Constructing Problems of Geometric Combinatorics

Daniel S. Hirschberg

Dept. of Computer Science
Univ. of California, Irvine
Irvine, CA 92697-3435 USA
dan@ics.uci.edu

Abstract

Global information is used to simplify and speed the construction of a geometric problem and ensure that it has unique solution. The combinatorial nature of this problem would typically lead to use of a backtracking procedure in its solution.

1 Introduction

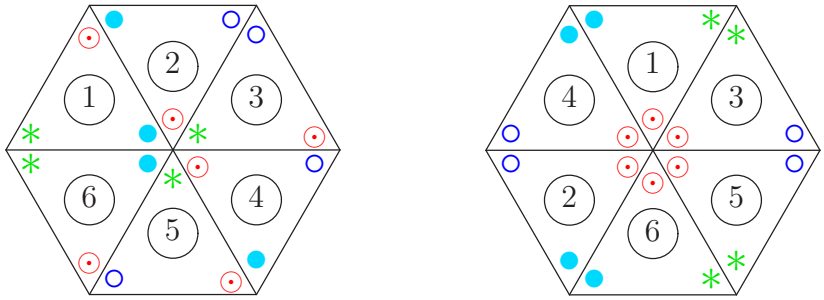
We develop an approach that can be used to more efficiently construct the following geometric problem and others like it. The solution to the problem is a *solution hexagon*, which is a hexagon having side dimension n and that can be viewed as being the result of gluing together $6n^2$ unit-edged equilateral triangles. The *nodes* of the solution hexagon are defined to be those locations on or within the hexagon at which the vertices of the unit-edged triangles are located. Each of the nodes is colored with one of a set of k colors. Each triangle vertex has the color of the node at which it is located. Accordingly, all triangle vertices that are associated with the same node will have the same color.

The solution hexagon is partitioned into pieces by cutting along some of the triangle edges. Therefore, each piece consists of one or more triangles that are glued together along their edges. The resulting set of puzzle pieces are presented as a *hexagon problem* by assembling them into a hexagon that differs from the original solution hexagon. In particular, at least one (and possibly even all) of the nodes in the hexagon problem will have associated triangle vertices which do not all have the same color.

It is required that there is only one hexagon solution into which the pieces of the hexagon problem can be reassembled. Also, it is desirable that the problem be relatively hard to solve using a straightforward approach, even though it may contain relatively few pieces.

Figure 1 is an example of a hexagon problem and its solution, for $n = 1$ and $k = 4$. For ease of reference, we use a different symbol for each of the colors: ⊙, ○, ∗, ●.

Figure 1: Example problem/solution



2 Related Problems

In the traditional jigsaw problem, puzzle pieces have four sides, each of which has a male, female, or neutral edge (usually edges on the puzzle border). The male edges are distinct in shape, and each has a mating female counterpart. When correctly put together, the top of the ensemble of puzzle pieces typically displays a picture.

If there were a simple way to index the male and female shapes, the puzzle solution could be rapidly obtained by evaluating the index of each puzzle male edge and then, for each female edge, evaluating its index and attaching it to its mate. This approach was used decades ago [1, 2], but it is difficult to quickly determine with assurance that two scanned pieces are really mates. Not having the ability to index shapes, a common initial approach is to segregate the pieces by their pictorial content or color and also by their having border edges, which constitute a small fraction of the set of puzzle pieces. The global approach of border segregation has recently been used to aid in automatic solution of apictorial jigsaw puzzles [3, 4].

In pure packing puzzles, each puzzle piece typically has only straight edges and its top does not have part of a big picture. Often, the piece shapes are from a small set (sometimes singleton) of allowable shapes. The problem is to place the pieces so that the ensemble fits in a desired outline.

Edge-mating puzzles add a constraint to the pure packing puzzle. There is a small picture or design that straddles each edge common to adjoining pieces. There are only a very few (perhaps only one) distinct such designs. In contrast to jigsaw puzzles, the problem is not of finding the one possible mate for each edge but, rather, of finding the correct mate from the many feasible matching candidates so that all mating requirements can simultaneously be satisfied while the ensemble fits in a desired outline. The term "edge-matching" is often used to describe edge-mating, but sometimes alludes to the problem in which the mate of a partial design is an exact replica of that partial design.

Here, we concern ourselves with vertex-matching. In general, all of these problems are NP-complete [5].

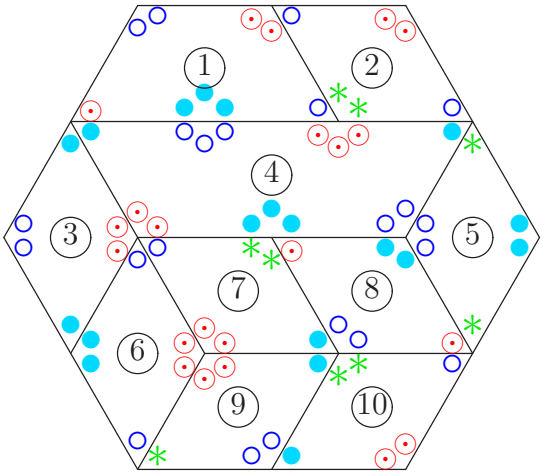
3 Global Information

To illustrate its construction, we consider a somewhat larger hexagon problem. Figure 2 shows a hexagon problem (with $n = 2$ and $k = 4$) having ten pieces, where piece 4 is a large trapezoid (consisting of five triangles), piece 1 is a small trapezoid (consisting of three triangles), and each of the other pieces is a rhombus (consisting of two triangles).

We are interested in creating hexagon problems that have unique solutions and that are not easily solved. Given such a problem, it is possible to find its solution by using local matching information. This straightforward approach may be efficient when there are very few vertices having particular colors to enable rapid vertex matches or pairs of colors that occur on ends of very few edges to enable rapid edge matches. Otherwise, we expect that relying on solely the application of local matching information will require a process that involves backtracking, which typically takes exponential-time.

Creating such a problem with guaranteed unique solution is not straightforward. However, the use of global information can greatly simplify the task of creating the problem, and guarantee solution uniqueness during the creation of the problem.

Figure 2: Hexagon problem with $n = 2$



We can add a requirement that some or all unit-distant symbols differ. In constructing this problem, there are few positions of the combinatorial set of possibilities consistent with that requirement. We can reduce the combinatorial set of possibilities by applying the unit-distant requirement conjunctively to different symbols.

In general, a hexagon contains $6n$ nodes on its perimeter, 6 of which correspond to two triangular vertices each, and $6n-6$ of which correspond to three triangular vertices each. The remaining nodes each correspond to six triangular vertices. There is a total of $18n^2$ triangular vertices, and so the hexagon has $3n^2 - 3n + 1$ internal nodes, and thus $3n^2 + 3n + 1$ nodes altogether.

4 Constraining Patterns

We begin our construction of a uniquely solvable problem by analyzing the distribution of symbols.

For our problem, $n = 2$, and so there are 72 vertices among 19 nodes, consisting of 6 corner nodes (each has 2 vertices), 6 midside nodes (each has 3 vertices), and 7 internal nodes (each has 6 vertices).

Our illustrative problem has the following symbol frequency: 22 ⊙, 22 ○, 19 ●, and 9 ✳.

Because a symbol X that appears in a node cannot be in a second node located a unit distance away, and because there are only seven internal nodes – six nodes in a hexagonal pattern surrounding a central node – any symbol X can be in at most 3 internal nodes.

If symbol X is in exactly three internal nodes (each corresponding to six vertices, i.e., instances of X) then three of the corner nodes and all of the midside nodes are eliminated from containing X, and there can be at most a total of 24 instances of X. This limit of 24 can be achieved in one way, plus its rotations, as shown in Figure 3.

If symbol X is in exactly two internal nodes then, by an exhaustive evaluation of all possibilities, there are at most a total of 20 instances of X.

If symbol X is in exactly one internal node then either that internal node is the center node or it is not. If it is a non-center node then it can be seen that at most 18 instances of X can occur. However, if it is the center node then there is a unique way to have 24 instances of X, as shown in Figure 4.

Figure 3: X in 3 internal nodes

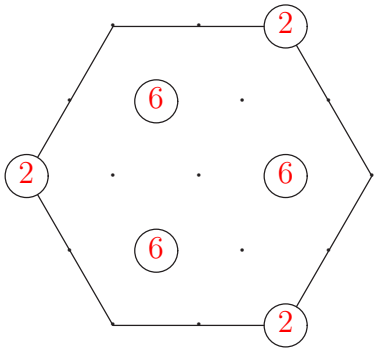
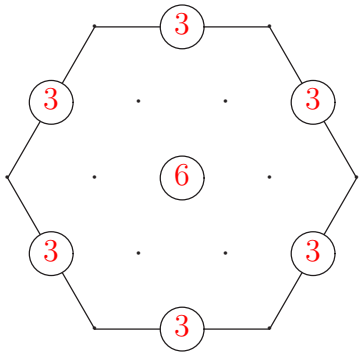


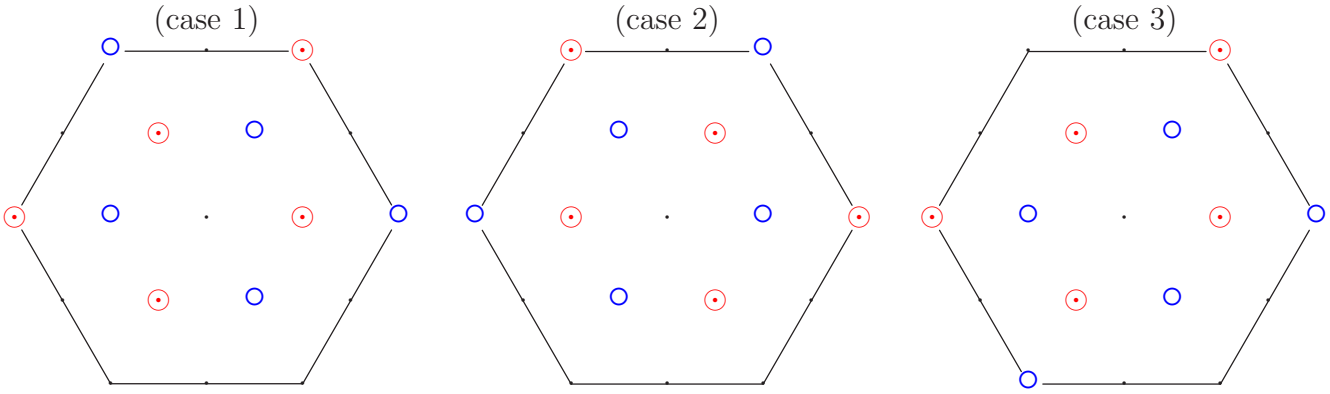
Figure 4: X in center node



Thus, a symbol cannot occur in more than 24 vertices, nor can it occur in exactly 23 vertices. There are only two placement patterns (plus their rotations) enabling a symbol to occur in exactly 24 vertices and there is only one placement pattern (plus its rotations) enabling a symbol to occur in exactly 22 vertices. Using these three placement patterns, it is easy to enumerate all feasible placement patterns that enable a symbol to occur in exactly 21, 20, or 19 vertices.

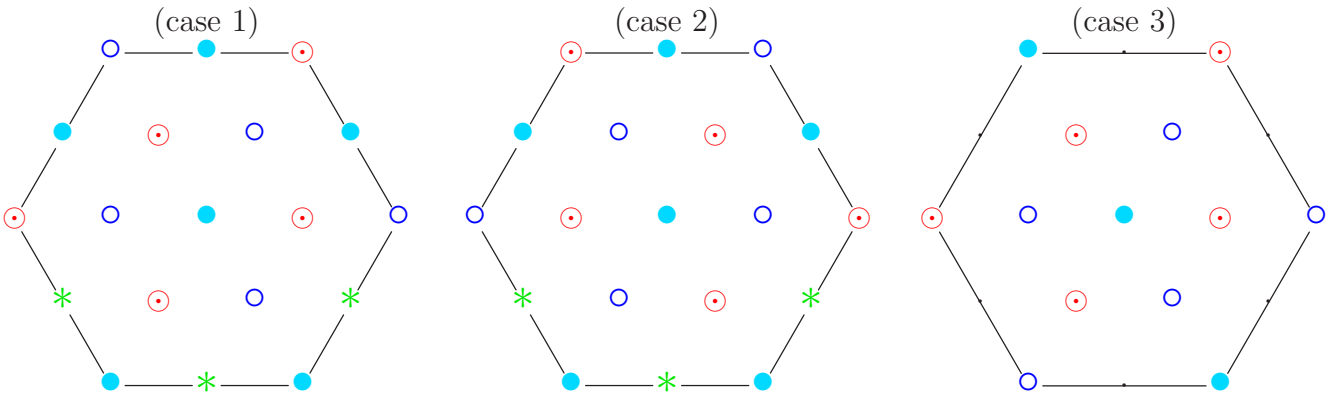
We now consider allowable patterns that combine the use of different symbols with the given frequencies.

Figure 5: Placing 22 ⊙'s and 22 ○'s



There are only 3 ways (plus rotations) of placing the 22 ⊙'s and 22 ○'s, as shown in Figure 5. Of these, case 2 is a mirror of case 1. If there are 19 ●'s they must be: center node (6) + 3 midside nodes (each with 3) + 2 corner nodes (each with 2). The 9 ∗'s use the other 3 midside nodes. As seen in Figure 6, case 3 is impossible as the 2 corners used by ●'s restrict 4 midsides.

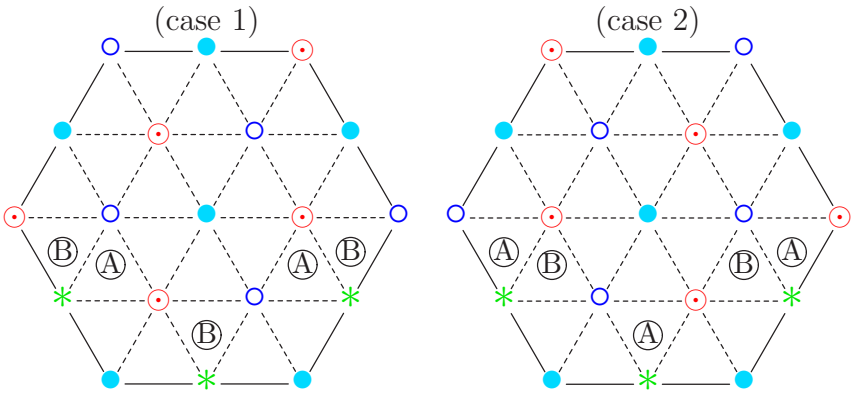
Figure 6: Placing 19 ●'s



5 Subpattern Frequencies

We can reduce the number of feasible cases by making use of the disparity of frequencies for some subpatterns. The existence of any such disparity is guaranteed to diminish the feasible set cardinality by at least half. As shown in Figure 7, we label with (A) those triangles whose vertex symbols are ∗ ○ ⊙ in clockwise order, and label with (B) triangles having those symbols in counter-clockwise order. We note that case 1 has two type-A triangles and three type-B triangles, while case 2 has three type-A triangles and two type-B triangles. In construction of the puzzle, we are now assured that the puzzle pieces will yield only one solution hexagon. We shall choose case 1 for our puzzle.

Figure 7: Cases labeled with type-A/B triangles



6 Ensuring a Unique Arrangement

During the puzzle construction, we wish to ensure that there is only one arrangement of the pieces (modulo piece equality) that yields the solution. We do this by iteratively arranging a unique placement within the solution hexagon of a puzzle piece.

In constructing the puzzle, we note that a rhombus (1) containing type-A and type-B triangles with ○'s at the narrow ends can be placed in only one location. That leaves only one location for placing another rhombus (2) containing a type-A triangle with a ○ at a narrow end, as shown in Figure 8.

Then a rhombus (3) containing a type-B triangle with a ○ at a narrow end can be located in only one place, which then leaves a rhombus (4) containing type-B triangle with a ∗ at a narrow end with only one possible placement. Figure 9 shows the result.

Figure 8: Solution hexagon

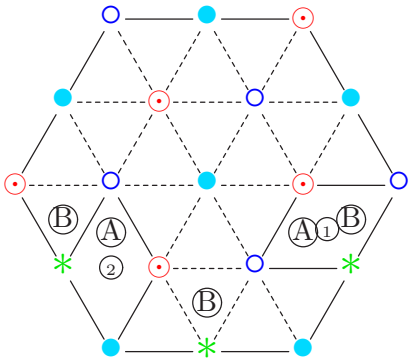
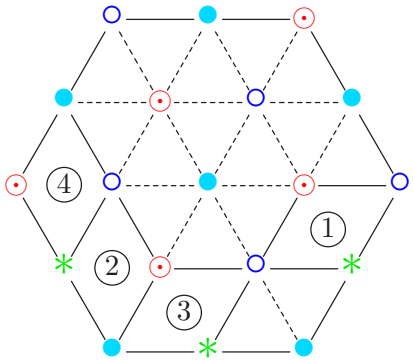


Figure 9: Solution hexagon



It is easy to see that a rhombus (5) containing ∗ at both narrow ends can now be placed only at the lower right of the solution hexagon. There is only one location for the base of a large trapezoid (6) with ●'s 3 units apart, and the placement of such a trapezoid is uniquely determined by the order of symbols in its base. We choose to place the large trapezoid on the periphery. The results obtained thus far are shown in Figure 10.

Figure 10: Solution hexagon

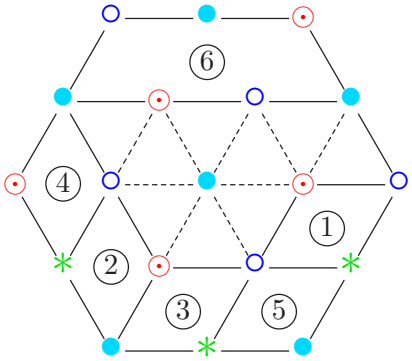
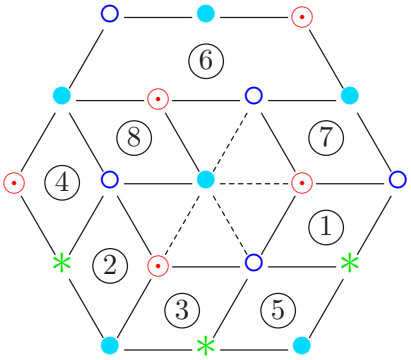


Figure 11: Solution hexagon



At this point, there are nine remaining triangles, which can be used to form 3 rhombi, each consisting of two triangles, plus one trapezoid, consisting of three triangles. By choosing the trapezoid to have a ⊙ in its upper right corner, the rightmost space must be used by a rhombus (7), the leftmost space must be used by a rhombus (8). See Figure 11.

The remaining space must be for a rhombus (9 and the trapezoid (10). The placement of these last two pieces will be uniquely determined by the symbols at the narrow ends of the rhombus. The complete solution hexagon is shown in Figure 12. The problem can be presented by rearranging the pieces within the hexagon, as shown for example in Figure 13.

Figure 12: Final Solution

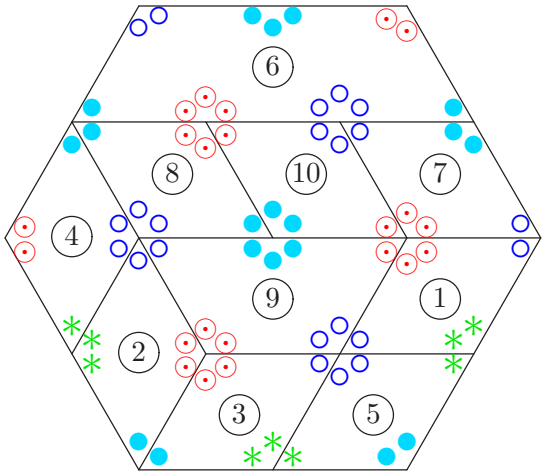
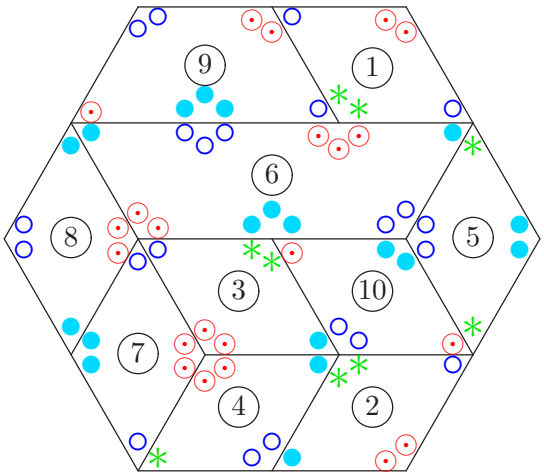


Figure 13: Hexagon problem



References

[1] H. Freeman and L. Gardner. Apictorial jigsaw puzzles: the computer solution of a problem in pattern recognition. *IEEE Trans. Electronic Computers* 13,2 (1964) 118-127.

[2] T. Altman. Solving the jigsaw puzzle problem in linear time. *Appl. Artificial Intelligence* 3 (1989) 453-462.

[3] H. Wolfson, E. Schonberg, A. Kalvin, and Y. Lamdan. Solving jigsaw puzzles by computer. *Annals of Operations Research*, 12,1 (1988) 51-64.

[4] D. Goldberg, C. Malon, and M. Bern. A global approach to automatic solution of jigsaw puzzles. *Computational Geometry* 28 (2004) 165-174.

[5] E. D. Demaine and M. L. Demaine. Jigsaw Puzzles, Edge Matching, and Polyomino Packing: Connections and Complexity *Graphs and Combinatorics* 23 Suppl 1 (2007) 195-208.