CMU-ITC-91-092
10 January 1991

EQUIVALENCE CLASSES AMONG PENTOMINO TILINGS OF THE 6x10 RECTANGLE

WILFRED J. HANSEN
Information Technology Center
Carnegie Mellon
Pittsburgh, PA 15213
USA

ABSTRACT

Although there are 2339 pentomino tilings of the 6 x 10 rectangle, many pairs of solutions are similar. We show that the solutions can be divided into 911 equivalence classes by the similarity transformations:

rotate or reflect a subset of the pieces
interchange two congruent subsets
rearrange two pieces

Early steps of the computation utilize the standard "polar" representation of solutions to rapidly determine the differences between two solutions. Next the pieces that differ are checked to see if they all are transformed the same in transition from one solution to the other; this trick substantially reduces the computation. Later steps manipulate bit mask representations of shapes and rely for speed on special data arrays to reflect a column at a time.

1. The problem

Many workers have found the 2339 unique solutions to tiling the 6 x 10 rectangle with pentominoes[1] [Golomb, 1965]. Occasionally a note will observe that one solution can be transformed into another by moving a few pieces, but no systematic study of this phenomenon has been reported. For example, the solution in Figure 1 can be transformed by

a) flipping the LNPT pieces about a vertical axis,
b) flipping the NZ pieces about the northeast-southwest diagonal,
c) interchanging the INPT and FUVX groups, or
d) rearranging the two pieces L and P.

The reader may be either amused or frustrated to find further transformations of Figure 1, including one where an end of the W is surrounded by the U. The latter is achieved with two of the above transformations and then two more. In all, the equivalence class containing this solution has twelve members.

[1] *Pentomino* is a registered trademark of Solomon W. Golomb.

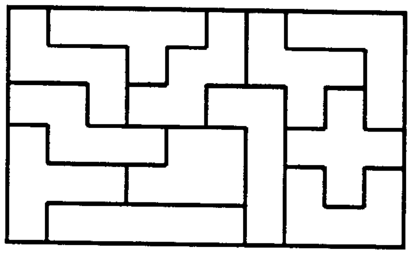

Figure 1. A tiling with four possible similarity transformations.

Having observed that in some cases a whole sequence of solutions are related by simple transformations, the problem naturally attracted me when looking for pattern specification problems as part of research on a new subsequence algebra [Hansen, 1990]. To provide a baseline, the program reported here was coded in C.

It is possible that the most important application of pentomino studies is to intrigue students and expand their spatial relations skills. However, the topic has proven a fertile topic of intellectual study with investigations into various tiling problems, especially on the infinite plane [Conway, 1990; Golomb, 1985]. Tiling problems have also been used to test hardware concepts such as parallel processor architectures [Furuichi, 1990] and neural network machines [Kajiura, 1989]. My own aspirations in undertaking the study included finding good computer representations for shapes and the hope that a better understanding of symmetries could lead to better techniques for manually solving the tiling problem.

2. Definitions

The twelve pentominoes are as given in Figure 2. For this paper, a *solution* is a tiling of the 6 x 10 rectangle with all 12 pentominoes. A *subset* of a solution is a subset of its tiles retained in their relative positions, although possibly rotated or reflected. It is not required that the pieces of a subset be connected, though they usually are.

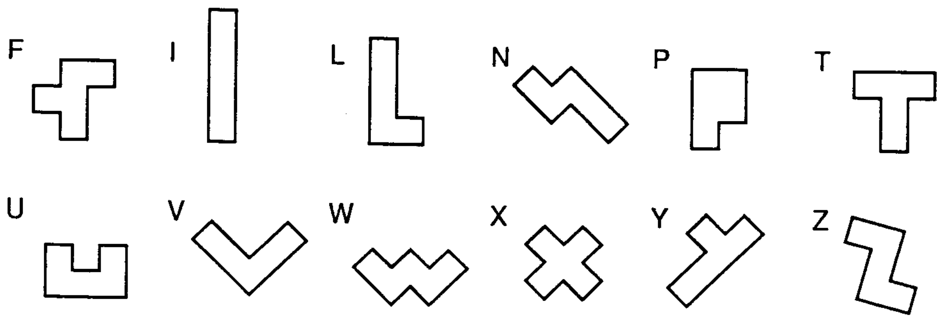

Figure 2. The twelve pentominoes. Each is named with a letter suggesting its shape.

Two solutions are in the same *equivalence class* if they are identical except for a *similarity transformation*, where the similarity transformations of interest are

1) a single symmetric subset,
2) two subsets of congruent shape, and
3) two pieces.

Note that by the definition of subset two solutions are not equivalent if the difference between them is the rearrangement of the pieces in a symmetric group. Thus the two solutions in Figure 3 are not equivalent despite differing only in a symmetric shape consisting of the LNPVWYZ pieces.

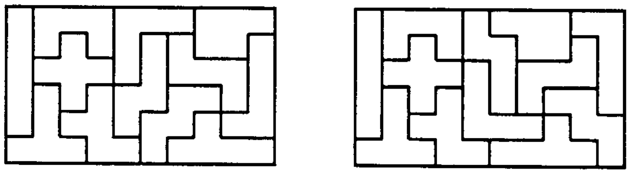

Figure 3. Two non-equivalent solutions. The 6x6 shape on the right is symmetric, but its pieces are rearranged.

Case 3 is met by only the FN and LP pairs shown in Figures 4a and 4b. When these patterns occur, they often merge two sets of solutions with interesting transformations, so we decided to add them to the definition of equivalence. Ninety-one instances of these patterns were found, resulting in 54 fewer equivalence classes.

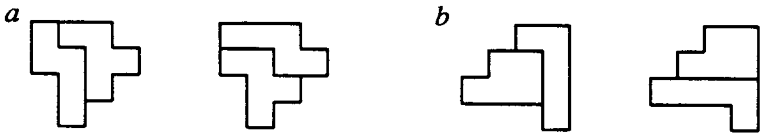

Figure 4. Asymmetric transformations with two pieces. These shapes appeared as symmetry transforms between two solutions. (a) 81 times; (b) 10 times. There are other pairs of pieces that can make a shape two ways, but none appear as similarity transformations between solutions.

The term *orientation* will refer to any rotation or reflection of a subset. If the subset exceeds six units in width, it has the four orientations consisting of itself, its half-turn rotation, and reflections about its horizontal or vertical axis. A subset not exceeding six units also has reflections about each diagonal and rotations of a quarter turn in each direction. When comparing two solutions, a *re-orientation* of a piece is the change in orientation it undergoes if moved from one solution to the other.

3. Program Outline

When working by hand equivalences are usually found by scanning a solution and looking for similarity transformations to other solutions. However, computer implementation of this approach would require consideration of 2^{12} subsets of each solution. Instead we adopted the approach of comparing each pair of solutions to see if the subset of pentominoes which differ between them meets one of the transformations. In most cases two solutions differ in the placement of all twelve pentominoes, so they cannot have any equivalence.

The principle data structures are those for "mask", "oriented pentomino", and "solution". A *mask* data structure represents the outline shape of one or more pentominoes; it contains the height, the width, and a bitmap with one byte for each of up to 10 columns. The bit map contents are normalized by shifting so neither the bottom row nor the leftmost column is all zeroes.

Each pentomino is represented with a set of up to eight *oriented pentomino* data structures, one for each orientation. The data structure has a mask giving the shape of the piece and an array of pointers to the other orientations of the same pentomino. With the latter a single array access yields the result of applying any given re-orientation to the pentomino. For example, suppose Lx is the array of pointers for the L piece oriented as on the left side of Figure 4b. Then Lx[NortheastSouthwestReflection] points to the oriented pentomino structure for the L on the right of the Figure.

The *solution* data structure has both a string form of the solution and its "polar" representation. The latter consists of an array of twelve elements, each pointing to an oriented pentomino structure and giving its row/column location in the solution. Two solutions have pentomino j in common if and only if they have the same values in the j'th element of their polar representations.

The steps of the computation are outlined in Algorithm 1. The first step (labeled {1}) is to assign to each solution its own unique equivalence class. Then {2} all pairs of solutions are compared--considering all four orientations of the first against the second--and when two are found equivalent, their classes are merged {9} by changing all solutions which have the class number of the first to have the class number of the second.

```
{1} Assign a unique equivalence class number to each solution.
{2} For each pair of solutions, in each of four relative orientations:
{3}      Compute Ndiff, the number of differing pieces.
{4}      If Ndiff is 12, the solutions are not equivalent.
{5}      (see text)
{6}      If Ndiff is 2,
{7}      Or if the differing pieces are related by symmetry,
{8}      Or if Ndiff is even and the differing pieces can both be
                  divided in the same two congruent subsets,
{9}               Then merge the two equivalence classes.
{10} Output solutions with final equivalence class numbers.
```

Algorithm 1. Computing equivalence classes. This algorithm is sufficient, but can be accelerated by adding the step 5 described in the text.

Step {3} computes Ndiff, the number of pentominos for which two solutions differ. A value of one is impossible; eleven cannot occur if there is a similarity transformation; twelve is the most common and indicates non-equivalence instantly {4}. When Ndiff is two, the solutions are equivalent by (at least) the third rule of equivalence {6}. For other values of Ndiff, the differing subsets are checked to see if they are symmetric {7}. If not and if Ndiff is even the two subsets are further tested to see if both have two congruent subsets which can be swapped to transform between the solutions{8}.

Symmetry of a subset {7} is tested by building a mask representing the shape of the subset and seeing if one of the symmetric re-orientations of the mask is identical to the mask. If this succeeds, the pieces are further tested to ensure both subsets have their pieces in the same relative positions. For this test a polar representation of one subset is successively re-oriented and checked against the other. For this reason, the re-orientation operations are programmed for both masks and the polar representation.

Checking the pentominos that differ to see if they can be split in two congruent subsets {8} is the computationally most intensive task. All $\binom{n-1}{n/2}$ possible halvings of the subset have to be tested; this is only 3 tests for Ndiff of 4, but 126 tests for Ndiff of 10. Fortunately, large values of Ndiff were uncommon because the solution list was partially equivalence-class ordered in earlier experiments. The congruence test for each halving was optimized by first computing and comparing the bounding rectangles before generating masks and checking congruence.

Steps {7} and {8} both require rotation and reflection of shapes represented as mask data structures. Only three such operations were implemented--reflections about the vertical, horizontal, and northeast-southwest axes--because all other re-orientations can be generated by multiple applications of these. Reflection about the horizontal axis was done simply by swapping bytes from opposite ends of the bit mask. Reflection about the vertical axis was computed via a precalculated array, Vreflect, giving the vertical reflection for each byte pattern. For instance, Vreflect[0xB] is the value 0x34 (that is, the reflection of 001011 is 110100). Reflection about the northeast-southwest axis used a precalculated array of bitmaps indexed by column position and contents. If the i^{th} column has contents j, the bitmap NESW[i][j] has one bits in the right positions to be a reflection of j in the i^{th} column. The bitmaps selected by each byte of the mask are OR'ed together to make a composite bitmap for the northeast-southwest reflected mask value.

As described so far, the algorithm required three and a quarter hours on an IBM RT/PC workstation with its APC card. This is not an unreasonable use of resource since workstations normally sit idle every night. However, it was a pleasant surprise to find an additional step which greatly reduced the computation.

Observe that when a subset of pieces is re-oriented for a symmetric transformation from one solution to another, all the pieces must undergo the same transformation: all reflected horizontally, or all rotated 180 degress, or whatever. Similarly when swapping two congruent subsets between two solutions all pieces must undergo the same re-orientation, except in the case of one-quarter turns, where half the pieces turn one way and half the other. The improved algorithm was created by introducing step {5} to test that all pieces have the same re-orientation between the two puzzles; for simplicity, all quarter turns were trteated as equal.

The improved algorithm took only 25 minutes on the same processor. Most of the work of the algorithm is done in steps {4} and {5}, the first of which rejects all but 636402 of the 10931601 solution pairs considered and the second of which rejects all but 2633 of the remainder.

4. Results

A total of 911 equivalence classes were identified with the largest class having fifty members. Sample solutions from this and other large classes appear in Appendix 1.

One interesting aspect of equivalence classes is their "bushiness". Some classes have a sequence of transformations from one solution to the next, where each solution has only one transformation in and another out; other classes have several transformations applicable in each solution so a few transformations can generate a relatively large class. This aspect of the various classes is revealed in Figure 5 where the x-axis orders the equivalence classes according to size and the y-axis is the number of transformations applicable to solutions in the class.

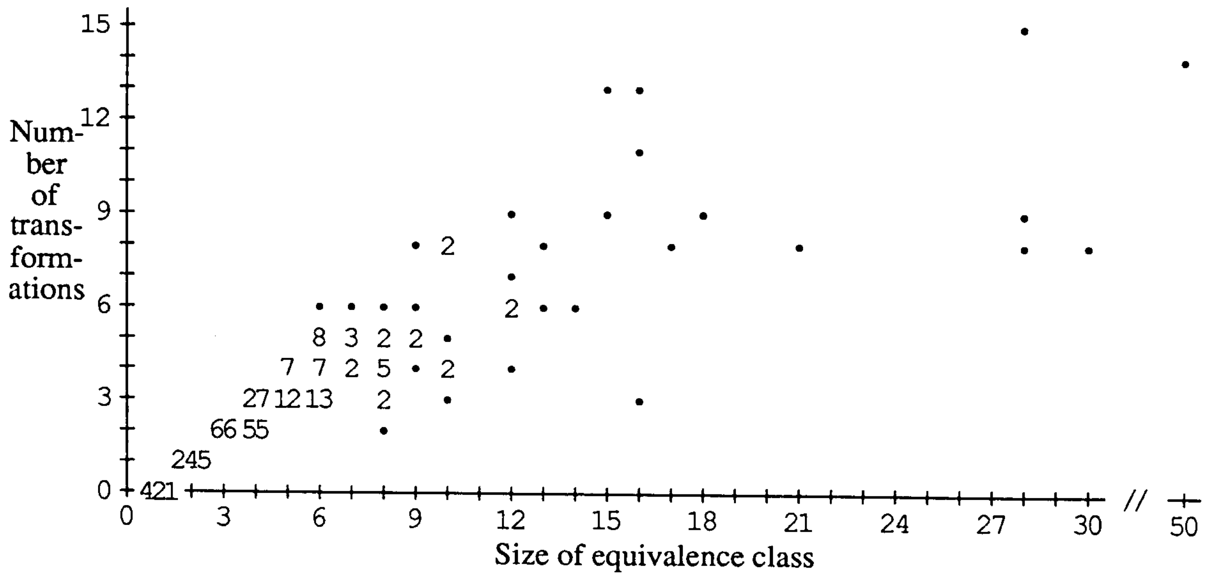

Figure 5. Equivalence classes organized by number of solutions and number of transformations that apply. A value of one is represented with a bullet.

From the Figure we observe that about half the classes have one member and half the remainder have only two. Altogether, however, these account for only about a third of all solutions. Local maxima occur in the table for class sizes which are a multiple of four solutions because symmetries contribute an even number of solutions and multiple symmetries drive the class size toward a multiple of a power of two. One class, denoted 16b in the Appendix, has 16 solutions generated from only 3 transformations; this happens because one of the transformations is the symmetries of the 3x5 rectangle formed from pieces LNV. This is a prolific transformation, appearing 54 times as a symmetry transform from one solution to another, including an appearance in the largest class.

The similarity transformations identified include

1550 cases of a symmetry transformation
433 cases of swapping two congruent pairs, and
91 cases where two pieces were rearranged asymmetrically.

The cases in the middle group can be further divided into 225 where the transformation was also symmetric and another 208 where it was not. The numbers above total to more than the number of equivalence classes because every similarity transformation between solutions is counted even though in many cases more than one sequence of transformation will convert one solution into another.

The definition of subset demands that the difference between two solutions be formed by moving the pieces of the subset as a block, without rearranging their internal order. Relaxing this restiction produces many more equivalences by symmettry with arbitrary rearrangement of pieces. Curiously, however, the number of equivalence classes is reduced by only three if the restriction is relaxed for the similarity transformations of swapping two congruent subsets. The additional equivalences arise from the shapes shown in Figure 6.

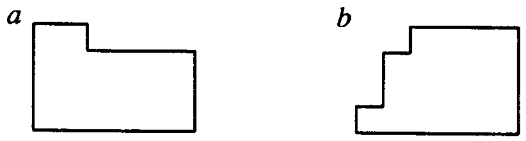

Figure 6. Shapes that add three more equivalences by swapping two congruent subsets. (a) Can be made two ways with pieces INPT. (b) Can be made three ways with pieces FVYZ.

The various similarity transformations involved 199 subset shapes, as detailed in Appendix 2. One shape--the 3x3 square with a unit square in the middle of one side--appears in both lists and is counted twice here. Curiously, it is the only symmetric shape that appears among the shapes which form pairs of congruent subsets. Many shapes can be made with more than one set of pieces; altogether 320 combinations of shape and composition occurred.

Four of the shapes discovered are not connected subsets. In each case, however, the pattern is composed of two symmetric and connected components, each of which applies individually, so the underlying solutions would be in the same equivalence classes even if disconnected shapes were disallowed.

Many of the smaller shapes appear as subsets of other, larger ones, so apparently the symmetry structure of pentomino tilings is not yet completely elucidated.

Acknowledgment: Rebecca L. Hansen's curiosity and enthusiasm contributed greatly to my choosing and finishing this work.

References

Conway, J.H., Lagarias, J.C., "Tiling with polyominoes and combinatorial group theory." *Journal of Combinatorial Theory, Series A, vol. 53,* no. 2, March 1990, pp. 183-208.

Furuichi, M., Taki, K., Ichiyoshi, N., "A multi-level load balancing scheme for OR-parallel exhaustive search programs on the Multi-PSI." *SIGPLAN Notices, vol. 25*, no. 3, March 1990, pp. 50-9; from proceedings of Second ACM Sigplan Symposium on Principles and Practice of Parallel Programming, Seattle, WA, USA, 14-16 March 1990.

Golomb, Solomon W., *Polyominoes*, Scribner (New York, 1965).

Golomb, S.W., "Polyominoes which tile rectangles." *Journal of Combinatorial Theory, Series A, vol. 51,* no. 1, May 1989, pp. 117-24.

Wilfred J. Hansen, "Programming Language Support for Multi-Media Text with an Algebra for Subsequences", Information Technology Center, Carnegie-Mellon, 1990.

Kajiura, M., Akiyama, Y., Anzai, Y., "Solving large scale puzzles with neural networks." in *Architectures, Languages and Algorithms*, IEEE Comput. Soc. Press (Los Alamitos, CA, 1989) pp. 562-9; from proceedings IEEE International Workshop on Tools for Artificial Intelligence, Fairfax, VA, 23-25 Oct. 1989.

Appendix 1. Representatives of the largest equivalence classes

Figure A1.1 gives one solution from each of the forty-nine equivalence classes whose size is seven or more. The integer in front of the solution is the size of the class; the lower case letters are merely to distinguish classes.

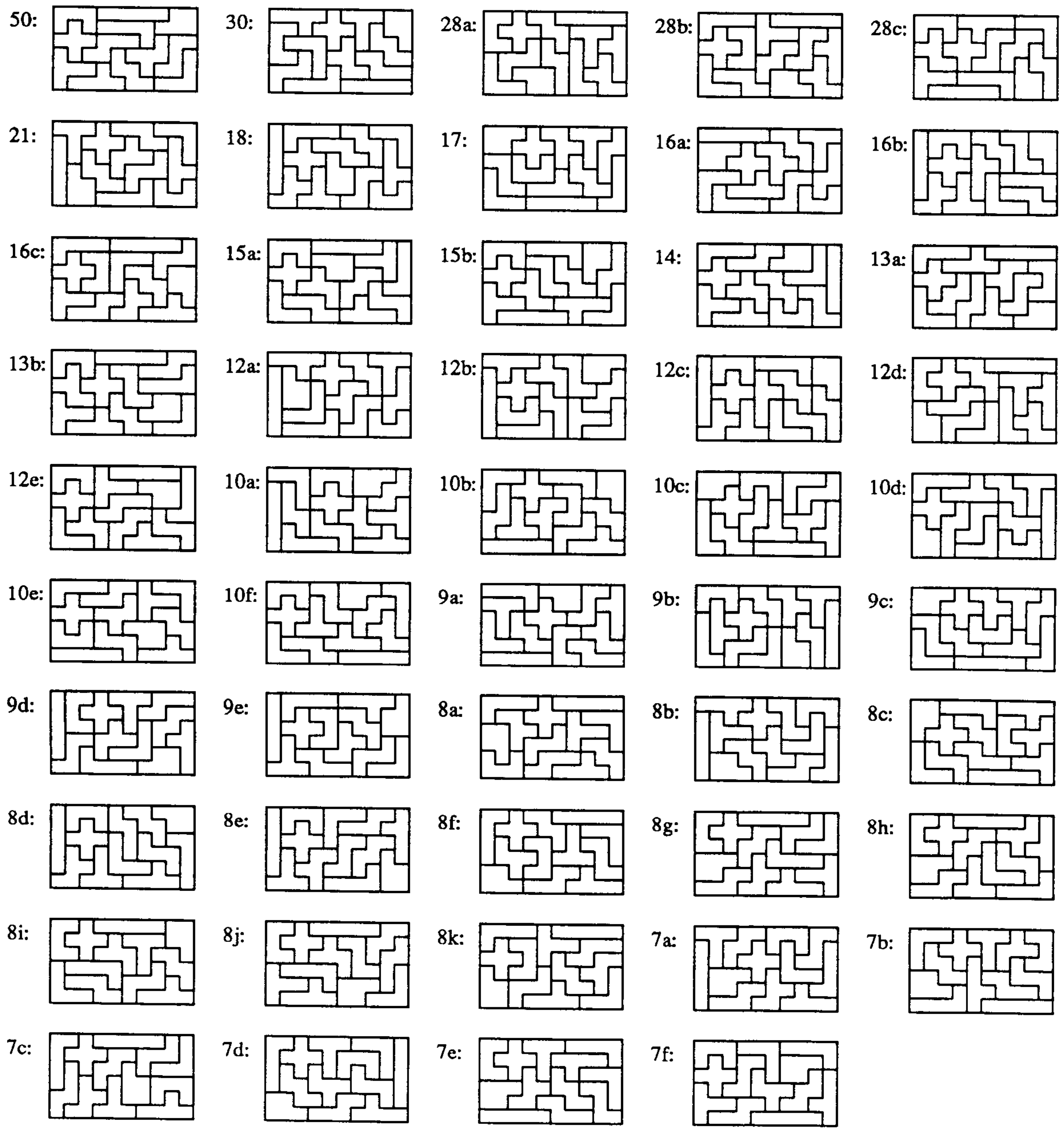

Figure A1.1 Solutions from large equivalence classes.

Appendix 2. Shapes of subsets

Where there are similarity transformations between pairs of solutions, the set of pieces re-oriented between the two solutions has one of the 199 shapes shown in the Figures of this Appendix. Under each shape is shown the set(s) of pieces that constituted that shape and the number of times that set appeared in the given shape.

Figure A2.1 shows the 73 shapes that occurred in swapping two congruent subsets; all but one are asymmetric. Sometimes a similarity transformation that could be expressed as swapping two congruent subsets could also have been expressed as a symmetry transform; the two counts for each piece set are for the cases where symmetry did not hold or did hold, respectively.

The shapes most often found in swapping two congruent subsets had the following number of occurrences and alternate piece compositions:

70	PV FP NP UX
42	LN PU VZ LP
36	FL TY
29	FTY LNU FNV PUX
26	NV TY IL
22	LP IT
21	FU PV LP PT PZ UY

Figures A2.2 and A2.3 show the 126 shapes that occurred when there was a symmetric subset transformation. Frequently occuring such shapes were:

112	FX
103	FT
81	PV FP NP
81	PT NZ
76	PW LP
76	PUV FPU PUY LNV LTY
68	TV
56	PY LW
52	LU

In addition to the shapes shown below, similarity transformations occurred with the shapes given in Figures 4a and 4b.

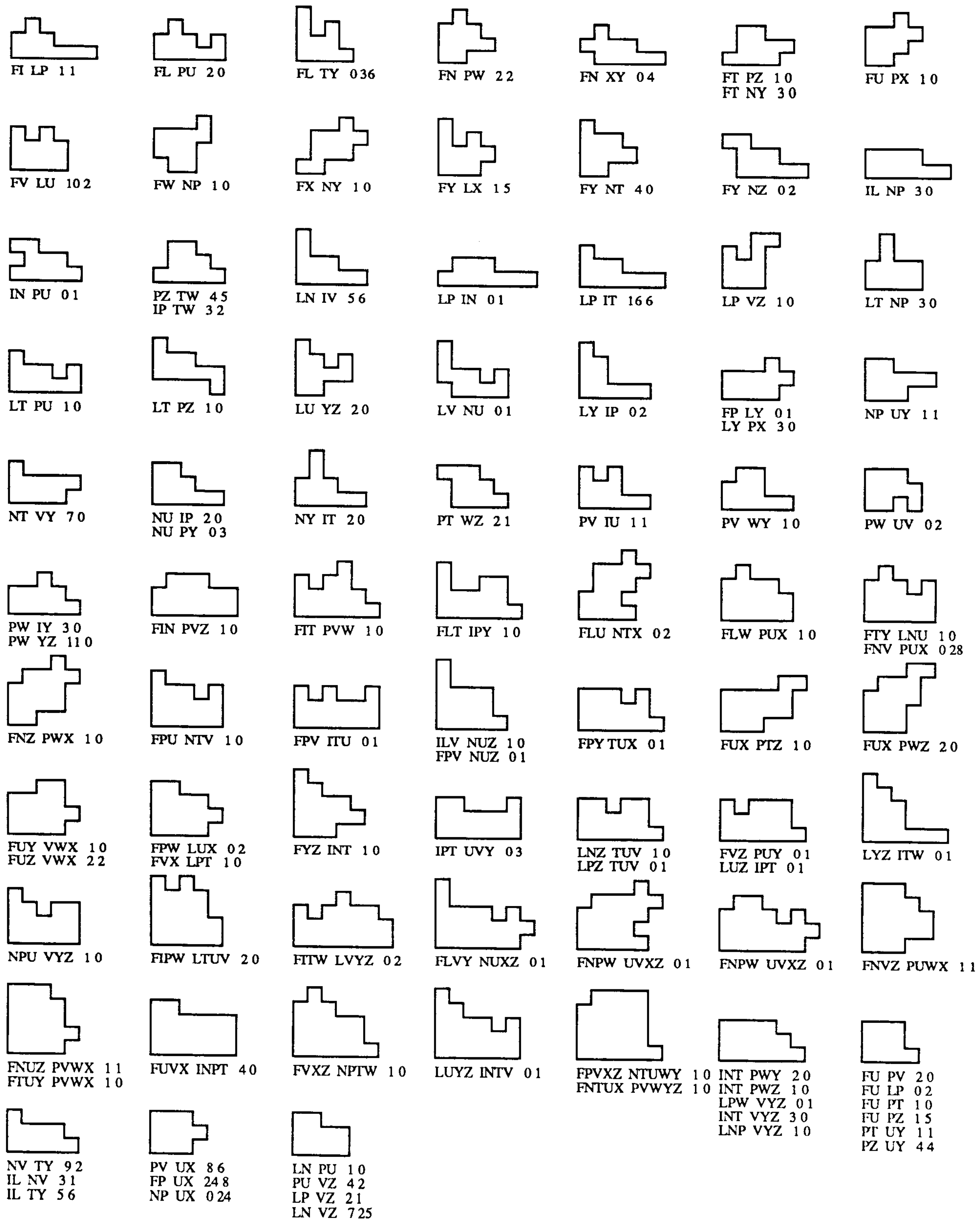

Figure A2.1 Shape of each member of swappable congruent pairs.

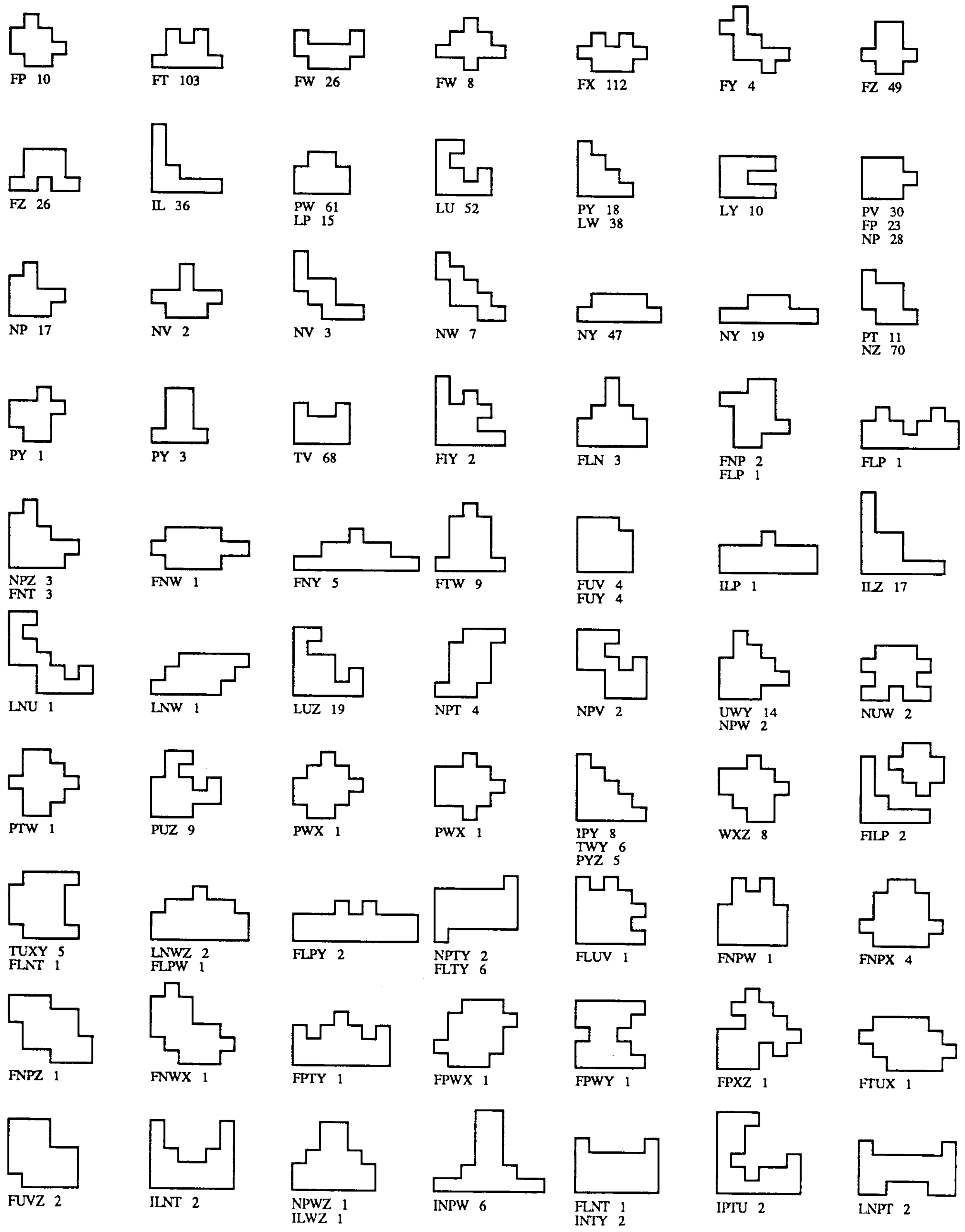

Figure A2.2 Shapes of symmetric groups; part 1.

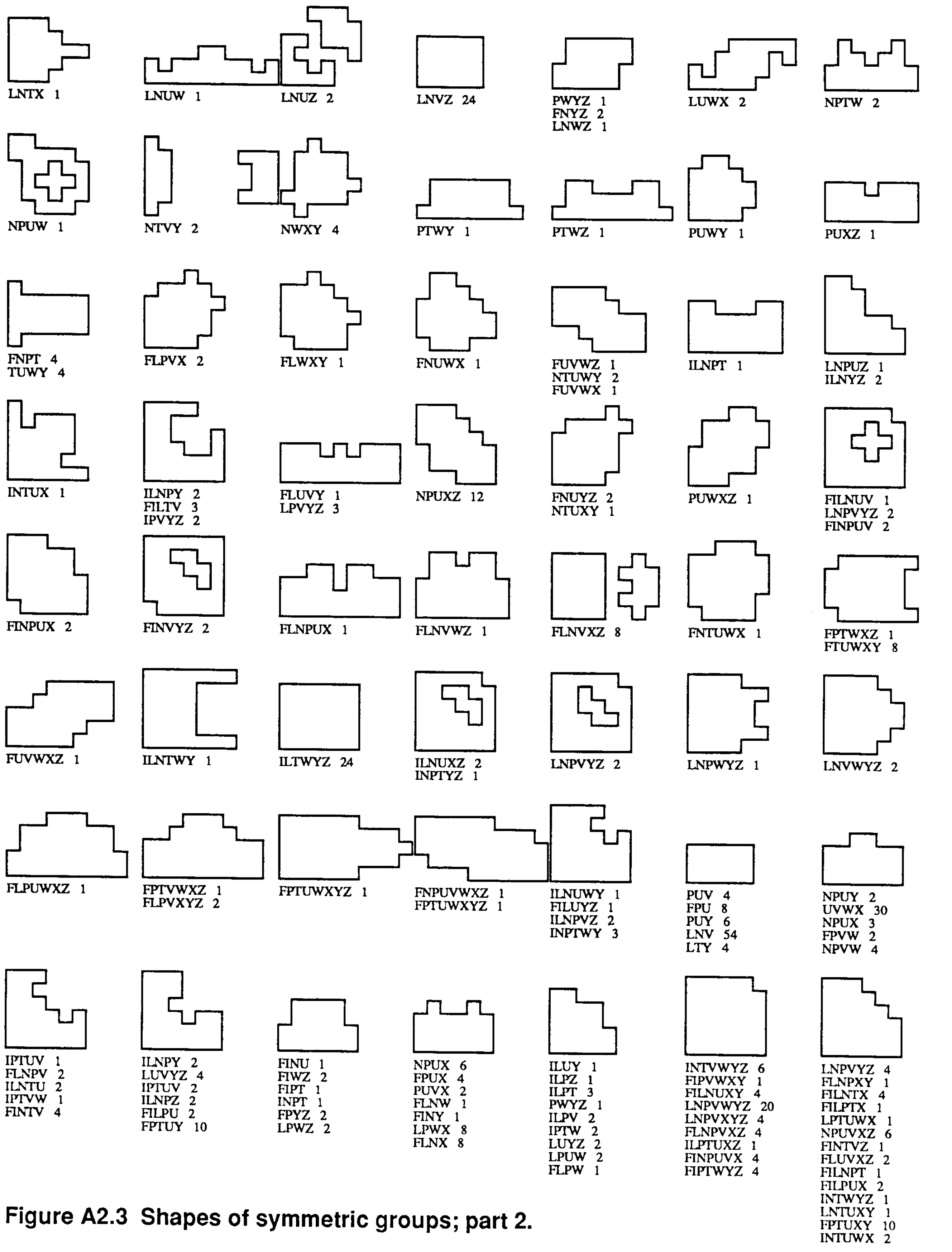

Figure A2.3 Shapes of symmetric groups; part 2.