Fully formed 2- and 3-lane assemblies contain 96 and 112 double-stranded base-pairs in total (see Appendix A.2 for detailed sequences). As compared across with DNA ladder, and taking the heat-induced

distortion of migration into account, the two visible bands (marked with red in Figure 4.9b) – assumed to be the intended assemblies– were excised. As mentioned earlier, the migration of DNA assembly cannot be determined solely by bp count. Hence, in an extended experiment, we recommend loading further controls using pre-prepared assemblies with known sizes. Well 8 is a control which contains 60-bp dsDNA lanes (the same lanes that are used in the assemblies). Clearly they appear at the expected location when compared across with DNA ladders.

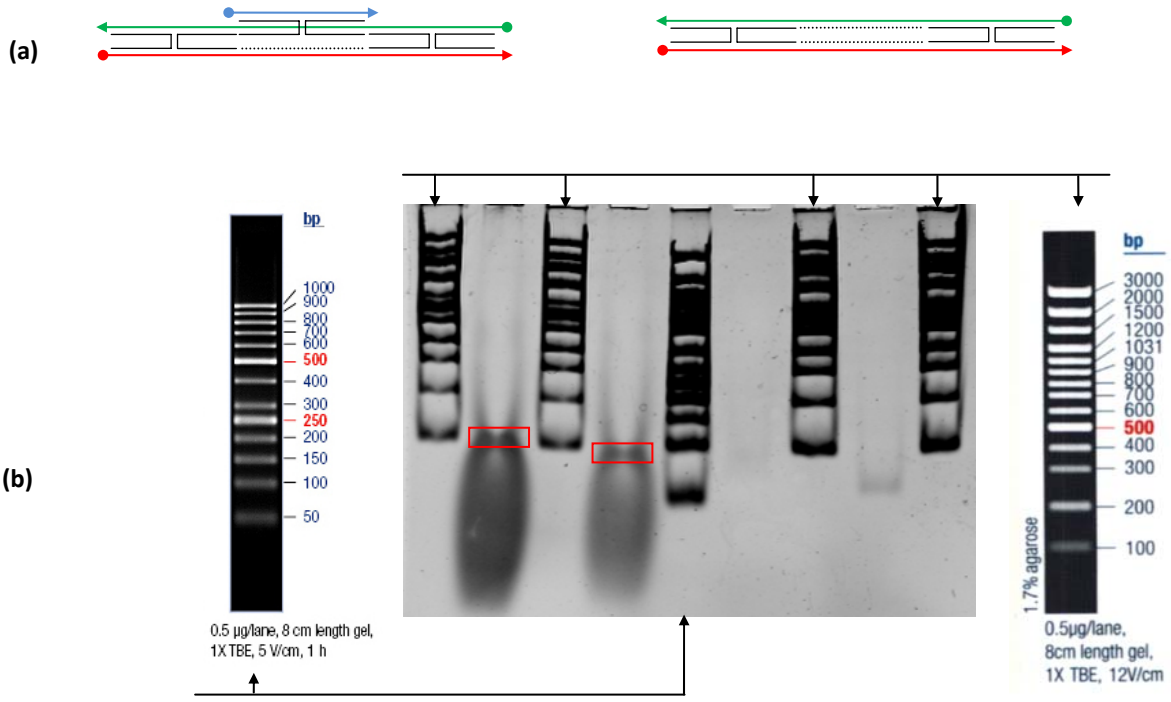


Figure 4.9: Results of bridging protocol. a) Schematic representation of the set of ssDNA oligonucleotides used in this demonstration. The blue, green and red denote 30, 60 and 60–bp lanes; solid black curved lines denote bridges and dashed lines denote filler ssDNA that complement inter-bridging areas. The two assemblies are close in size so as to determine the resolution of assemblies with small differences in bp. **b)** Image of the ultraviolet-exposed polyacrylamide gel. From left to right, wells 1, 3, 7, and 9 contain a 100-bp DNA ladder, while well 5 contains a 50-bp DNA ladder. The scale of each ladder is shown. Notice that, despite conducting electrophoresis at 4 °C, the "smiley face" effect can still be seen: a band travels slightly faster the closer it is to the centre of the gel. Well 2 and 4 contain the 3- and 2-lane assemblies, respectively. The trail in these wells reflects left-over ssDNA bridges, which were added in the hybridization operation at a 100x ratio relative to lanes' concentration. The two visible bands (marked with red rectangles) assumed to be the intended assemblies were excised and purified. Well 8 is a control which contains 60-bp dsDNA lanes (the same lanes that are used in the assemblies, loaded from stock). The 2- and 3- lane assemblies contain 96 and 112 dsDNA in total, respectively.

4.3.2 Verification:

PCR reactions were carried out, using templates from the excised and purified bands shown in Figure 4.9b. The rationale behind using PCR for verification is that, if a band is assumed to contain l_x, l_y, and l_z lanes, then using that band as a template in three separate PCR reactions: the first containing primers of l_x, the second primers of l_y etc. then the three reactions should produce exponential amounts of l_x, l_y, and l_z respectively. The mutual success of all these PCR reactions (as their outcome is loaded into agarose gel and electrophoresed) is a clear indication that all lanes making up the assumed assembly were in fact present in the excised band and so the assembly must have formed.

Figure 4.10 shows the results of PCR verification. For each excised band (red rectangle in Figure 4.9), two PCR reactions were performed, one to amplify the green lane, and one to amplify the red lane. As the result below show, PCR conclusively proves that the two lanes were present[20]. No amplification of the blue lane was performed since the 3-lane assembly (Well 2 of Figure 4.9) appears slightly higher than that of the 2-lane assembly which can only be interpreted as due to added dsDNA from the blue line.

[20] We note here that the same verification experiment was repeated with **primer-dimer controls** (i.e. carrying a PCR reaction that contains the primers but not the template, and if nothing gets amplified, one concludes that there is no unintended amplification in those reactions which do in fact contain templates). The results were once again positive.

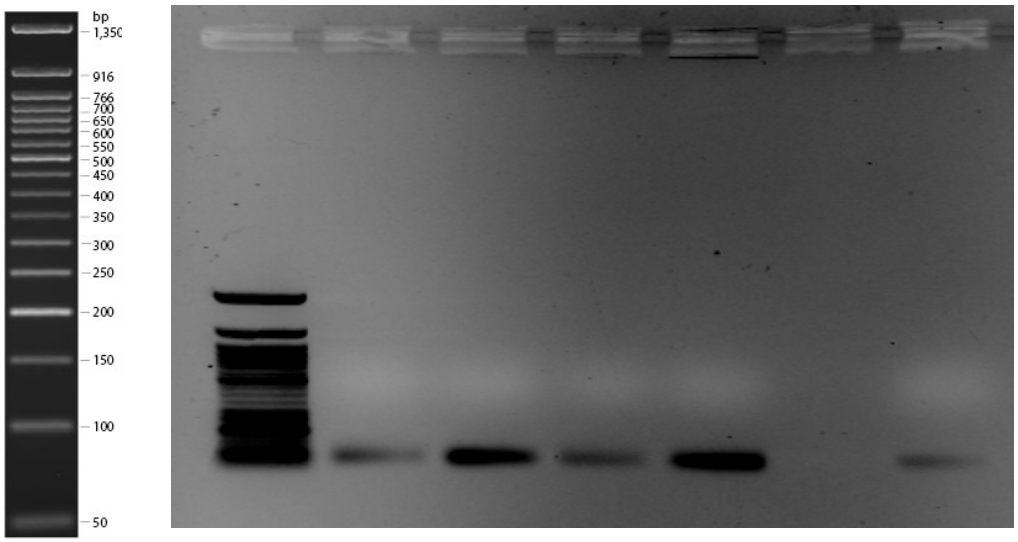


Figure 4.9: PCR-verification of 2- and 3-lane assemblies.
W1: 50-bp DNA ladder
W2: Green lane amplified from the first excised band (60-bp)
W3: Red lane amplified from the first excised band (60-bp)
W4: Green lane amplified from the second excised band (60-bp)
W5: Red lane amplified from the second excised band (60-bp)
W6: <intentionally empty>
W7: control; red and green lanes loaded from stock (60-bp)

Chapter 5 Reflection and Future Work

Our knowledge of the marvelous DNA molecule has come a long way. And we expect to continue to be marveled by new success stories of DNA outside the cell. We say that with confidence that stems from the impressive track record of DNA in the past, well, thousands of millions of years, in which it has been the ever efficient and resilient carrier of massive blueprint information encoding machine instructions of living organisms. We owe this new perspective of DNA, i.e. the computational perspective, to Adleman's insightful HPP demonstration [4]. One can only look at the many new avenues of using DNA for solving problems, fabricating nano-scale structures, experiment with new and interesting models of universal computations and even the strive for "tiny" DNA computers that can intelligently improve the way medical care is provided.

At the heart of all of these various promising prospects lies a central seat for the computationally-minded. Some of the most fundamental questions that arise in the pursuit of all these frontiers are basically computational questions. If a DNA computer is to control gene expressions, and we have a complete map of the genetic regulatory network, then a question as "what combinations of on/off states of genes result in the release of chemical X?" is quickly spotted by a computer scientist as the SAT problem itself. Nano-technologists rushing to design a computer program to generate a set of half-tiles that would grow into nano-scale structures of their interest would be in a much better position with the knowledge that finding such set is a *computationally hard* problem, and that a lot of time and money would be saved if they accepted a set of half-tiles that *approximately but not exactly* produces that shape. DNA can compute, and computer science and the well-established complexity theory tell us what is

computable in the first place, and if so *how much* computation can realistically be done. And so the marriage of the two is the more appropriate.

We have formally defined the half-tile model which we used to devise a DNA-based solution to the NP-Complete bounded edge-matching puzzles. The PCR-enabled half-tile model is also relevant in two other aspects: as a Turing-complete computational model, and as a framework for nano-scale fabrications. We discuss these three aspects further in what follows.

5.1 How Powerful is DNA Computing?

DNA oligonucleotides can be synthesized by commercial vendors relatively cheaply (~$0.08-0.20/nucleotide at the time of writing) and in concentrations of up to micro-mole amounts ($\sim 10^{17}$ copies, maybe slightly more). This is the primary limit on how much computation can be done. Beyond the first pool generation step (which, again, is limited by the molarity of synthesis), PCR comes into play as a supremely powerful, efficient and cheap heuristical and processing-power tool. We must, however, let go of the illusion[1] that the massive parallelism of DNA strand hybridizations can "crack" intractable problems and surpass silicon-based computing. Instead, we ought to precisely characterize what PCR can do and employ that power in areas where DNA Computing is actually needed.

Implicit in our discussion of Chapter 4 is the fact that PCR can make up for low synthesis molarity[2]. In other words, were it not for PCR, we would have to synthesize at much higher molarity so as to be able to yield the correct computation in sufficiently large/detectable amounts *at the pool generation* step. With PCR at hand, however, we can do with as few as

[1] See 1.4 for more on this point.
[2] Compare upper-bounds of the stapling and bridging phases in Chapter 4.

dozens[3] of correct outcomes from the pool generation step –we can fish them out with the help of PCR. The importance of BEMP's NP-Completeness in this context is as the benchmark by which one can make statements about DNA computational power.

Another interesting result, which we have not mentioned in Chapter 4, is that the problem of designing DNA strands to minimize unintended hybridizations and hairpin/secondary structure formations need not be hard. Asking a computer program to design oligonucleotide sequences that are at a maximum Hamming distance can be quite difficult as the pool of strands grows larger. Researchers have explored this problem extensively (see [56b] for example). But we have found that the problem may not be hard in reality. Using Tetramethylammonium chloride (TEACl) or tetramethylammonium chloride (TMACl) salts in reaction buffers can hugely increase the hybridization sensitivity to sequence length [28b].

Using an example from our context, if stapler strands are of length 16-bp, then the design problem need only ensure that no two strands have, say, >12 strands common subsequences, and any commonalities of less (or more in fact) need not be ruled out. TMA^+ and TEA^+ (as opposed to Na^+ in most buffers) seem to cause a remarkable increase in the dependence of melting temperature on sequence length [28b]. Furthermore, as few as 2-bp difference can result in big difference in T_m. This prevents a strand of length x from hybridizing at the melting temperature of strands of length $x \pm 2$. A demonstration of the utility of TEACl and TMACl in DNA Computing is a planned future work. We should also mention that the protocol devised in this work will be carried out on the full set of strands encoding a 4×4 BEMP.

[3] We make this assertion based on successful PCR reactions that we have carried out, in which a small segment from among large and diverse genomic DNA (Arabidopsis) was successfully amplified and with impressive efficiency.

5.2 Half-tile Assembly Model is Turing-complete

Wang tiles have a fixed orientation, and so rotation of tiles is not allowed. In EMPs, rotation is of course needed and so our half-tile model allows for it. But rotation can be disabled in the presented half-tile model by two simple steps: 1) only one pair is encoded (all tiles would be dissected at one and the same diagonal), and 2) Bridging *up* and *down* operations are made distinct by simply having different sequence lengths for the part of the ssDNA bridge that is destined to bridge *up* from that which bridges *down*.

Half-tile assembly model (hTAM) is then Turing-complete because, trivially, the union operation can be applied to get the original mother tiles –and tiles have already been proven to be capable of simulating a Turing machine. That is a weak version of proving the Turing-completeness of the half-tile assembly model. *Weak* in the sense that what we have shown is the equivalence between tiling with half-tiles and tiling with their mother tiles, and concluded that since Tile Assembly Model is Turing-complete then so must be tiling with the *corresponding* set of half-tiles.

But a stronger proof is possible. We can prove that the execution of a Turing machine can be simulated by the growth of some set of half-tiles. Even more is possible, in that diagonals need not be unique to two half-tiles (just like edges can staple to more than one half-tiles all possessing that edge colour). The proof and a demonstration of

the Turing-completeness of the half-Tile Assembly Model (hTAM) using 2-dimensional DNA self-assembly is a planned future work[4].

5.3 Half-tile Assembly Model is a PCR-powered DNA Nano-Construction Model

The PCR-powered half-Tile Assembly Model (hTAM) has potential in nano-scale constructions. Not only do we have a model for defining molecular motifs that –by design– self-assemble into the desired nano-structure, but it is one that allows also for the insertion of PCR every step of the way. A partially fabricated nano-structure can be denatured at will and the constituent lanes of DNA can be amplified once more. With hTAM, DNA nano-technology can be carried out at massive scales. We plan to pursue this avenue further, but with an extra ambition. We seek to investigate the problem of finding a "programmable" (or "re-usable") set of half-tiles $HT = \{ht_1, ht_1, \ldots, ht_n\}$ whereby mixing of different subsets of HT can produce different (yet desirable/useful) assemblies.

[4] This work is being done in collaboration with Ibrahim Al Abdulmohsin (Elec. Eng. Dept., Stanford University) to whom the formal proof is due.

References

[1] Aaronson, S. NP-Complete Problems and Physical Reality. *SIGACT News* **36**:1 (2005), 30-52.

[2] Aaronson, S. The limits of quantum computers. *Scientific American*, **298**(3):62 – 69, Mar. 2008.

[3] Aaronson, S. Speaking Truth to Parallelism. Web. 09 Jan. 2011. <http://www.scottaaronson.com/blog/?cat=17>.

[4] Adleman, L.M. (1994) Molecular computation of solutions to combinatorial problems. *Science* **266**: 1021–1024.

[5] Adleman, L. On the potential of molecular computing. *Science* 28 April 1995: 483-484.

[6] Amos, M. *Theoretical and Experimental DNA Computation*. Berlin: Springer, 2005.

[7] Ansótegui, C., Béjar, R., Fernández, C., Mateu, M. How Hard is a Commercial Puzzle: the Eternity II Challenge. *Proceedings of the 11th International Conference of the Catalan Association for Artificial Intelligence,* 2008.

[8] Bach, E. Condon, A. Glaser, E. and Tanguay, C.. DNA models and algorithms for NP-complete problems. In *Proceedings of the 11th Conference on Computational Complexity*, pages 290–299. IEEE Computer Society Press, 1996.

[9] Baum, E. and Lipton, R.J. (eds.): *DNA Based Computers*. *Proc. of the Second Annual Meeting, Princeton*, 1996.

[10] Baumgardner, J*. et. al*. Solving a Hamiltonian path problem with a bacterial computer. *Journal of Biological Engineering*, **3**(1):11, 2009.

[11] Benenson, Y. et al. An autonomous molecular computer for logical control of gene expression. *Nature* **429**, 423–429 (2004).

[12] Berger, R.. *The Undecidability of the Domino Problem*. Providence, RI: American Mathematical Society, 1966.

[13] Boneh, D., Dunworth, C., and Lipton, R. J. *Breaking DES using a molecular computer*. In Lipton and Baum (1996), pages 37–65.

[14] Braich, R. S., Nickolas Chelyapov, Cliff Johnson, Paul W. K. Rothemund, and Leonard Adleman. Solution of a 20-Variable 3-SAT Problem on a DNA Computer. *Science*: **296** (5567), 19 April 2002: 499-502.

[15] Bunow, B. On the potential of molecular computing. *Science* 28 April 1995: 482-483.

[16] *Concentration of Solution by Evaporation and Lyophilization - Proteins, Peptides, DNA, RNA and Other Solutes*. Web. 28 Dec. 2010. <http://www.labconco.com/product_lit/appnotes/concentration.asp>

[17] Cook, S. (1971). The complexity of theorem proving procedures. *Proceedings, Third Annual ACM Symposium on the Theory of Computing, ACM, New York*. pp. 151–158.

[18] Crick, F. Central dogma of molecular biology. *Nature* **227**, 561–563 (1970, August 8).

[19] Culik, K. II. An Aperiodic Set of 13 Wang Tiles. *Discrete Mathematics* (1996) **160**:245-251.

[20] Demaine E., Demaine M.: Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics 23* (Jun 2007), 195–208.

[21] Devlin, K. J. *The Millennium Problems: the Seven Greatest Unsolved Mathematical Puzzles of Our Time*. New York: Basic, 2002.

[22] Diekmann,S. and Lilley,D.M. The anomalous gel migration of a stable cruciform: temperature and salt dependence, and some comparisons with curved DNA. *Nucleic Acids Res*., **15** ((1987) 5765-5774.

[23] *Eternity 2 Board Game | The Puzzle With The $2 Million Prize - Eternity II*. Web. 20 Oct. 2010. <http://uk.eternityii.com/>.

[24] Garey, M. R., David S. J. *Computers and Intractability: a Guide to the Theory of NP-completeness*. San Francisco: W.H. Freeman, 1979.

[25] Hartmanis, J. On the Weight of Computation. *Bulletin of the European Association for Theoretical Computer Science.* **55**:136-138, 1995.

[26] Holliday, R. A mechanism for gene conversion in fungi. *Genet. Res. Camb*. **5**, 282–304 (1964).

[27] Haubrich, J. Compendium of Card Matching Puzzles. Self-published, May 1995. Three volumes.

[28] Heule, M. J.H. Solving Edge-matching Problems with Satisfiability Solvers. *Proceedings of the Second International Workshop on Logic and Search (LaSh 2008)* (2008): 88-102.

[28b] Jacobs, K.A., R. Rudersdorf, S.D. Neill, J.P. Dougherty, E.L. Brown, and E.F. Fritsch. 1988. The thermal stability of oligonucleotide duplexes is sequence independent in tetraalkylammonium salt solutions: application to identifying recombinant DNA clones. *Nucleic Acids Res*. **16**:4637-4650.

[29] Kari, J. A small aperiodic set of Wang tiles. *Discrete Mathematics* (1996) **160**: 259–264.

[30] Lévy, A. *Basic Set Theory*. Berlin: Springer-Verlag, 1979.

[30] Linial, M., Linial, N. On the potential of molecular computing. *Science* 28 April 1995: 481.

[31] Lipton, R. J. Using DNA to solve NP-complete problems. *Science*, **268**: 542-545, Apr. 28, 1995.

[32] Liu, Y. and West, S. C.: Happy Hollidays: 40th anniversary of the Holliday junction. *Nat. Rev. Mol. Cell. Biol*. **5**, 937-44 (2004)

[33] Lo, Y.M., Yiu, K.F., and Wong, S. L. On the potential of molecular computing. *Science* 28 April 1995: 481-482.

[34] Mao, C., Sun, W., Seeman, N.C. Designed two-dimensional DNA Holliday junction arrays visualized by atomic force microscopy. *J. Am. Chem. Soc.* **121**: 5437–5443 (1999).

[35] Mao, C., LaBean, T. H., Reif J.H., Seeman, N.C. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. *Nature* **407**: 493–496 (2000).

[35b] Megiddo, N. Partial and Cyclic Orders. *Bull. of Am. Math. Soc.* **82** (1976), 274-276.

[36] Ralston,A., Reilly, E.D. and Hemmendinger,D. *Encyclopedia of Computer Science*, 4th Edn, Nature Publishing Group, London, UK (2000).

[37] Reishus, D., Shaw, B., Brun, Y., Chelyapov, N., Adleman, L. Self-Assembly of DNA Double-Double Crossover Complexes into High-Density, Doubly Connected, Planar Structures*. Journal of the American Chemical Society*, **127(50)** : 17590–17591, 2005.

[38] Rinaudo, K. et al. A universal RNAi-based logic evaluator that operates in mammalian cells. *Nature* Biotechnol. **25**, 795–801 (2007).

[39] Rohrig, R. A theory for qualitative spatial reasoning based on order relations. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, volume 2, pages 1418–1423, Seattle, 1994.

[40] Paun, G., Rozenberg, G., and Salomaa, A. *DNA Computing: New Computing Paradigms*. Berlin: Springer, 1998.

[41] Rothemund, P. W. K., Papadakis, N., Winfree, E. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biol* **2:**2041–2053 (2004).

[42] Rothemund, P. W. K. Folding DNA to Create Nanoscale Shapes and Patterns. *Nature* **440**.**7082** (2006): 297-302.

[43] Sa-Ardyen, P., Vologodskii, A. V., & Seeman, N. C. (2003). The flexibility of DNA double crossover molecules. Biophysical Journal, 84, 3829–3837.

[44] Sambrook, J., Russell, D. W. *Molecular cloning : a laboratory manual,* 3rd ed. Cold Spring Harbor, N.Y. : Cold Spring Harbor Laboratory Press, c2001

[45] Schmidt, K. A., Henkel, C. V., Rozenberg, G., and Spaink, H. P. DNA computing using single-molecule hybridization detection *Nucleic Acids Res 23 September 2004: 4962-4968.*

[46] Sakamoto, K. et al. Molecular Computation by DNA Hairpin Formation. *Science* **288**, 1223 (2000);

[47] Seeman, N. C. *et al.* New motifs in DNA nanotechnology. *Nanotechnology.* **9**:257–273 (1998).

[48] Seeman, N. C. An Overview of Structural DNA Nanotechnology. *Mol. Biotechnol*., 2007, **37**: 246–57.

[49] Seeman, N. C. 1982. Nucleic acid junctions and lattices. *J. Theor. Biol*. **99**:237–247.

[50] Seeman N.C., Chen,J.H. and Kallenbach,N.R. (1989) Gel electrophoretic analysis of DNA branched junctions. *Electrophoresis*, **10**, 345–354

[51] Seeman, N. C. De novo design of sequences for nucleic acid structural engineering. *Journal of Biomolecular Structure & Dynamics*, 8(3):573–581, 1990

[52] Seeman, N.C. DNA in a material world. *Nature* **421**: 427-431 (23 January 2003)

[53] Simmel, F.C., Dittmer, W.U. DNA Nanodevices. *Small*, **1**:284–299 (2005).

[54] Shor, P. W. (1997), Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, *SIAM J. Comput*. **26** (5): 1484–1509.

[55] Stojanovic, M. N. & Stefanovic, D. A deoxyribozyme-based molecular automaton. *Nature Biotechnol*. **21**, 1069–1074 (2003).

[56] Su, X. and Smith, L. M. Demonstration of a universal surface DNA computer *Nucleic Acids Res.* 4 June 2004: 3115-3123.

[56b] Tanaka, F., Kameda, A., Yamamoto,, M., Ohuchi A. Design of nucleic acid sequences for DNA computing based on a thermodynamic approach *Nucleic Acids Res.* **33**(3): 903–911. 8 February 2005

[57] Watson JD, Crick FH (April 1953). Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid. *Nature* **171** (4356): 737–738

[58] Weaver, R. F. *Molecular Biology*. Boston: McGraw-Hill, 2008.

[59] Wang. H. Dominoes and the AEA case of the decision problem. In Proc. Symp. Math. *Theory of Automata*, p. 23–55, New York, 1963. Polytechnic Press.

[60] Wang, H. Proving Theorems by Pattern Recognition, II. *Betl System Tech. J.* 40 (1961): 1-41.

[61] Winfree, E. , Liu, F. R. , Wenzler, L. A. & Seeman, N. C. Design and self-assembly of two-dimensional DNA crystals. *Nature* **394**, 539–544 (1998)

[62] Winfree, E. Simulations of computing by self-assembly. Technical report CS-TR:1998.22. Pasadena, California: California Institute of Technology (1998).

[63] Winfree, E. Algorithmic Self-Assembly of DNA. PhD Thesis, Caltech (1998).

[64] Wolfe-Simon, F. *et. al*. A Bacterium That Can Grow by Using Arsenic Instead of Phosphorus. *Science* **1197258** Published online 2 December 2010.

[65] Xiao, S., Liu, F., Rosen, A., Hainfeld, J., Seeman, N. C., Musier-Forsynth, K. M. & Kiehl, R. A. Selfassembly of metallic nanoparticle arrays by DNA scaffolding. (2002*) Journal of Nanoparticle Res*. 4, 313–317.

Appendix A

A.1 Stapling Trial Experiment:

Lane 01: 91-bp

TCACTCCACTTAACTATAACCACAAACTCATTACACCAATTCTCTTCTCCTACAATTCCTAATGTCCAACATACTCTCATCCTCTAACATA

Lane 02: 151-bp

CTATCCAATAACCTCTTCATACACTTACACTATTCTCACCCATAAACACCTAACTAGACTACACTATCAGCATCACTACCCTATTCTACTAACTTC
ACCCCTATATTTCTTCCATCACATAACTCATACATACTCACTATTTATATCCACC

Hap01	TCACTCCACTTAACTATAACCACAAACTCA	30 bp
Hap02	TTACACCAATTCTCTTCTCCTACAATTCCTA	31
Hap03	ATGTCCAACATACTCTCATCCTCTAACATA	30
Hap04	CTATCCAATAACCTCTTCATACACTTACAC	30
Hap05	TATTCTCACCCATAAACACCTAACTAGACT	30
Hap06	ACACTATCAGCATCACTACCCTATTCTACT	30
Hap07	AACTTCACCCCTATATTTCTTCCATCACAT	30
Hap08	AACTCATACATACTCACTATTTATATCCACC	31
Staple01	ATTGGTGTAATGAGTTTGTG	20
Staple02	TGTTGGACATTAGGAATTGT	20
Staple03	GGTGAGAATAGTGTAAGTGT	20
Staple04	CTGATAGTGTAGTCTAGTTA	20
Staple05	GGGTGAAGTTAGTAGAATAG	20
Staple06	TGTATGAGTTATGTGATGGA	20
BluePrimer01	TCACTCCACTTAACTATAAC	20
BluePrimer02	TATGTTAGAGGATGAGAGTA	20
RedPrimer01	CTATCCAATAACCTCTTCAT	20
RedPrimer02	GGTGGATATAAATAGTGAGTA	21

The codenames are given as IDs to species to help run the experiment. “Haps” are strands being stapled. The PCR primers above are used in PCR reactions.

Sequencing results:

The two lanes (lane 01 and lane 02) are cloned into a plasmid and sent for sequencing. The sequencing results below (McGill University-Genome Quebec, Montreal) show the formation of the two lanes. The sequence of lane 01 and 02 are highlighted in blue and red, respectively; while their respective reverse complement are highlighted with dimmed red and blue.

Lane01
Lane02
ReverseComplementLane01
ReverseComplementLane02

>ID|7823763 L22_P1002642_033.ab1

TGCATGGAACGCCGCAGACTTCGGATGCTCGAGTTTTTCGCAGATGGTGGATATAAATAGTGAGTATGTATGAGTTATGTGATGGAAGAAAT
ATAGGGGTGAAGTTAGTAGAATAGGGTAGTGATGCTGATAGTGTAGTCTAGTTAGGTGAGAATAGTGTAAGTGTATGAAGAGGTTATTGGA
TAGGGTGGTGGATATAAATAGTGAGTATGTATGAGTTATGTGATGGAAGAAATATAGGGGTGAAGTTAGTATAATAGGGTAGCGATGCTGA
TCCTGTAATCCTACTACGCGTCTCTGGGTGACAATCGCTGTCGCGTCCGACGAGCTCTCGGATACCCCTTTCTCACATATCCCCTACCATACTCT
CTGTCGGCCCCGCCACTCCCCCCTCTCCCTCCCCACTCTCCCACGCACCCCAGCCCCCCCTATACTAGCCTATCCTCAAATCCTTCCGCCTCCCCC
ACCCCCCACACCACCCCCTCCCTCGCTGCTCGCCCTCTCTCCTGCCCTTCCCCCTCTCCTCGCCTCCCCTCCCCCCCTCCCCATCCCCCCCTCCCCC
ACCCCTACCACCTCTTTCCCCTCCCGTCCCCCCCTCACCCCCTCCTCCCCCCCCCTGACCTCCCCCTCCCCTCCCCCTCCCCCTCCCACCTCCTCCCCC
CCGACACTCCTCCTCCTACCCCCCGCCCTCCCTCCCCACTCCCCACCCCCCCTCCCCGCCTCTCCTCCCCCCCCCCCACTCTCCTCACCACTACATCC
CCCCCCCTCCCCTTCCTCCCTCCACTCCTCATCCCCCTCCCCCCCCACCCCCCTCCCCATCCCCCCTCTCCCTCCTCTCCCCCTTCCCCTACCTCCCGC
CCCCCCCTCCCCCCCCCCTCCCCCCCCCTCCCCCCTCCCCCCCCCCTCCCCCCCCTCCCTCCCTCTCCCCGTCCCTCTTCCCCCCCCGCTCCCCCTGCCCT
CCTCCTCTCCCCTCCCTCACCCCCTCCCCGCCCCCAACTCTTTCCCCTCGCATCTCTTCCCCCCCCTCCCCCTCTCCCCACTCCCCCCCCCCCCCCCC
CTCTCCCCCCTTCTGCGCTCCCCCGCGCCCACCCCCTCCCCTCCCCAGCCCCCCCCCCCCCCCTCTCCTCCCTCCTCCCCCCCTCCCCTCCCTCCTTTC
CCCCCCCCTCCTCTCCCCCCCCACCCCTCTGCCCTCCCCCTCCCCCCTCTCCCCCTTCCCCTCCCTTCATCCCCTCCTCCCCCCTTCCCCTCCTCCCTC
CCCCTTCTC

>ID|7823757 L21_P1002642_018.ab1

TTTCTAGTTATTCGCCTGAATCTTGAGAGAATAAAGAAGACATCGATTTTCCATGGCAGCTGAGAATATTGTAGGAGATCTTCTAGAAAGATG
GTGGATATAAATAGTGAGTATGTATGAGTTATGTGATGGAAGAAATATAGGGGTGAAGTTAGTAGAATAGGGTAGTGATGCTGATAGTGTA
GTCTAGTTAGGTGTTTATGGGTGAGAATAGTGTAAGTGTATGAAGAGGTTATTGGATAGATCTTGCTGAAAAACTCGAGCCATCCGGAAGAT
CTGGCGGCCGCTCTCCCTATAGTGAGTCGTATTACGCCGGATGGATATGGTGTTCAGGCACAAGTGTTAAAGCAGTTGATTTTATTCACTATG
ATGAAAAAAACAATGAATGGAACCTGCTCCAAGTTAAAAATAGAGATAATACCGAAAACTCATCGAGTAGTAAGATTAGAGATAATACAACA
ATAAAAAAATGGTTTAGAACTTACTCACAGCGTGATGCTACTAATTGGGACAATTTTCCAGATGAAGTATCATCTAAGAATTTAAATGAAGAA
GACTTCAGAGCTTTTGTTAAAAATTATTTGGCAAAAATAATATAATTCGGCTGCAGGGGCGGCCTCGTGATACGCCTATTTTTATAGGTTAATG
TCATGATAATAATGGTTTCTTAGACGTCAGGTGGCACTTTTCGGGGAAATGTGCGCGGAACCCCTATTTGTTTATTTTTCTAAATACATTCAAA
TATGTATCCCCCTCATGAGACAATAACCCTGATAAATGCTTCCATAATATTGAAAAAAGAACAGTATGATTATTCAACATTTCCGTGTCCCCCTT
TATCCCTTTTTTTGCCGCATTTTGCCTTTCCTGTTTTTGCTTCCCCCAGAAACCCCGGTTTAAAGTAAAAAGATGCCTGAAGATTCAGTTTGGGT
GCCCCAAGTGGGTTTACATCCAAACTGGGTTCTTCACCAACGGGGAAAGATTCCTTGAAGAGTTTTTCCCCCCCCGAAAAAACGTTTTTCTCAA
TGATTGAACCCCTGTTTTAAAGGTCCTTCCCTTTTTGGCGCCCCCGGAAATTATTCCTCGCTTATTGCACCCCCGGGTCCAAAAAACCCATTTCG
GGTTCCCCCCCGATTACACCTAATTTCTTCTAAAAAATACACCTGGCCTGTAAACCACCCTCACCC

>ID|7818507 L12_P1002632_091.ab1

GCATGGCGAGGGCGCAGATCTTCCGGATGGCTCGAGTTTTTCGCAAGATTCACTCCACTTAACTATAACCACAAACTCATTACACCAATTCTCT
TCTCCTACAATTCCTAATGTCCAACATACTCTCATCCTCTAACATAATCTTTCTAGAAGATCTCCTACAATATTCTCAGCTGCCATGGAAAATCGA
TGTTCTTCTTTTATTCTCTCAAGATTTTCAGGCTGTATATTAAAACTTATATTAAGAACTATGCTAACCACCTCATCAGGAACCGTTGTAGGTGG
CGTGGGTTTTCTTGGCAATCGACTCTCATGAAAACTACGAGCTAAATATTCAATATGTTCCTCTTGACCAACTTTATTCTGCATTTTTTTTGAAC
GAGGTTTAGAGCAAGCTTCAGGAAACTGAGACAGGAATTTTATTAAAAATTTAAATTTTGAAGAAAGTTCAGGGTTAATAGCATCCATTTTTT
GCTTTGCAAGTTCCTCAGCATTCTTAACAAAAGACGTCTCTTTTGACATGTTTAAAGTTTAAACCTCCTGTGTGAAATTATTATCCGCTCATAAT
TCCACACATTATACGAGCCGGAAGCATAAAGTGTAAAGCCTGGGGTGCCTAATGAGTGAGCTAACTCACATTAATTGCGTTGCGCTCACTGCC
AATTGCTTTCCAGTCGGGAAACCTGTCGTGCCAGCTGCATTAATGAATCGGCCAACGCGCGGGGAGAGGCGGTTTGCGTATTGGGCGCTCTT
CCGCTTCCTCGCTCACTGACTCGCTGCGCTCGGTCGTTCGGCTGCGGCGAGCGGTATCAGCTCACTCAAAGGCGGTAATACGGTTATCCACAG
AATCAGGGGATAACGCAGGAAAGAACATGTGAGCAAAAGGCCAGCAAAAAGCCAGGAACCGTAAAAAAGCCGCGTTGCTGGCGTTTTTCCA
TAGGCTCCGCCCCCCTGACGAGCATCACAAAAATCGACGCTCAAGTCAGAGGTGGCGAAACCCGACGGGACTATAAAGA

>ID|7818467 L21_P1002632_010.ab1

TTCTACTTATTCGCCTGAAACTTGAGAGATAAAGAAGACATCGATTTTCCTGGCAGCTGAGAATATTGTAGGAGATCTTCTAGAAAGATGGTG
GATATAAATAGTGAGTATGTATGAGTTATGTGATGGAAGAAATATAGGGGTGAAGTTAGTAGAATAGGGTAGTGATGCTGATAGTGTAGTC
TAGTTAGGTGTTTATGGGTGAGAATAGCGTTAGTGTATGTAAAGGTTACTCGAATACATCTTGCTCATCCTTAGATCCCACCTCCAGGATCTGG
CTGACGCTCTCCCCATGTCTAATCGGCCCACCCCCTACGGATATGGTGTCCAGGCGCCATTGTCAAATCAAATCCTTTTACTCCCTATGCTAACT
AAAACCGTGACTGGAACCTGTCTCCCCTCACATCTCCAGATAATCCTTAACAGACATCCAGCAGCAAGAGTTACCGACGATCCACTCACTACCA
CCTCGGCTTCCAACTTTTCTCCCCCTCGTGAACCTACTCCTGAGACCACAAACTCCTCTACCCAATCATCCTCTCAGGACCTACCTCTCACTACAC
TCTCCGCACCTCCCGTACCCAGCTTATTTCGCGCCCAGCCTCCTACTAACTCGACTCTTCTTCCTAGCGCTCTCCCGCTACCCCCCCAATCATTAT
CCACTCTTCCCTTCCTGCACACTCACTACTGTTTCCACCTACGCTCCGGCCCCATACCCTTTTCGCGTACACCTCTGCACCCGCACCCCTTCAACC
TCTCTTCTATCTTCTCCCCCTTTACCCTTTCCTACTCGTCCCTCTCCCTGCCTTCCCACCCGCAATCCCCCCTCCATCAAACTCGCTCCCACTCTTAC
CTCCCCCCATCCCCTCTCACTCTCCCAATAGCCTCACCCCCTTTTCCTGCACCCCCCCCCATAATTCCTTATAACCATGCCATACCCCTCACCCCCTT
CCCCCGTCTCCTACACTCCTCCTCTTCTCCCCGCTCCCATGCTTATCCTCAACAACCACTCCCCTTCTCCTCCTCCCCTCCACTCTCACTCCATACTC
CTACACTATCTCCCACCTTCCCGCCTCATTTCACTCACTTCCAGTTCCATCAACCCCGGCCCTCACTTCTCTACCACCCCACTATAACTAAGCTCA
CTCACCCACATCTACTTCCACCTGTATCTGCCCGCTTCCCACTCCATCTCTCGCACAAGCTCCTCACCACGTTTACTCCCCCTCTCAGC

>ID|7818506 L11_P1002632_076.ab1

AACTAGTTATTCGCCTGAATCTTGAGAGATAAAGAAGACATCGATTTTCCATGGCAGCTGAGAATATTGTAGGAGATCTTCTAGAAAGATTCA
CTCCACTTAACTATAACCACAAACTCATTACACCAATTCTCTTCTCCTACAATTCCTAATGTCCAACATACTCTCATCCTCTAACATAATCTTGCTG
AAAAACTCGAGCCATCCGGAAGATCTGGCGGCCGCTCTCCCTATAGTGAGTCGTATTACGCCGGATGGATATGGTGTTCAGGCACAAGTGTT
AAAGCAGTTGATTTTATTCACTATGATGAAAAAAACAATGAATGGAACCTGCTCCAAGTTAAAAATAGAGATAATACCGAAAACTCATCGAGT
AGTAAGATTAGAGATAATACAACAATAAAAAAATGGTTTAGAACTTACTCACAGCGTGATGCTACTAATTGGGACAATTTTCCAGATGAAGTA
TCATCTAAGAATTTAAATGAAGAAGACTTCAGAGCTTTTGTTAAAAATTATTTGGCAAAAATAATATAATTCGGCTGCAGGGGCGGCCTCGTG
ATACGCCTATTTTTATAGGTTAATGTCATGATAATAATGGTTTCTTAGACGTCAGGTGGCACTTTTCGGGGAAATGTGCGCGGAACCCCTATTT
GTTTATTTTTCTAAATACATTCAAATATGTATCCGCTCATGAGACAATAACCCTGATAAATGCTTCAATAATATTGAAAAAGGAAGAGTATGAG
TATTCAACATTTCCGTGTCGCCCTTATTCCCTTTTTTGCGGCATTTTGCCTTCCTGTTTTTGCTCACCCAGAAACGCTGGTGAAAGTAAAAGATG
CTGAAAATCAGTTGGGTGCACGAGTGGGTTACATCGAACTGGATCTCAACAGCGGTAAGATCCTTGAAAAGTTTCGCCCCGAAAAAAGTTTT
CCAATGAAGAACACTTTTAAAGTTTTGCTATGTGGCGCGGTTATTATCCCCGTAATGGACCCCGGGCAAAAACCAAACTCGGTCCGCCGCAAA
AACCTATTTT

>ID|7818513 L21_P1002632_009.ab1

TTATTGGGGGCGGCCGCCGATCTTCCGGATGGCTCGAGTTTTTCGCAAGATCTATCCAATAACCTCTTCATACACTTACACTATTCTCACCCATA
AACACCTAACTAGACTACACTATCAGCATCACTACCCTATTCTACTAACTTCACCCCTATATTTCTTCCATCACATAACTCATACATACTCACTATT
TATATCCACCATCTTTCTAGAAGATCTCCTACAATATTCTCAGCTGCCATGGAAAATCGATGTTCTTCTTTTATTCTCTCAAGATTTTCAGGCTGT
ATATTAAAACTTATATTAAGAACTATGCTAACCACCTCATCAGGAACCGTTGTAGGTGGCGTGGGTTTTCTTGGCAATCGACTCTCATGAAAAC
TACGAGCTAAATATTCAATATGTTCCTCTTGACCAACTTTATTCTGCATTTTTTTTGAACGAGGTTTAGAGCAAGCTTCAGGAAACTGAGACAG
GAATTTTATTAAAAATTTAAATTTTGAAGAAAGTTCAGGGTTAATAGCATCCATTTTTTGCTTTGCAAGTTCCTCAGCATTCTTAACAAAAGACG
TCTCTTTTGACATGTTTAAAGTTTAAACCTCCTGTGTGAAATTATTATCCGCTCATAATTCCACACATTATACGAGCCGGAAGCATAAAGTGTAA
AGCCTGGGGTGCCTAATGAGTGAGCTAACTCACATTAATTGCGTTGCGCTCACTGCCAATTGCTTTCCAGTCGGGAAACCTGTCGTGCCAGCT
GCATTAATGAATCGGCCAACGCGCGGGGAGAGGCGGTTTGCGTATTGGGGGCTCTTCCGCTTCCTCGCTCACTGACTCGCTGCGCTCGGTCG
TTCCGGTGGGGGCGAGCGGTATCAGCTCACTCCAAAGGCGGGAAATACGGGTTATCCCACAGAATCAGGGGGAAAAACGCAGGAAAGAAA
CATGTGAAGCCAAAAGGGCCAGCAAAAAGGGCCAGGAAAACCCGTAAAAAAGGCCCGCCGTTTGCCTTGGCGGTTTTTTTCCCAAAGGGGC
TCCGGGCCCCCCCCTGGAACGAAGCCATTCCACCAAAAAAATCTCGACGCG

>ID|7818468 L22_P1002632_025.ab1

TTCCCATGGAAAACGGCCGCCCGAATCTCTCCGGATGGCCTCAGTTTTTCCAGCCAAGATGGTGGATATAAATAGTGCAGTATGCTATTGAGT
TATGCTCGATGCGAAGCAAATATCAGGCGGCTCGAAGCTTAGCTACGCAATACCGGCTAGCTCGATCGCCTGCACTAGTGCTACCTCCCAGCT
CTCGGCTGCACAACTACCTGCTCACCTGCTCCTGCAACACGGCCTATCTCCCCACCCGCCCCCCGCGCCACCCCCCGCCCCCCCCACCCTCCTCC
CCCCCCTCCCCCCTCCATCACCTCCCCCCCCCCCCCCCCTCCCCCCCCTCCCCACCCCCTCCCCGCCCCCCCCTCCCCCCCCCCCCCACCCCCCCTTC
CCTCCCCTCCCTCTTCCCCCCCCCCCCCCCCCCCCCCCTCTCCCCCCCCCTCCCCCTCCCCCCCCCCCCCCACCCTCCCCCCCCCCCCCCCCCTCCCTCT
CCCCCCCCTCTCCCCCCCCTCCCCCCCCCCCTCCCCCCCCCCCCCCTCTCCCCCTCCTCCTCTCCCCCCCCCCCCCCCCGCCACCCCCCCCCCCCCCCTTC
ACACTCACCCTCCCCCCCCCTCCCCCACCCCCACCCCCCCCCCCTCCCCCGCCCCCCCCCCCCCACACCCCCCCCCACCCCCCTTCCCCCCCCCCCCCC
CCCCCTCCTCCCCCTCCCCCCCCCCTCTCCCCCCCCCCCCCCCCCCCCCCCTCCCCCCCTCCCCTCCCCCCCCCCCCCCTCTCCCCTCCCGCTCTTCCT
CCCTCCCCCTCCCCTCCTCCCCCCCTCTCCTCTCCTCTCCCCCTCCCCCCCCCTCCCCCGCCTCCTTCCCCCCCCCCTCCCTCTCCCCCCCCCCCTCC
CCCCTCCCCTCCCCCCCCTCTCCCCCCCTCCACCCCTCTCCCCCACCCCCCTCCCTCCCCCTCCCTTCCCCCCCCCTCCTCCTCCCCCCCCCCCCTCC
CTCCCCTCCCTCCCACCCCTCTCCCCCTCGCCCCCCCCCCCCCACCTTCCTCCCCCCCCACCCCCCCCCTCCCCCCCCCTCGCCCCCCCCTCTCCCAC
CCCCCCTCTCCCCTCCCCCCCTCCCACCCCCCTCCTCTCCTCCCTCCCCCCTCCCCACCACCCACCCTCCCCCTCCCTCTCCCCGCCCCCCTCTTCCC
CTCCCCCCTACCCCTCCCCCCCCTCCTCCTCTCACCCCCCTCCCCCCCCTCTCTCCCTCCCCCTCTGTCTACCTCCTCCCTCACCTCCCTCCCCCTCCT
CTTCCTCCTCCTCCCCTACTCTCCCTCACTCCCTCCCCCGTCCCCCCCTCCCTCCCCCTCACCTCTCCCCTCCCGCCCCTCCCCTCCTCCCTCTCCCC
CCACCTCCACCCTCCTCTCCGCCCCTCCCTTCTCCTCCTCCCCCCCCCACTCCCCCCACCCCCCCCCTTCCCCTCCTCCCCCACCTCTCCCCCCCCCCC
TCCCTCCCTCCCCCCTCTCCCTGCCCCCCGCCCACCTTCACCTTCTCCCGCCACCCCCTCCTCCTCCCACCCTCCCTCGCCTCTCCCCCTCCCCACTC
CCCCTCACACCACTCCCCCCTCCCCACCTCCTCCGCCCTC

>ID|7818511 L11_P1002632_075.ab1

GGCTAGGGACGGCGCAGATCTTCGGATGGCTCGAGTTTTTCAGCAAGATTATGTTAGAGGATGAGAGTATGTTGGACATTAGGAATTGTAG
GAGAAGAGAATTGGTGTAATGAGTTTGTGGTTATAGTTAAGTGGAGTGAATCTTTCTAGAAGATCTCCTACAATATTCTCAGCTGCCATGGAA
AATCGATGTTCTTCTTTTATTCTCTCAAGATTTTCAGGCTGTATATTAAAACTTATATTAAGAACTATGCTAACCACCTCATCAGGAACCGTTGT
AGGTGGCGTGGGTTTTCTTGGCAATCGACTCTCATGAAAACTACGAGCTAAATATTCAATATGTTCCTCTTGACCAACTTTATTCTGCATTTTTT
TTGAACGAGGTTTAGAGCAAGCTTCAGGAAACTGAGACAGGAATTTTATTAAAAATTTAAATTTTGAAGAAAGTTCAGGGTTAATAGCATCCA
TTTTTTGCTTTGCAAGTTCCTCAGCATTCTTAACAAAAGACGTCTCTTTTGACATGTTTAAAGTTTAAACCTCCTGTGTGAAATTATTATCCGCTC
ATAATTCCACACATTATACGAGCCGGAAGCATAAAGTGTAAAGCCTGGGGTGCCTAATGAGTGAGCTAACTCACATTAATTGCGTTGCGCTCA
CTGCCAATTGCTTTCCAGTCGGGAAACCTGTCGTGCCAGCTGCATTAATGAATCGGCCAACGCGCGGGGAGAGGCGGTTTGCGTATTGGGCG
CTCTTCCGCTTCCTCGCTCACTGACTCGCTGCGCTCGGTCGTTCGGCTTGCGGCGAACGGTAACAGCTCACTTCAAAGGCGGAAATAACGGTT
ATCTCACACAATCAGGGGAATAACGCCAGGAAAAAAACCTTGTGAACCAAAGGGCCTTGCAATAAGGGCCAGGAAACCCGACAAAAAGGGC
CCCCCTTTGCTTGACTGCTTTTTTCCCATTAGGCCTCCCGCCCCCCCCCTGAATCCAGCCCACCCCCAAAACAAATCCTGGCCCCTTCTAACATTC
AAAAAGGTTGGGGCGAATAACCCCCCGATCCGGGGAACCTAATTAAACAGAATTCCCTCCTGGGCCCGTTTTATCCTCCCCCCTGTGAA

>ID|7818512 L12_P1002632_092.ab1

AACTGTTATTCGCCTGAATCTTGAAGATAAAGAAGACATCGATTTTCCTGGCAGCTGAGAATATTGTAGGAGATCTTCTAGAAAGATTATGTT
AGAGGATGAGAGTATGTTGGACATTAGGAATTGTAGGAGAAGAGAATTGGTGTAATGAGTTTGTGGTTATAGTTAAGTGGAGTGAATCTTG
CTGAAAAACTCGAGCCATCCGGAAGATCTGGCGGCCGCTCTCCCTATAGTGAGTCGTATTACGCCGGATGGATATGGTGTTCAGGCACAAGT
GTTAAAGCAGTTGATTTTATTCACTATGATGAAAAAAACAATGAATGGAACCTGCTCCAAGTTAAAAATAGAGATAATACCGAAAACTCATCG
AGTAGTAAGATTAGAGATAATACAACAATAAAAAAATGGTTTAGAACTTACTCACAGCGTGATGCTACTAATTGGGACAATTTTCCAGATGAA
GTATCATCTAAGAATTTAAATGAAGAAGACTTCAGAGCTTTTGTTAAAAATTATTTGGCAAAAATAATATAATTCGGCTGCAGGGGCGGCCTC
GTGATACGCCTATTTTTATAGGTTAATGTCATGATAATAATGGTTTCTTAGACGTCAGGTGGCACTTTTCGGGGAAATGTGCGCGGAACCCCT
ATTTGTTTATTTTTCTAAATACATTCAAATATGTATCCGCTCATGAGACAATAACCCTGATAAATGCTTCAATAATATTGAAAAAGGAAGAGTAT
GAGTATTCAACATTTCCGTGTCGCCCTTATTCCCTTTTTTGCGGCATTTTGCCTTCCTGTTTTTGCTCACCCAGAAACGCTGGTGAAAGTAAAAG
ATGCTGAAGATCAGTTGGGTGCACGAGTGGGTTACATCGAACTGGATCTCAACAGCGGGTAAGATCCTTGAGAGTTTTCCCCCCCAAAGAAC
GTTTTTCCAATGATGACCACTTTTAAAAGTTCTTGCTATGTGGGCCCGGGATTTATCCCCCTAATTGACCCCC

>ID|7818476 L22_P1002632_026.ab1

CCTATGTTATTCGCCTGAATCTTGAGAGAATAAAAGAAGACATCGATTTTCCATGGCAGCTGAGAATATTGTAGGAGATCTTCTAGAAAGATC
TATCCAATAACCTCTTCATACACTTACACTATTCTCACCCATAAACACCTAACTAGACTACACTATCAGCATCACTACCCTATTCTACTAACTTCA
CCCCTATATTTCTTCCATCACATAACTCATACATACTCACTATTTATATCCACCACCCTATCCAATAACCTCTTCATACACTTACACTATTCTCACC
TAACTAGACTACACTATCAGCATCACTACCCTATTCTACTAACTTCACCCCTATATTTCTTCCATCACATAACTCATACATACTCACTATTTATATC
CACCATCTTGCTGAAAAACTCGAGCCATCCGGAAGATCTGGCGGCCGCTCTCCCTATAGTGAGTCGTATTACGCCGGATGGATATGGTGTTCA
GGCACAAGTGTTAAAGCAGTTGATTTTATTCACTATGATGAAAAAAACAATGAATGGAACCTGCTCCAAGTTAAAAATAGAGATAATACCGA
AAACTCATCGAGTAGTAAGATTAGAGATAATACAACAATAAAAAAATGGTTTAGAACTTACTCACAGCGTGATGCTACTAATTGGGACAATTT
TCCAGATGAAGTATCATCTAAGAATTTAAATGAAGAAGACTTCAGAGCTTTTGTTAAAAATTATTTGGCAAAAATAATATAATTCGGCTGCAG
GGGCGGCCTCGTGATACGCCTATTTTATAGGTTAATGTCATGATAATAAGGGTTTCTTAGACGTCAGGTGGCACTTTTCGGGGAAATGTGCGC
GGAACCCCTATTTGTTTATTTTTCTAAAATCATTCAAATATGGTTTCCGCTCATGAGAACATAACCCTGAATAAAGGGTTCCAATAATTTTGGAA
AAAGGGAAGAAGTAGGAGTATTTCCACATTTCCCGGGTCGCCCTTTATTCCCCTTTTTTTGGCGGGAATTTTGGCCCTTCCGGGTTTTTGGCTC
CACCCCAAAAACGCTTGGGGTAAAAAGTAAAAGAAAGGCTTAAAAAATCAAGTTGGGGGGGGACT

A.2 Bridging Trial Experiment:

oligo	role in assembly
3-TCATCTTATCCCATCACTACGACTATCACA-5	Lane 1
5-TCACTCCACTTAACTATAACCACAAACTCA-3 5- ATGTCCAACATACTCTCATCCTCTAACATA -3	Lane 2
3- CCACCTATATTTATCACTCATACATACTCAA-5 3- CACATTCACATACTTCTCCAATAACCTATC-5	Lane 3
5-TTGGACATTGAGTTTG-3	a filler oligo on Lane2
5- TATGAGTTGTGTAAGT -3	a filler oligo on Lane3
5- ATAAATAGGTTAAGTG-3	together bridge Lane2 and Lane3
5-TGGTTATAGAGTATGT-3	
5- GTATGAAGAGAGTATG-3	together bridge Lane2 and Lane3
5-AGAGGATGAGGTTATT-3	
5-TAGGGTAGTGAGTTTG-3	together bridge Lane1 and Lane2
5- TTGGACATTGATGCTG -3	

Assembly of <u>two</u> lanes:

Lane2 (shown green in Chapter 5) 5-TCACTCCACTTAACTATAACCACAAACTCA|ATGTCCAACATACTCTCATCCTCTAACATA -3

3-GTGAATTGATATTGGT-5 3-GTTTGAGTTACAGGTT-5 3-GTATGAGAGTAGGAGA-5

5- ATAAATAGGAGTATGT -3 5-TATGAGTTGTGTAAGT-35-GTATGAAGAGGTTATT-3

Lane3 (shown red in Chapter 5) 3- CCACCTATATTTATCACTCATACATACTCAA|CACATTCACATACTTCTCCAATAACCTATC-5

Assembly of <u>three</u> lanes:

Lane1 (shown blue in Chapter 5) 3-TCATCTTATCCCATCACTACGACTATCACA-5

5-TAGGGTAGTGATGCTG-3

3-GTTTGAGT TACAGGTT-5

Lane2 5-TCACTCCACTTAACTATAACCACAAACTCAATGTCCAACATACTCTCATCCTCTAACATA -3

3-GTGAATTGATATTGGT-5 3-GTATGAGAGTAGGAGA-5

5- ATAAATAGGAGTATGT -3 5-TATGAGTTGTGTAAGT-35-GTATGAAGAGGTTATT-3

Lane3 3- CCACCTATATTTATCACTCATACATACTCAACACATTCACATACTTCTCCAATAACCTATC-5