ESANN'2005 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 27-29 April 2005, d-side publi., ISBN 2-930307-05-6.

Chemical Similarity Searching Using a Neural Graph Matcher

Stefan Klinger and Jim Austin *

Advanced Computer Architectures Group - Department of Computer Science
Heslington, York, YO10 5DD - UK

Abstract. A neural graph matcher based on Correlation Matrix Memories is evaluated in terms of efficiency and effectiveness against two maximum common subgraph (mcs) algorithms. The algorithm removes implausible solutions below a user-defined threshold and runs faster than conventional mcs methods on our database of chemical graphs while being slightly less effective.

1 Introduction

The purpose of chemical similarity searching is to find the most similar chemical graphs in a large database to a molecular graph that is known to exhibit a certain activity [1]. This assertion is based on the structure-similarity principle [2] which states that two structurally-similar molecules are likely to exhibit a similar activity. The term activity is commonly referred to a property of a molecule that enables it to inhibit or alter the functions of proteins. Molecular similarity searching is an integral part of the early stages of the drug discovery process [3]. The idea is to take an active molecule and to find other compounds with similar structure in a database. Usually, a certain fraction of the most similar molecules are considered for further testing. Corporate-sized databases contain millions of compounds and the number is likely to grow in the future. Therefore, efficient methods are required to search these databases.

Many similarity search methods are based on the popular fingerprinting technique. Here, the properties of molecules are encoded as bits on fixed-sized bitstrings. The bitstrings are then compared using a suitable distance metric [4]. Graph matching methods have also been applied in this area [5, 6]. Most of the chemoinformatics literature uses a maximum common subgraph (mcs) algorithm. A common subgraph S is a graph that is isomorphic to a subgraph in graphs G_1 and G_2. A common subgraph is called a mcs if there exists no other common subgraph of G_1 and G_2 that has more vertices than S. The mcs problem is then transformed into a maximum-clique algorithm of an association graph [7]. We propose to present a neural graph matching algorithm. This algorithm is based on the Relaxation Labelling technique [8] and was implemented on Correlation Matrix Memories (CMMs) [9]. We compare its efficiency and effectiveness against the Bron-Kerbosch [10] and RASCAL [6] algorithms using a sample database of chemical graphs.

*Work funded By EPSRC Studentship GR/P03292/01

2 Algorithm

2.1 Definitions and Terminology

The graphs referred to in the following text are assumed to be undirected attributed relational graphs (ARGs). The ARG is represented by $G = (V, E, X, Y)$ and consists of a set of vertices $V = (v_1, v_2..., v_{|V|})$ which represent the objects in the model. The set $E = (e_1, e_2..., e_{|E|})$ is the set of graph edges, and these represent the presence of a relationship of some sort between a pair of objects. A set of vertex attributes is available on the objects in the graph that is denoted by $X = (x_1, ..., x_{|V|})$. A set of edge attribute $Y = (y_1, ..., y_{|E|})$ defined on the relations is available as an additional source of information. The neighbourhood of a vertex v_i is its set of adjacent vertices that is denoted by $N(v_i)$.

2.2 Relaxation By Elimination

Our algorithm is based on an optimisation technique, called Relaxation Labelling [8]. The general idea behind relaxation labelling is to update individual mappings for graph vertices based not only on their feature measurements, but also by combining contextual evidence from their spatial environment. Discrete relaxation [11] visits each vertex in turn and updates the label on that vertex in order to gain maximum improvement in the matching criterion of the problem. Probabilistic Relaxation [12] assigns probabilities to each label in the set giving an estimate of the likelihood that the particular label is correct one for that feature. It then tries to maximise the probabilities iteratively taking into account the probabilities associated with neighbouring features.

Unlike discrete and probabilistic relaxation, relaxation by elimination (RBE) [13] initially keeps all plausible solutions and iteratively removes implausible solutions below a defined similarity threshold. Consider a query graph G_q and a model graph G_m. A vertex v_q of the query graph G_q is assigned a set of candidates $c_q = (c_{q1}, c_{q2}, ..., c_{q|c_q|}, c_{qx} \in V_m, x = 1, ..., |c_q|)$ that correspond to the currently plausible mappings from a query graph vertex to a set of model graph vertices. We denote the set of all vertex correspondences by $C = (c_1, c_2, ..., c_{|V_q|})$.

In the initial stage, each query graph vertex is allocated a set of feasible model graph vertices based on the unary attributes they have in common. To put it more precisely, we add model vertex v_m to the set of query vertex candidates v_q if $||x_q - x_m|| < \epsilon$, where $v_q \in V_q$ and $v_m \in V_m$.

During the relaxation process, the sets of candidates are pruned iteratively until a stopping criterion is met. We introduce a support function $S(c_{qi}) = \sum_{j=1}^{N(v_q)} h(c_{qi}, c_j)$ for candidate i of query vertex q that counts the number of query neighbour vertices that have candidates which support the current assignment c_{qi}. The consistency measure $h(c_{qi}, c_j)$ is a discrete quantity and verifies a satisfied local constraint. It is defined as

$$h(c_{qi}, c_j) = \begin{cases} 1, & \text{when } ||y_{qj} - y_{c_{qi}c_j}|| < \epsilon \\ 0, & \text{otherwise} \end{cases}$$

The candidates c_{qi} that have a support count below a defined threshold $S(c_{qi}) < \lambda$ are subsequently removed. This process assumes that one knows *a priori* a suitable similarity threshold λ for a given graph matching problem. In the present study, we have pursued two strategies. The first sets the threshold as a fraction of the query graph size, while the second approach keeps a percentage of the candidates with the highest support. We denote these strategies by Threshold Willshaw [9] and Threshold Lmax [14], respectively.

2.3 Neural Architecture

The process described above has the potential of fast and efficient implementation using an architecture of inter-connected Correlation Matrix Memories (CMMs) [9]. A CMM is a simple binary associative neural network that offers quick training and highly flexible and fast search capability. The CMM has been used as a match engine in a number of successful applications, e.g. symbolic reasoning in the AURA (Advanced Uncertain Reasoning Architecture) approach [15] and post code matching.

The list of candidates c_q can be represented as a binary array. Furthermore, if measurements are discretised, then the support function can be executed through the use of bitwise operations on binary arrays. The bit vector of current candidates c_q of query vertex v_q is used as an input to the processing of evidence counts for candidates $c_{N(v_q)}$ of adjacent query vertices $N(v_q)$. This process can be performed for each query node candidate list c_q in parallel. The use of CMMs also allows the sharing of rows in memory by multiple binary patterns. This enables efficient use of memory at the expense of introducing false positives. By superimposing the set of vertices from more than one model graph in the candidate list c_q, multiple graph correspondences can be matched in parallel. In order to keep the number of false positives to a minimum, we ensure that all single vertex patterns are orthogonal to each other.

2.4 Complexity

The maximum common subgraph, maximum clique and subgraph isomorphism problems are all known to be NP-complete [16]. All of the maximum-clique algorithms are optimal methods which means they will experience exponential order of time growth in the worst case. However, they might be much faster in the average case. For example, Wilf [17] has shown that the maximum independent set problem has sub-exponential time complexity of $O(n^{logn})$ in the average case. The maximum-clique algorithms used in this study are all branch and bound methods, i.e. they prune branches of the search tree if they cannot possibly exhibit a better solution than the current best. Estimating the size of these search trees is too difficult analytically [18] because of the sheer number of combinations to consider. Conventional relaxation procedures replace the problem with a polynomial-time algorithm, however, this guarantees to find solutions that are only locally optimal. Relaxation By Elimination has a worst-case time complexity of $O(|V_q|^2|V_m|^2)$ [13].

3 Simulations

We propose a set of experiments to verify the efficiency of our Neural Graph Matcher (NGM) in comparison to a standard maximum clique algorithm [10]. The second algorithm [6] is a more recent development and employs a fast initial screening stage where implausible graphs with a similarity below a minimum similarity index are eliminated. A more rigorous maximum common edge subgraph procedure is applied on the remaining graphs. The tests were conducted on the P38 data set of molecular structures supplied by Evotec OAI (http://www.evotecoai.com).

Data Set	No. of Targets	No. of Molecules	No. of Comparisons	Avg $\|V\|$	Std. Dev $\|V\|$
P38	102	10,102	1,030,404	23.01	6.33

Table 1: Data Set

There exist two common approaches of presenting molecular graphs for similarity searching. The first approach represents the vertices as the atoms and connects vertices in the graph if there exists a bond between them. A different procedure connects all atom pairs and labels the edges with the number of bonds separating them in the shortest path. Note that the mcs determined by both methods is not necessarily equivalent. We refer to the two approaches as Bond Types (BT) and Topological Distances (TD), respectively. In addition, we have chosen the atom types as vertex labels.

The algorithms were implemented in C++ using gcc 3.2.2 and executed on a Ultra Sparc III Cu 900Mhz machine running Solaris 9. A time limit of 24 hours was set for each trial. The total execution times are shown in Table 2.

Test Sets	[10]	RASCAL			NGM Willshaw			NGM Lmax		
		0.6	0.7	0.8	0.6	0.7	0.8	1%	5%	10%
BT	>24h	>24h	18,792	160	12,073	15,782	11,377	2,825	3,050	3,433
TD	9,763	n/a	n/a	n/a	5,265	4,374	3,628	4,175	4,345	4,801

Table 2: Execution times in seconds

To compare the effectiveness of the methods, we use the Guner-Henry (GH) score [19] based on the precision (P) and recall (R) of the search [4]. Precision is defined as the fraction of the active structures retrieved (a) over the number of structures retrieved, i.e. $P = \frac{a}{n}$. Recall is defined as the fraction of retrieved active structures (a) over the total number of active structures in the database (A), i.e. $R = \frac{a}{A}$. We apply threshold values of $R \geq 0.05$ and $P \geq 0.5$ because lower values represent unacceptable levels of performance. Queries resulting in precision or recall value below these thresholds are removed from consideration and are referred to as discards (D). The mean precision and recall values for the resulting set above the similarity thresholds are depicted in table 3.

Algorithm	Test	P	R	D
NGM Willshaw 0.6	TD	0.816	0.099	74
NGM Willshaw 0.7	TD	0.824	0.082	83
RASCAL 0.7	BT	0.850	0.068	93
NGM Willshaw 0.8	TD	0.905	0.107	94
NGM Willshaw 0.5	TD	0.692	0.105	94

Table 3: Precision and recall values of non-discarded queries

We now rank the structures in the result set by applying a suitable similarity metric. Here, we have chosen the score of Wallis *et.al.* [20] which is defined as $d(G_1, G_2) = \frac{|G_{12}|}{|G_1|+|G_2|-|G_{12}|}$. Once the pair-wise similarities between each target structure and each member of the database have been calculated, the similarity values are sorted in order of decreasing similarity. We then determine the GH score for every structure in the ranking, until the graph with the maximum GH score as well as precision and recall values above the thresholds is determined. The GH score [19] is defined as

$$GH = \left(\frac{a(3 \cdot A + n)}{4 \cdot n \cdot A}\right)\left(1 - \frac{n - a}{N - A}\right) \tag{1}$$

The position in the ranking corresponding to the maximum GH score is used as the cut-off point and subsequent structures in the list are removed from further consideration. The mean precision and recall values over the remaining sets of 102-D non-discarded queries are shown in table 4.

Algorithm	Test	P	R	D	Algorithm	Test	P	R	D
Bron-Kerbosch	TD	0.868	0.073	47	NGM W. 0.7	TD	0.921	0.074	80
NGM L. 1 %	TD	0.864	0.075	60	NGM L. 1 %	BT	0.633	0.073	86
NGM W. 0.6	TD	0.879	0.070	61	RASCAL 0.7	BT	0.892	0.064	88
NGM L. 5 %	TD	0.794	0.071	65	NGM L. 5 %	BT	0.642	0.070	90
NGM L. 10 %	TD	0.767	0.064	74	NGM W. 0.8	TD	0.927	0.087	93
NGM W. 0.5	TD	0.891	0.074	76	NGM L. 10 %	BT	0.690	0.064	92

Table 4: Maximum GH score retrieval results

4 Discussion

This study showed that the proposed NGM is an efficient and versatile tool for chemical similarity searching. Unlike the maximum clique algorithms that are only efficient in the case when the association graph is sparse, or the RASCAL algorithm that takes advantage of the sparseness of the chemical graphs and its edge labels, our NGM can be applied efficiently to a wider range of labelled graphs. The tests also indicated that our NGM seems to be less effective than the Bron-Kerbosch [10] algorithm for the given vertex and edge features. This is probably due to the ambiguous vertex correspondences that were not removed by the relaxation process. However, these results seem to be encouraging and we want to apply the NGM to a wider range of data sets using a variety of different structural features.

Acknowledgements

The project was undertaken as an EPSRC CASE studentship no GR/P03292/01 with the support of Evotec OAI.

References

[1] P. Willet. Chemical similarity searching. *J. Chem. Inf. Comput. Sci.*, 38:983–996, 1998.

[2] A.M. Johnson and G.M. Maggiora. *Concepts and Applications of Molecular Similarity.* Wiley, New York, 1990.

[3] R.P. Sheridan. Why do we need so many chemical simiarity search methods? *Drug Discovery Today*, 7(17):903–911, 2002.

[4] J.W. Raymond and P. Willet. Effectiveness of graph-based and fingerprint-based similarity measures ... *J. of Comput.-Aided Mol. Des.*, 16:59–71, 2002.

[5] A.T. Brint and P. Willet. Algorithms for the identification of three-dimensional maximal common substructures. *J. Chem. Inf. Comput. Sci.*, 27:152–158, 1987.

[6] J.W. Raymond, E.J. Gardiner, and P. Willet. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *Comput. J.*, 45:631–644, 2002.

[7] H. Barrow and R. Burstall. Subgraph isomorphism, matching relational structures and maximal cliques. *Inf. Proc. Lett.*, 4:83–84, 1976.

[8] A. Rosenfeld, R.A. Hummel, and S.W. Zucker. Scene labelling by relaxation operations. *IEEE Trans. Systems, Man. and Cybernet.*, 13:353–362, 1983.

[9] D.J. Willshaw, O.P. Buneman, and H.C. Longuet-Higgins. Non-holographic associative memory. *Nature*, 222:960–962, 1969.

[10] C. Bron and J. Kerbosch. Finding all cliques of an undirected graph. *Comm. ACM*, 16(9):575–577, 1973.

[11] D.L. Waltz. *Understanding Line Drawings of Scenes with Shadows.* McGraw-Hill, New York, 1975.

[12] W.J. Christmas, J. Kittler, and M. Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE Trans. Pattern Anal. Machine Intell.*, 17:353–362, 1995.

[13] M. Turner and J. Austin. Graph matching by neural relaxation. *Neural Computing and Applications*, 7:238–248, 1997.

[14] D.P. Casasent and B.A. Telfer. High capacity pattern recoginition associative processors. *Neural Networks*, 5(4):251–261, 1992.

[15] J. Austin. Distributed associative memories for high-speed symbolic reasoning. *Fuzzy Sets and Systems*, 82:223–233, 1995.

[16] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, New York, 1979.

[17] H.S. Wilf. *Algorithms and Complexity.* Prentice-Hall, New Yersey, 1986.

[18] A. Levitin. *Introduction to the Design and Analysis of Algorithms.* Addison-Wesley, New York, 2003.

[19] O.F. Guner. *Pharmacophore Perception, Development and Use in Drug Design*, page 194. International University Line, La Jolla, CA, USA, 2000.

[20] W.D. Wallis, P. Shoubridge, M. Kraetz, and D. Ray. Graph distances using graph union. *Pattern Recogn. Lett.*, 22:701–704, 2001.