Relational Indexing

Mauro S. Costa[1] and Linda G. Shapiro[1,2] *

[1] Department of Electrical Engineering, Box 352500
[2] Department of Computer Science & Engineering, Box 352350
University of Washington
Seattle WA 98195
U.S.A.

Abstract. We address the problem of recognizing 3D objects in scenes containing multiple objects by means of a new indexing technique called *relational indexing*. Given a database of relational models, we determine those models whose relational descriptions are most similar to subsets of the relational description of an unknown scene. The relational indexing technique has worst-case complexity $O(m\binom{k}{2})$ for relational graphs of up to k nodes and a database of m models. This paper evaluates the performance of the technique using Monte Carlo experiments.

1 Introduction

In a model-based object recognition system, the task of matching image features to model features, in the general case, implies searching the space of all possible correspondences. Indexing is one of the techniques that have been largely utilized to reduce this search space. In recent years, several systems have made use of different approaches to indexing ([1], [4], [5], [8]). In this paper we describe *relational indexing*: a new approach to indexing into a database of models that makes use of features and the spatial relationships among them. In this new matching technique each model in the database is described by a relational graph of all its features, but small relational subgraphs of the image features are utilized to index into the database and retrieve appropriate model hypotheses. For a database of m models and for relational graphs of up to k nodes the algorithm has worst-case complexity $O(m\binom{k}{2})$. This paper investigates the use of this new technique in a model-based 3D recognition system; Monte Carlo experiments are used to evaluate its performance as a hypotheses generation mechanism.

2 Relational Indexing Notation

An *attributed relational description* D is a labeled graph $D = (N, E)$ where N is a set of attributed nodes and E is a set of labeled edges. For each attributed

---

* This research was supported by the National Science Foundation under grant number IRI-9023977, by the Boeing Commercial Airplane Group, and by the Washington Technology Center.

node $n \in N$, let $A(n)$ denote the attribute vector associated with node n. Each labeled edge $e \in E$ will be denoted as $e = (n_i, n_j, L_{i,j})$ where n_i and n_j are nodes of N and $L_{i,j}$ is the label associated with the edge between them. $L_{i,j}$ is usually a scalar, but it can also be a vector.

A relational description $D = (N, E)$ can be broken down into subgraphs, each having a small number of nodes. We will consider subgraphs of two nodes, called ***2-graphs.*** All of our graphs are complete graphs, so a graph of k nodes has $\binom{k}{2}$ 2-graphs, each consisting of a pair of attributed nodes and the labeled relationship between them. The relationship between the two nodes may be a meaningful spatial relationship or the null relationship *none*. We will refer to the set of 2-graphs of a relational description D_l as T_l. Figure 1 illustrates a partial graph representing an object and all the 2-graphs for the given relational graph.

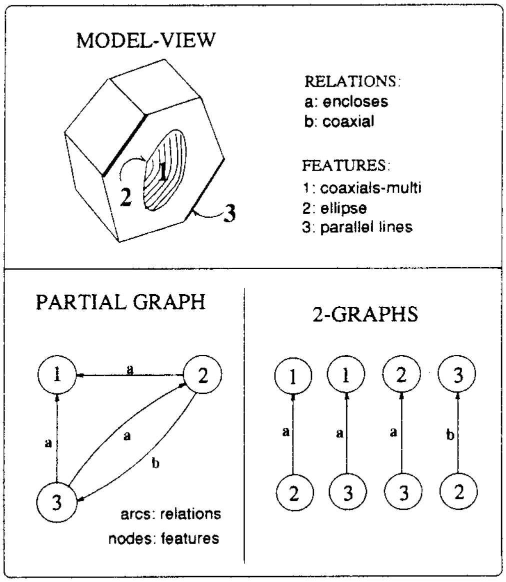


Fig. 1. Sample graph and corresponding 2-graphs for the "hexnut" object.

3 Relational Indexing Algorithm

Let $DB = \{M_1, M_2, \ldots, M_m\}$ be the database of models, where each $M_i = (N_i, E_i)$ is an attributed relational description. Let $D = (N, E)$ be a relational description that has been extracted from an image and T be the set of all 2-graphs of D. We would like to find the closest models to D. This is accomplished in two steps: an off-line preprocessing step to set up the indexing mechanism and an on-line hypotheses generation step. The off-line step is as follows. Let T^{M_i} be the set of 2-graphs of M_i. Each element $G_l^{M_i}$ in this set is encoded to produce an index $I_l^{M_i}$, which is used to access a hash table. The bin corresponding to the particular encoded 2-graph $G_l^{M_i}$ stores information about which model M_i gave rise to that particular index. This encoding and storing of information in the hash table is done off-line and for all models in the database DB.

In the on-line step the relational indexing procedure keeps an accumulator A_i for each model M_i in the database (all the accumulators are initialized to zero). Each 2-graph G_l in T is encoded to produce an index I_l. The procedure then uses that index to retrieve from the precomputed hash table all models M_i that contain a 2-graph that is identical to G_l. Identical means that the two nodes have the same attributes and the edge has the same label. For each 2-graph G_l of T, the accumulator A_i of every retrieved model M_i is incremented by one. After the entire voting process, the models whose accumulators have the highest votes are candidates for further consideration. Since the procedure goes through all $\binom{k}{2}$ 2-graphs of T and for each one can retrieve a maximum of m models, the worst-case complexity is $O(m\binom{k}{2})$. However, the work performed on each model is very small, merely incrementing its accumulator by one. This is very different from methods that perform full relational matching on each model of the database. The relational indexing algorithm is given below.

RELATIONAL INDEXING ALGORITHM

Preprocessing (off-line) Phase

1. For each model M_i in the database DB do:
   - Encode each 2-graph $G_l^{M_i}$ to produce an index.
   - Store M_i and associated information in the selected bin of the hash table.

Matching (on-line) Phase

1. Construct a relational description D for the scene.
2. For each 2-graph G_l of D do:
   - Encode it, produce and index, and access the hash table.
   - Cast a vote for each M_i associated with the selected bin.
3. Select M_i's with enough votes as possible hypotheses.

Since some models share features and relations, it is expected that some of the hypotheses produced will be incorrect. This indicates that a subsequent verification phase is essential for the method to be successful. It is important to mention that the information stored in the hash table is actually more than just the identity of the model that gave rise to a particular 2-graph index. It also contains information about which specific features (and their attributes) are part of the 2-graph. This information is essential for hypothesis verification and eventual pose estimation.

4 Matching with Relational Indexing: an Example

In this section we give an example of the experiments we have conducted to demonstrate the use of the relational indexing technique for 3D object recognition with appearance-based features [2].

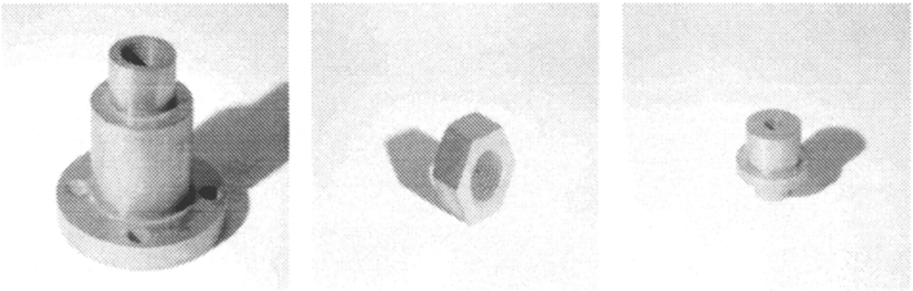
An *appearance-based model* of an object is defined as the collection of the features that can be reliably detected from a training set of real images of the object. We chose to use *view-class models*, in which an object is represented by a small set of characteristic views, each having its own distinct feature set [6]. We have created a database of appearance-based object models for a set of mechanical parts that have both flat and curved surfaces, holes, and threads. The relational descriptions $D_{V,M}$ of all the model-views were derived from a large set of training pairs of real images (280 image pairs of 7 models).

In order to illustrate our recognition methodology, we matched nine test images of both single and multiple object scenes to the database of model-views. The nine test images used are shown in figure 2. The database of models was created by encoding all 2-graphs for each of the model-views. For each test scene, features and relations were detected, the relational description was built, and all 2-graphs were encoded. Relational indexing was then performed and the generated hypotheses were normalized by the number of 2-graphs in the original models and ranked in order of strength. Hypotheses that exceeded a preset strength threshold were dubbed "strong hypotheses." These hypotheses are to be passed to the verification procedure for further consideration.

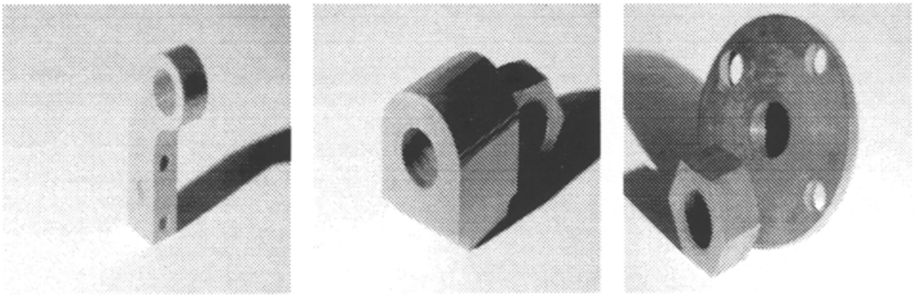
In each of the nine tests, the strong hypotheses were classified as type A, type B, or type C. Type A hypotheses are those where the correct model and the correct (closest) view class were identified. Type B hypotheses are those where the correct model was identified, but the chosen view class was not closest to the view in the image. Type B hypotheses can still be verified and used to determine pose if enough corresponding features are found. Type C hypotheses are those where an incorrect model was selected. These incorrect hypotheses should be ruled out in the verification step. The results of the nine tests are as follows: all the objects in the scenes have been correctly recognized (18 type A hypotheses); there were 9 type B hypotheses, and 4 type C hypotheses.

Figure 3(a) shows the results for test scene 9, which contains four objects: the "stacked cylinder," the "hexnut," the "wrench," and the "cylinder-block." The system produced five strong hypotheses; four were correct and are overlaid on the image. These hypothesized models were taken through pose computation (affine correspondence of appearance-based model features and scene features) without verification. The fifth strong hypothesis (not shown) matched the object "hexnut" to an incorrect view of the correct object model. The subgraph indices shown in Figure 1 were among those that were used in the matching process.

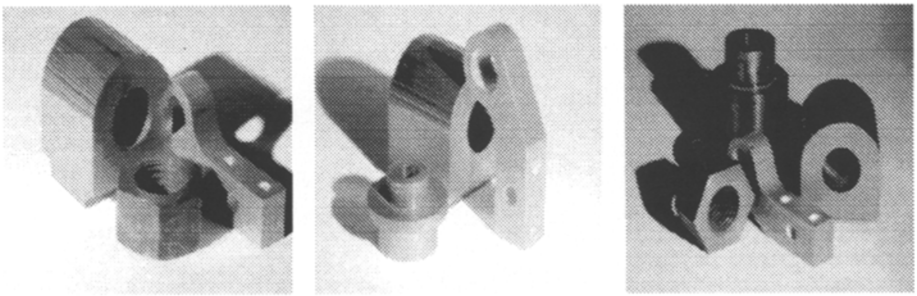
Figure 3(b) illustrates the correct (type A) hypotheses generated for test scene 5. Of the three type B hypotheses generated, one was for the "cylinder-block" object and two were for the "hexnut" object, both of which are present in the scene. As seen, the method shows promising results. However, a more



(a) Image 1 (left) (b) Image 2 (right) (c) Image 3 (left)



(d) Image 4 (left) (e) Image 5 (left) (f) Image 6 (right)



(g) Image 7 (left) (h) Image 8 (right) (i) Image 9 (right)

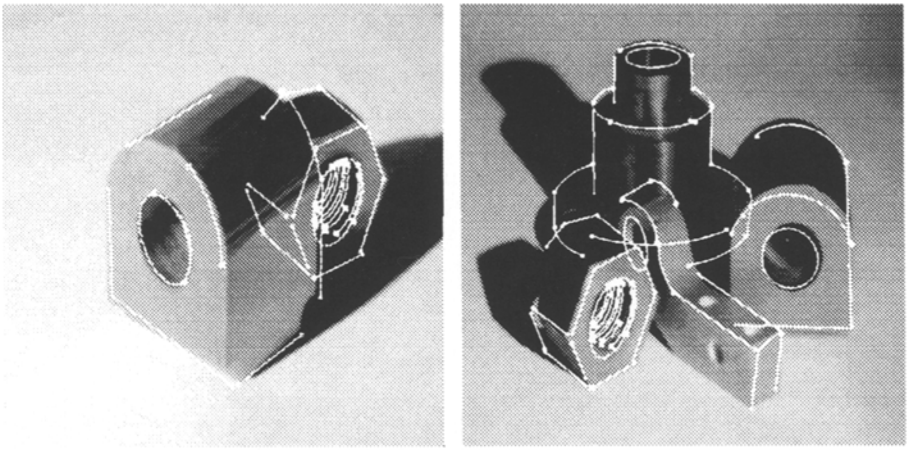
Fig. 2. The nine test scenes used.

thorough characterization of the indexing mechanism performance needs to be assessed. The next section addresses this issue.

5 Monte Carlo Experiments

We performed Monte Carlo experiments to investigate how our relational indexing algorithm performs as the following factors are varied:

1. Number of model graphs in the database.



(a) (b)

Fig. 3. (a) Left image of test scene 5 overlaid with the appearance-based features of the hypothesized model matches. The objects in this scene are the "cylinder-block" and the "hexnut." (b) Right image of test scene 9 overlaid with the appearance-based features of the hypothesized model matches. The objects in this scene are the "stacked cylinder," the "hexnut," the "wrench," and the "cylinder-block."

2. Average size of model graphs in the database.
3. Size of image graph, measured in terms of the percentages of extra and missing 2-graphs from a given model graph.
4. Size of cluster of similar models in the database.
5. The degree of similarity between models.

In order to define similarity between model graphs, let $DB = M_1, M_2, ..., M_m$ be a database of m model graphs. Consider the set of 2-graphs T_i of each model M_i. Let $s(i,j) = |Ti \cap Tj|$ be the measure of similarity between models M_i and M_j. If $s(i,j) = 0$, then M_i and M_j are fully discriminable by relational indexing.

Synthetic data was obtained by generating random databases of models graphs whose nodes and edges were features and relations used in our current system. In order to generate realistic databases, the physical constraints between the features and their relations were taken into account. For each test database, each model was transformed into an "image graph" by randomly removing 2-graphs and adding extra 2-graphs and was then matched against the database of models. A model was retrieved only if at least 50% of its 2-graphs were found in the image. For each such set of image graphs used, we recorded the following quantities:

- **PCM:** The average percentage of correct models retrieved, with respect to the total number of models retrieved
- **PHV:** The average percentage of correct models retrieved *with the highest vote*, with respect to the total number of models retrieved

- **PMR:** The average percentage of models retrieved, with respect to the total number of models in the database

Since we were interested in investigating the effect of similar models in the database on the performance of the technique, the above quantities were measured within and outside a cluster of similar models.

For each experiment described above, 100 random replications ($R = 100$) were performed. Each database generated possessed one cluster of similar models. All the parameters involved in this investigation are listed below, along with the individual values each assumed:

- D: Number of models in the database = 50, 100, 200, 500
- G: Average size of model graphs (measured in terms of 2-graphs) = 10, 15, 20, 25, 30.
- M: Percentage of missing 2-graphs = 0, 20, 50.
- E: Percentage of extra 2-graphs = 0, 20, 50.
- S: Similarity among the models in the cluster (measured as a percentage of G) = 20, 30, 40.
- C: Cluster size (measured as a percentage of D) = 10, 20, 30

The total number of experiments performed was then: $R \times D \times G \times M \times E \times S \times C = 162,000$. We generated plots of PCM, PHV, and PMR as a function of the average size model in the database (measured in terms of 2-graphs), for every combination of the following parameters: D, S, C, M, and E. Given the large number of plots obtained, we only illustrate those for which the trends in performance are most significant (for the complete set of results, please see [3]).

PCM, the first quantity measured, for both within the cluster and outside the cluster, had value 100% irrespective of the combination of the parameters involved. This means that the correct model was *always* among the retrieved models. Since relational indexing is a hypotheses generation mechanism, verification has to be performed for each model retrieved. In our current system, we start the verification with the models that obtained the largest number of votes, therefore we are also interested in finding out how often the correct model received the most votes. Quantity PHV reflects this aspect of the performance of the technique.

For PHV, measured within the cluster, the plots of Figure 4 are representative of the trends in performance. From 4(a), (b), and (c), it can be seen that the average percentage of correct models retrieved decreases with an increase in the percentage of missing 2-graphs and in the size of the database. This is expected since the total number of models retrieved will increase with the size of the database.

Figures 4(d), (e), and (f) show plots of PHV for different values of similarities within the cluster. The trend observed is that performance only changes significantly for small values of the average size models and with the percentage of missing 2-graphs, when the similarity between models is varied. The effects of the size of the cluster of similar models on performance can be seen on the plots of Figures 4(g), (h), and (i). Again, performance decreases with an increase in the cluster size and in the percentage of missing 2-graphs.

One important thing to notice is that regardless of the combination of parameters used, the number of extra 2-graphs had a negligible effect on the performance, indicating that the indexing technique is robust with respect to clutter.

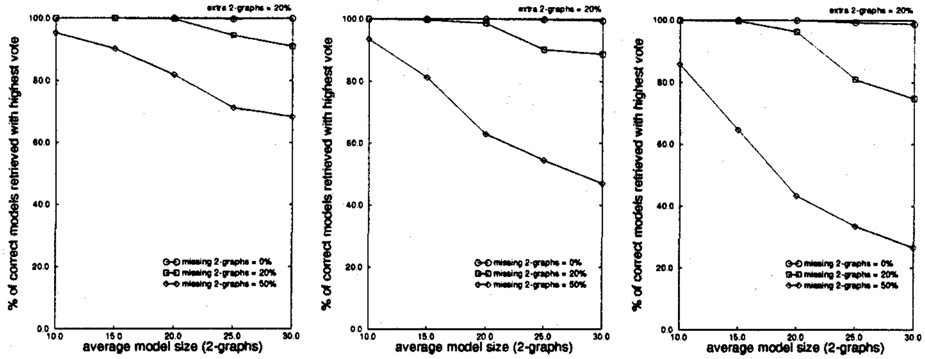
Quantity PHV measured outside the cluster had a much more stable behavior than for inside the cluster. In fact, the plots obtained are essentially flat, regardless of the combination of parameters used. The only parameters that slightly affect the performance are again the percentage of missing 2-graphs and the size of the database. Since outside the cluster the models are randomly different, the value of PHV is always very close to 100%, except for a larger percentage of missing 2-graphs (50%), for which it decreases to around 85% for the largest size database (500 models).

Figure 5 shows some of the results obtained when measuring quantity PMR, within the cluster. It can be seen that the average percentage of total models retrieved remains very low, regardless of the size of the database, according to the plots in Figures 5(a), (b), and (c). The effects of changes in similarities among models in the cluster are depicted in Figures 5(d), (e), and (f). As expected, with an increase in the similarity between models, there is an increase in the percentage of total models retrieved. The larger the cluster size, the larger the average percentage of models retrieved will be. This trend can be observed in Figures 5(g), (h), and (i).

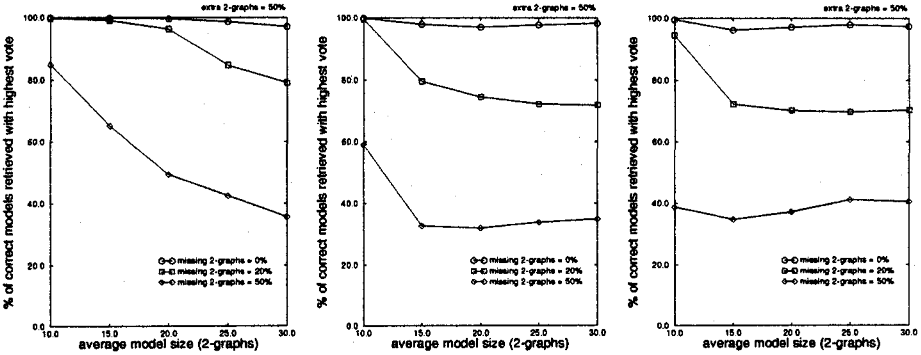
As for the case of PHV, the percentage of extra 2-graphs has close to no affect in the performance of the technique in terms of the measured quantity PMR. This also holds for outside cluster measurements. Quantity PMR measured outside clusters, stays fairly constant (ranging from around 1% to 3% of the database size), regardless of the actual values of the parameters involved in the investigation.

References

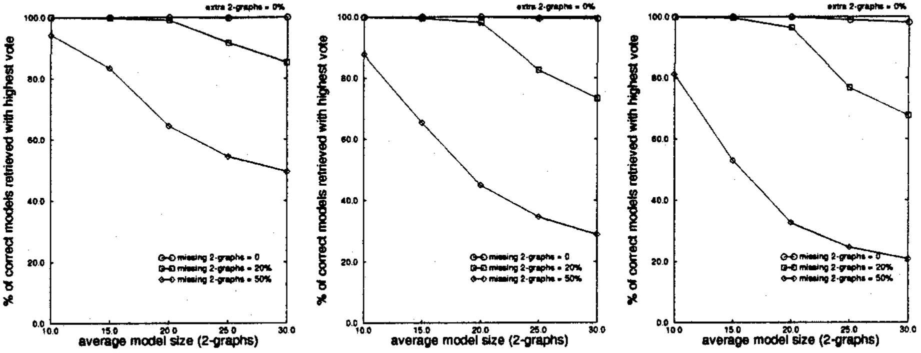
1. A. Califano and R. Mohan. Multidimensional Indexing for Recognition of Visual Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):373–392, 1994.
2. M. S. Costa and L. G. Shapiro. Scene Analysis Using Appearance-Based Models and Relational Indexing. In *International Symposium on Computer Vision*, pp. 103–108, Coral Gables, Florida, November 1995.
3. M. S. Costa and L. G. Shapiro. Relational Indexing for Object Recognition. *Intelligent Systems Lab Technical Report #ISL-05-96*, Department of Electrical Engineering, University of Washington, May 1996.
4. Y. Lamdan and H. J. Wolfson. Geometric Hashing: A general and efficient Model-based Recognition Scheme. In *Second International Conference on Computer Vision*, pp. 238–249, 1988.
5. C. F. Olson. Probabilistic Indexing for Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):518–522, 1995.
6. L. G. Shapiro and M. S. Costa. Appearance-Based 3D Object Recognition. In *Proc. of the NSF/DARPA Workshop on 3D Object Representation for Computer Vision*, New York, NY, December 1994.
7. L. G. Shapiro and R. M. Haralick. A Metric for Comparing Relational Descriptions *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7,pp. 90-94, 1985.
8. F. Stein and G. Medioni. Structural Indexing: Efficient Three Dimensional Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):125–145, 1992.



(a) D=50, S=20, C=10 (b) D=100, S=20, C=10 (c) D=200, S=20, C=10

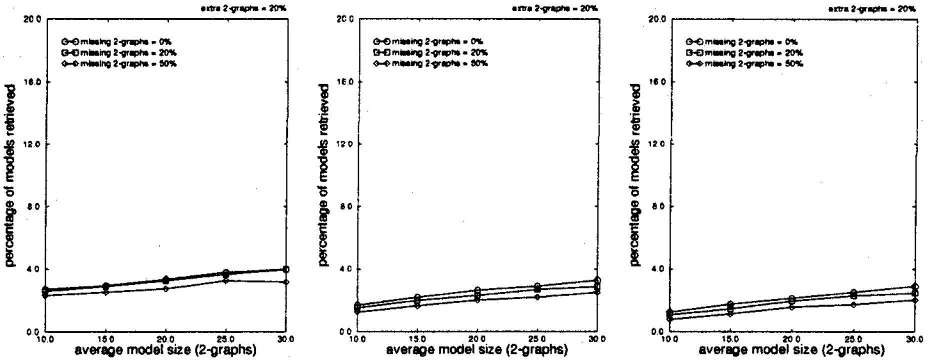


(d) S=20, D=50, C=30 (e) S=30, D=50, C=30 (f) S=40, D=50, C=30

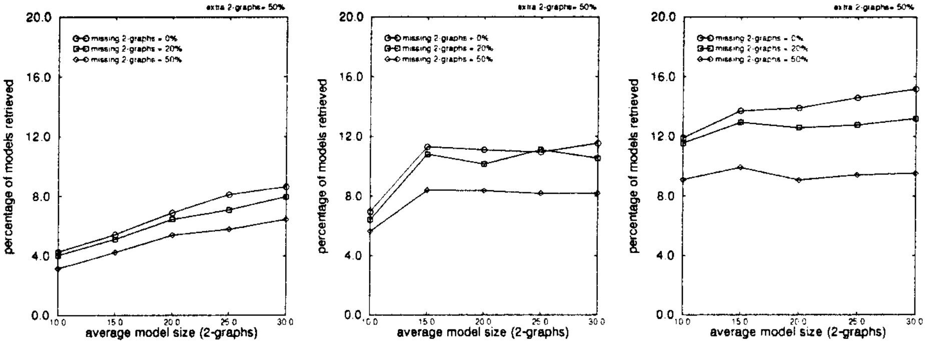


(g) C=10, D=100, S=20 (h) C=20, D=100, S=20 (i) C=30, D=100, S=20

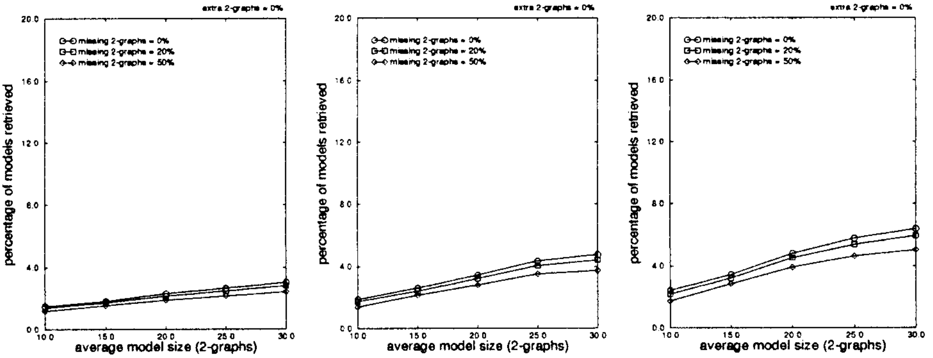
Fig. 4. Sample results for quantity PHV (average percentage of correct models retrieved with the highest vote) measured within the cluster of models in the database.



(a) D=50, S=20, C=10 (b) D=100, S=20, C=10 (c) D=200, S=20, C=10



(d) S=20, D=50, C=30 (e) S=30, D=50, C=30 (f) S=40, D=50, C=30



(g) C=10, D=100, S=20 (h) C=20, D=100, S=20 (i) C=30, D=100, S=20

Fig. 5. Sample results for quantity PMR (average percentage of total models retrieved) measured within the cluster of models in the database.