Feature-based Part Retrieval for Interactive 3D Reassembly

Devi Parikh[1] dparikh@cmu.edu
Rahul Sukthankar[2,1] rahuls@cs.cmu.edu
Tsuhan Chen[1] tsuhan@cmu.edu
Mei Chen[2] meichen@ieee.org

[1]Carnegie Mellon University
Pittsburgh, PA 15213

[2]Intel Research Pittsburgh
Pittsburgh, PA 15213

Abstract

We propose a novel framework for 3D reassembly, the task of assembling a solid object from its broken pieces. The primary challenge in this under-explored problem is to robustly establish compatibility between parts from one object. Feature-based techniques have shown success in domains such as 3D similarity search; unfortunately, the global features typically employed to quantify whole-object similarity are unsuitable for identifying part-level compatibility. Therefore, we propose the use of local features which, in conjunction with robust matching, have become popular for object recognition in 2D images. This paper demonstrates that an analogous framework can be successful for 3D reassembly. Automating part-level compatibility enables the construction of an interactive system for 3D reassembly, where the user can easily assemble a desired object from a large collection of pieces (many of which are irrelevant) by iteratively selecting compatible parts. We evaluate our approach on a simulated database of broken objects and show that it scales well in the presence of noise and extraneous pieces.

1. Introduction

3D reassembly, the problem of assembling a broken solid object given a collection of its parts, forms an important component of several real-world applications including molecular biology [21], archeology [14] and spacecraft post-crash failure-analysis [18]. Despite its broad applicability, computer-aided 3D reassembly has been relatively under-explored by the computer vision community. Figure 1 illustrates a typical instance of the problem where, given a large number of 3D parts, the goal is to identify those that could be assembled to form a previously unknown object (the cube). At the lowest level, a valid solution must ensure that adjacent parts are locally compatible. At a higher level, one may need to employ domain knowledge to select among multiple candidates that are compatible with a given part. This paper focuses on the former aspect and enables users to interactively specify higher-level preferences.

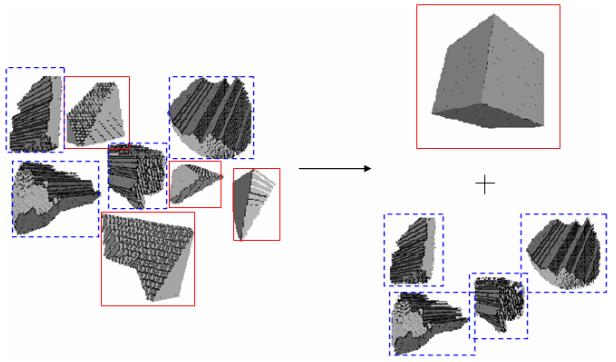

Figure 1: *Given a large set of parts from a mixture of broken solid objects (left), our goal is to identify those that are locally compatible so that they may be interactively reassembled to construct a complete object, while the extraneous parts are left unused (right).*

3D reassembly poses several unique challenges beyond those explored in 3D object similarity. First, global features that work well for the latter problem fail on our task. This is because two parts that are locally-compatible can be very dissimilar at a global level. Figure 2 (left) shows a typical scenario in 3D object similarity [3] where global features can be employed to map a query to similar shapes in the database. The challenge there is to find features that generalize over the shape variations within the target class. Figure 2 (right) shows a corresponding scenario in 3D reassembly, where locally-compatible parts appear very dissimilar from a global viewpoint. Naively representing each part as a single point in a high-dimensional feature space would fail to represent the structure in corresponding local regions that defines compatibility between the parts.

Second, although part compatibility is a local phenomenon, independently identifying locally-compatible regions between two parts is insufficient since a valid match must also satisfy global geometric constraints (i.e., the two parts should assemble without stretching, warping or interpenetration). It is this combination of local compatibility

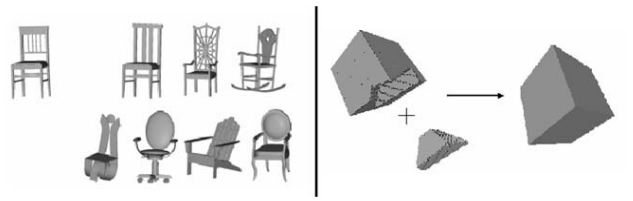

Figure 2: *Differences between 3D object similarity and 3D reassembly. (left) Indexing objects using suitable global features enables a query in 3D object similarity to match similar shapes in the database. (image courtesy of [3], used with permission); (right) Two compatible parts in 3D reassembly typically look very different at a global scale .*

and non-local consistency that makes the problem interesting.

Third, the problem of identifying whether two given parts are physically compatible is inherently a continuous problem — two 3D shapes can contact in an infinite number of ways; naively sampling the space of such configurations to determine whether the parts are locally compatible is unlikely to succeed. This motivates the need for efficient methods for determining part compatibility.

Finally, in practical 3D reassembly scenarios, the database of parts could be missing pieces or contain large numbers of "extra" pieces from other objects. Automatic reassembly of objects under these constraints without *a priori* knowledge of the target object is extremely challenging. However, an interactive system that automatically identifies candidate parts that are compatible with a partial reconstruction could enable a user with appropriate domain knowledge to efficiently reconstruct complex objects by significantly reducing the search space. For instance, an archaeologist could find the pieces that form a given vase while ignoring pottery shards from other broken objects (Figure 1 illustrates a subset of available pieces forming a cube).

This paper proposes a feature-based approach to determine part compatibility in the context of an interactive 3D reassembly task. The rest of the paper is organized as follows. Section 2 summarizes the related work in this area. Section 3 presents the proposed framework and our implementation choices. Section 4 describes the experimental setup and shows results. Section 5 concludes the paper and identifies some promising directions for future research.

2 Related work

Early research on 2D jigsaw puzzle solving includes [7, 25]. However, these approaches focus on commercially-produced jigsaw puzzles, where part contours are typically regular [23]. An attempt to remove these restrictions is made in [8, 10, 15, 20], which present an approach for the reassembly of real-world 2D objects.

There has been relatively little research in 3D reassembly. [4, 15, 22] advocate an approach based on curve matching. However, measuring curvature and torsion for 3D curve-matching involves higher-order derivatives that are sensitive to noise. The papers show no results for more than two 3D pieces. The underlying assumption of [19] is that the pieces are nearly planar and match each other completely. This does not allow for partial fitting. Willis *et al.* [23] propose a Bayesian framework to automatically reassemble 3D pieces. However, they assume that the fragments assemble to form an axially-symmetric object (such as a pot). Winkelbach *et al.* [24] employ a generate-and-test approach to determine the optimal relative pose between two parts, which scales poorly.

3D similarity matching, unlike 3D reassembly, is a well-studied problem and comprehensively surveyed in [3]. The majority of established techniques employ global representations that are unsuitable for our task. Feature-based methods have computational benefits over generate-and-test approaches, and have become popular in multimedia retrieval [5] and 3D similarity. This motivates us to explore feature-based approaches for 3D reassembly.

As [2] indicates, feature-based methods may not be as effective for reassembly since reassembly is inherently a continuous problem, and projecting an object onto a single point in the feature space can be problematic. This may be alleviated if numerous local features are extracted from each piece, so that the piece maps onto several points in the feature space. As [15] observes, local shape analysis causes several ambiguous matches in the database, especially for large databases; this can be effectively addressed by enforcing non-local geometric agreement among the locally compatible regions.

In very recent work, Huang *et al.* [11] independently propose a feature-based approach for automatic 3D reassembly. Two key differences between our methods is that we employ spectral techniques for non-local matching and focus on the interactive aspect of the problem.

3. Proposed approach

We formulate the 3D reassembly problem as an iterative retrieval problem. When a 3D piece (or partial-assembly of pieces) is posed as a query, our system computes a compatibility score between the query and candidate pieces in the database and displays promising candidates. The user selects one of these matches and requests the system to assemble the chosen part with the query. The new assembly can then be used as the next query. The object is thus assembled interactively and the user can employ domain knowledge to choose among multiple possible compatible pieces

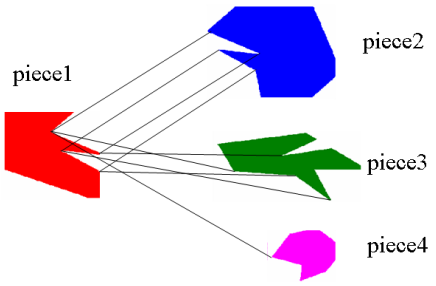

Figure 3: *Illustrative example. Given a query (piece1), the most promising match (piece2) has features that are both locally compatible and geometrically consistent.*

for a location or vice-versa.

Figure 3 presents an overview of the approach, using a 2D example. When *piece1* is posed as a query, the system identifies the set of pieces in the database that have at least a single interest point that is compatible with features on the query. This eliminates the majority of pieces in the database. Among the remaining pieces, some, such as *piece4*, are low-scoring candidates because they match at only one local feature. Others, such as *piece3*, match at multiple locations that are geometrically inconsistent. The most promising candidates, such as *piece2*, have multiple matching features that are also geometrically compatible.

Our framework consists of a set of sequential stages: (1) interest region detection; (2) local description of each interest region; (3) near-neighbor based local correspondence; (4) geometric agreement; (5) spectral technique based scoring. Each of these stages is detailed in the following subsections.

3.1 Interest region detection

The first step is to identify stable interest regions on each piece at which local descriptors can be computed. In the 2D example, the obvious candidates for interest regions are the vertices. For 3D reassembly, reasonable candidates include corners [9, 16], edges and uneven surfaces. In our implementation, we define interest regions to be *key edges* and identify them on each piece. This is done by computing the occupancy within a small sphere at each point on the surface of every piece, and rejecting those that are approximately half-occupied (flat surfaces). We then fit lines through points of similar occupancy using RANSAC [6] to identify key edges.

3.2 Local description of interest regions

The next step is to find a compact but discriminative representation for the local neighborhood around each region of interest. In the 2D example from Figure 3, this could be the angle formed at the vertex. For 3D, established descriptors include measures of angularity, spin images [12], mass per shell shape histograms [1] and spherical harmonics of the Gaussian Euclidean distance transform [13]. If available, these could be augmented using any available appearance information (e.g., surface texture). Our implementation uses the angularity of key edges, measured by the occupancy of a sphere placed at a point on the edge. The key edges (lines) are parameterized with a direction vector and a point on the line.

3.3 Near-neighbor based correspondence

Having extracted local features from the query and a candidate piece the next step is to find potentially compatible regions. Note that two compatible regions are *complementary* rather than similar (e.g., a convex region on the query piece is compatible with concave regions on candidate pieces). In our implementation, since the descriptors are scalar (normalized fraction of occupancy), a natural notion of piece compatibility is how well the two descriptors sum to one.

3.4 Geometric agreement

Having found the correspondences between the query piece and database candidate piece, the next step is to ensure geometric agreement among these correspondences. This is done on a pair-wise basis, i.e., we quantify the degree to which two correspondences are in geometric agreement. Clearly, two distinct correspondences cannot be in agreement if they start (or end) at the same interest region. More generally, each feature correspondence imposes a constraint between respective points on each piece; two correspondences are in agreement if these constraints are consistent. This is illustrated in Figure 4. A 2D example with key points (verteces) as the interest regions are shown for illustrative purposes. In our implementation, where correspondences map key edges to edges, the geometric agreement is quantified as the degree to which the distance between the line features is preserved. Specifically, we assign a weight that decreases linearly with the disparity in the distance between a pair of edges in the query piece and the distance between the corresponding edges on the database candidate piece (and is zero once the disparity exceeds 10%). This definition of geometric agreement is inherently invariant to the relative pose between the two pieces.

3.5 Spectral technique based scoring

Given weights for each pair of correspondences between the two pieces, where only some of the correspondences may be correct, our goal is to identify the cluster of correspondences that is most in mutual agreement and thus generate

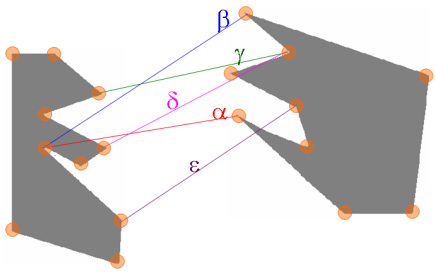

Figure 4: *Correspondences* α *and* β *disagree because they start at the same interest region. Correspondences* α *and* γ *are in greater mutual geometric agreement than* α *and* ε*. This figure illustrates a subset of the correspondences.*

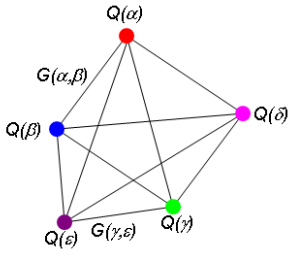

Figure 5: *Graph employed for clustering correspondences for the example in Figure 4. Node weights encode the quality Q of that correspondence while edge weights encode the geometric compatibility G between two correspondences.*

an overall compatibility score. We employ a spectral technique introduced in [17] to accomplish this. We summarize the approach below.

A fully connected graph is constructed whose nodes are the correspondences (the graph has the same number of nodes as the number of correspondences). For the example in Figure 4, the five correspondences map to the five nodes shown in Figure 5. Each node weight encodes the quality of that correspondence (based on the descriptor compatibility). Edge weights encode the geometric agreement between the *pair* of correspondences connected by that edge. We represent this graph by its adjacency matrix, where the diagonal and off-diagonal elements correspond to node and edge weights, respectively.

The primary eigenvector of the adjacency matrix indicates the largest cluster of correspondences that are in mutual geometric agreement [17]. Specifically, we can binarize the eigenvector to obtain a hard assignment of correspondences to the cluster. In our implementation, we perform binarization using an iterative greedy scheme: the highest value in the eigenvector is set to 1, and those entries for correspondences that disagree with the selected correspondence are set to 0 (eliminated). We repeat this process until the entire vector is binarized. The set of retained correspondences could be used to estimate the relative pose between the two pieces when the user requests the system to assemble them. Thus, for an adjacency matrix $\mathbf{M}$ and a binarized primary eigenvector $\mathbf{x}$, the compatibility score between two pieces is given by $\mathbf{M^T x M}$.

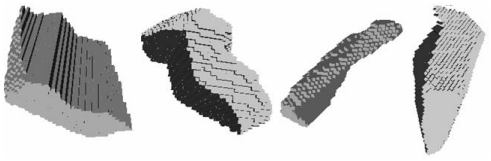

Figure 6: *Examples of 3D pieces generated by simulated breaking of cubes and spheres (100 pieces are used).*

In summary, we detect regions of interest on the two pieces for which a compatibility score is to be computed. Correspondences are established between features extracted at each interest region using a local compatibility metric. We identify the cluster of correspondences between the two pieces that are most in geometric agreement using spectral techniques. This generates a final compatibility score between the two candidate pieces.

4 Experimental setup and results

Experiments were conducted on synthetic data, where 25 solid objects (cubes and spheres) were each broken into four irregular pieces using a random walk algorithm with a cutting plane or band saw to generate a database of 100 pieces (see Figure 6). We describe three experiments below.

4.1 Experiment 1: 100 piece database

The goal of the first experiment was to assess the retrieval quality of our method at the piece level and to examine the effects of noisy data. Each piece in the database was selected as a query and the highest-matching piece was retrieved. Under noise-free conditions, the part-level retrieval accuracy is 100%, providing initial validation of the approach. Figure 7 is a visualization of the compatibility score matrix, where each entry shows the part-level compatibility between a query (column) and candidate piece (row). To simplify visualization, the first four pieces in the database come from the first broken object, the next four from the next broken object, and so on, and the queries are presented in the same order (this information is obviously not presented to the algorithm). We observe that individual objects appear as connected sub-blocks in Figure 7 (right), showing that part-level compatibility between pieces can reliably identify adjacent pieces within an object, without *a priori* knowledge about the shape of the object.

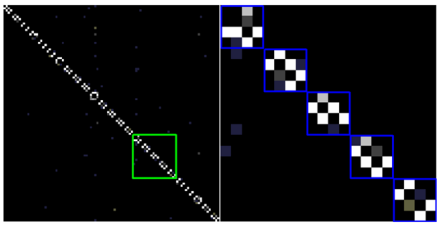

Figure 7: *A visualization of the compatibility score matrix obtained for the noise-free case. (left) The entire matrix (100×100) (right) A zoomed in version of the region marked on the left (20×20). We observe that a non-zero score is typically assigned only to those pieces that come from the same broken object, even in the presence of several other broken objects.*

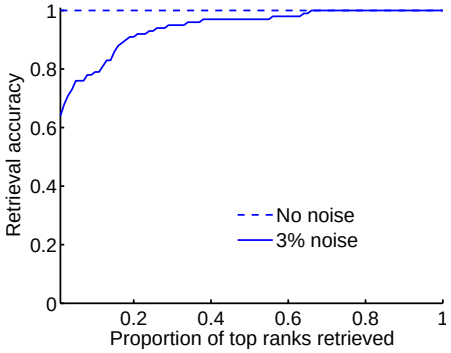

Figure 8: *A cumulative match characteristic curve obtained for noise-free scenario and after adding 3% noise to the voxels.*

A similar experiment was run on noisier 3D data. Gaussian noise with zero mean and 3% standard deviation was added to each voxel of the 3D pieces. Figure 8 summarizes the retrieval accuracy of the system using the cumulative match characteristic (CMC) curve as a function of retrieved rank. Performance under noise-free conditions is perfect and degrades slightly with noise (area under CMC = 0.94).

4.2 Experiment 2: varying database sizes

The second experiment examines the behavior of the algorithm as the size of the database increases — both in terms of retrieval accuracy and computation time. The database size was varied from 4 to 100 pieces (1 to 25 broken objects). Without noise, the rank-1 accuracy was 100% regardless of database size; with 3% noise, the accuracy decreased slightly. Figure 9 plots the area under the CMC curve as database size is changed. We note that the performance of the proposed framework does not significantly degrade as the database grows, even under noisy conditions.

The average time required to retrieve the top match among a 100-piece database in noise-free scenarios is 47s on a standard processor, of which only a few ms are consumed by matching and the bulk of the remainder by

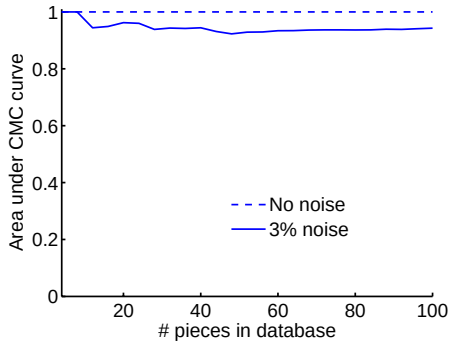

Figure 9: *Retrieval performance of the algorithm for varying database sizes.*

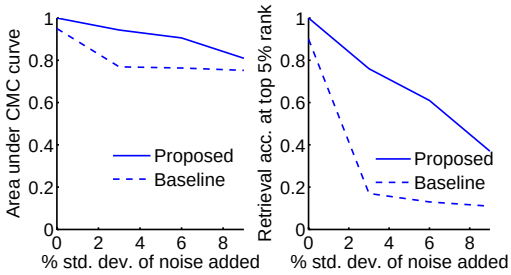

Figure 10: *Retrieval performance of the proposed algorithm compared to a baseline at several different noise levels. (left) uses the area under the CMC as a metric for comparison. For an interactive retrieval system it is important to have higher accuracies at lower ranks, so (right) uses the accuracy at the top 5% rank as a metric. In both cases the proposed method significantly outperforms the baseline approach at several noise levels.*

RANSAC on the query piece (constant for any database size). Thus, there is negligible difference in processing time for different database sizes if feature extraction on the database pieces has been performed in advance. For noisy data, matching takes longer (about 2.5s per piece) and thus the elapsed time increases linearly with database size.

4.3 Experiment 3: varying noise levels

The third experiment conducted was to observe the behavior of the algorithm for different noise levels. The performance of the proposed algorithm was compared to that of a baseline algorithm that uses only nearest neighbor for each local interest region on the query piece, without enforcing geometric agreement. Figure 10 compares the retrieval accuracy according to two metrics: (left) area under CMC curve, and (right) retrieval accuracy at 5% rank. The proposed system shows significant improvements over the baseline according to both criteria. The latter (10right) is particularly relevant for an interactive system since the user will only be shown the best-ranked results; here the proposed approach

is clearly superior.

5 Conclusions

This paper proposes a feature-based approach for establishing robust part-level compatibility with applications in interactive 3D reassembly. Each part is described by a set of local descriptors that generate candidate pair-wise matches between parts. We employ spectral techniques that enforce global geometric agreement to eliminate false matches. The algorithm is robust to noise and scales well to large databases containing the aggregated parts from multiple broken objects. In future work, we plan to integrate our algorithm into a system where noisy 3D scans of broken objects can be assembled under the interactive high-level guidance of a user.

Acknowledgments

The authors would like to thank L. Mummert, M. Satyanarayanan and S. Schlosser for useful feedback on this research.

References

[1] M. Ankerst, G. Kastenmuller, H. Kriegel, and T. Seidl. 3D shape histograms for similarity search and classification in spatial databases. In *Proc. Intl. Symp. Advances in Spatial Databases*, 1999.

[2] S. Berchtold, D. Keim, and H. Kriegel. Using extended feature objects for partial similarity retrieval. *Intl. Journal of VLDB*, 6(4), 1997.

[3] B. Bustos, D. Keim, D. Saupe, T. Schreck, and D. Vranic. Feature-based similarity search in 3D object databases. *ACM Computing Surveys*, 37(4), 2005.

[4] J. Cohen. Computational procedures for extracting landmarks in order to represent the geometry of a sherd. *Undergraduate thesis*, Brown University, 2002.

[5] C. Faloustos. Searching multimedia databases by content. *Kluwer Academic Publishers*, 1996.

[6] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *CACM*, 24(6), 1981.

[7] H. Freeman and L. Garder. A pictorial jigsaw puzzle: the computer solution of a problem in pattern recognition. *IEEE Trans. Elect. Comp.* , 13(2), 1964.

[8] D. Goldberg, C. Malon, and M. Bern. A global approach to automatic solution of jigsaw puzzles. In *Proc. Conf. Computational Geometry*, 2002.

[9] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[10] K. Hori, M. Imai, and T. Ogasawara. Joint detection for potsherds of broken earthenware. *CVPR*, 1999.

[11] Q. Huang, S. Flory, N. Gelfand, M. Hofer, and H. Pottman. Reassembling fractured objects by geometric matching. *SIGGRAPH*, 2006.

[12] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Trans. PAMI*, 21(5), 1999.

[13] M. Kazhdan, B. Chazelle, D. Dobkin, T. Funkhouser, and S. Rusinkeiwicz. A reflective symmetry descriptor for 3D models. *Algorithmica*, 38(1), 2003.

[14] D. Koller and M. Levoy. Computer-aided reconstruction and new matches in the Forma Urbis Romae. *Bulletino Della Commissione Archeologica Communale di Roma*, to appear.

[15] W. Kong and B. Kimia. On solving 2D and 3D puzzles using curve matching. *CVPR*, 2001.

[16] I. Laptev and T. Lindeberg. Space-time interest points. *ICCV*, 2003.

[17] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. *ICCV*, 2005.

[18] S. McDanels, B. Mayeaux, T. Collons, G. Jerman, R. Piascik, R. Russell, and S. Shah. An overview of the space shuttle Columbia accident from recovery through reconstruction. *Failure Analysis and Prevention*, 6(1), 2006.

[19] G. Papaioannou, E. Karabassi, and T. Theoharis. Reconstruction of three-dimensional objects through matching of their parts. *IEEE Trans. PAMI*, 24(1), 2002.

[20] J. Stolfi and H. Leitão. A multiscale method for the reassembly of two-dimensional fragmented objects. *IEEE Trans. PAMI*, 24(9), 2002.

[21] M. Teodoro, G. Phillips, and L. Kavraki. Molecular docking: A problem with thousands of degrees of freedom. *ICRA*, 2001.

[22] G. Ucoluk and I. Toroslu. Automatic reconstruction of broken 3-D surface objects. *Computers and Graphics*, 23(4), 1999.

[23] A. Willis and D. Cooper. Bayesian assembly of 3D axially symmetric shapes from fragments. *CVPR*, 2004.

[24] S. Winkelbach, M. Rilk, C. Schonfelder, F. Wahl. Fast random sample matching of 3D fragments. *Pattern Recognition, LNCS*, 3175, 2 2004.

[25] H. Wolfson, E. Schonberg, A. Kalvin and Y. Lambdan. Solving jigsaw puzzles by computer. *Annals of Operations Research*, 12(1), 1988.