ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 107-112

An Associative Memory for the Automorphism Group of Structures

Brijnesh J. Jain, Fritz Wysotzki
Technical University Berlin, Computer Science Department,

Franklinstr. 28/29, 10587 Berlin, Germany

Abstract. We present an associative memory, which memorizes the automorphisms of any reducible labeled structure. After storing the automorphism group the associative memory provides an unlabeled presentation of the labeled structure under consideration. Recalling the retrieval states links the energy minimization process of neural networks with the theory of permutation groups.

1 Introduction

Three main computational problems in algebraic combinatorics are analytical and constructive enumeration of structures, testing isomorphism of graphs, and finding the automorphism group of graphs. These problems arise for example in isomer generation, in the identification of chemical objects in huge databases, and in the detection of symmetries of chemical graphs [5]. For all three problems no polynomial time algorithm is known. However, the above problems are closely related and their solutions are more or less conveniently obtained, once the automorphism group is known [5].

In this contribution we show that the Hopfield model can be used as an associative memory for storing the automorphism group of any discrete reducible structure σ defined on a point set X. In contrast to the conventional associative memory we want to store patterns, namely the automorphisms of σ, which are not known in advance. Nevertheless, to memorize the unknown automorphism group we store the *labeled* structure σ itself to obtain an *unlabeled* presentation of σ. We then show that the automorphisms of σ correspond to the global minima of the underlying energy function. After learning, the associative memory carries an inherent finite algebraic structure and links the neurodynamic energy minimizing process to the theory of permutation groups. Permutation groups are a basic concept in algebra and combinatorics. In various combinatorial problems group theoretical information is permanently used. The link to the theory of permutation groups facilitates access to mathematical results in algebra and combinatorics and may deliver a new insight into the potential capabilities of neural networks for solving combinatorial problems.

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 107-112

2 Species of structures

We embed discrete structures like graphs, linear codes, designs, or finite geometries in the theory of species, which is a modern way in algebraic combinatorics of formalizing the examination of generating functions for many labeled structures [1]. Besides basic definitions from the theory of species, we introduce novel concepts, namely the notion of *induced substructure*, *reducible species* and the *modular product* to facilitate a memory model for storing the automorphisms.

Discrete structures are built on a ground set X of points, e.g. a graph structure is built on the set of its vertices. Structures usually occur in a labeled or in an unlabeled form. In the labeled form the points of X carry numbers, letters or other kind of labels in order to distinguish the individual points of X. In many combinatorial applications, the labeling often does not really matter. So the corresponding unlabeled structure is defined to be the equivalence class of all labeled structures, arising from each other by relabeling. In order to formally introduce discrete structures let $X = \{1, \ldots, n\}$ be a finite set.

Species of structures: A *species of structures* is a rule $\mathcal{S}$ which

1. produces for each finite set X a finite set $\mathcal{S}[X]$,
2. produces for each bijection $\phi : X \to Y$ a function $\mathcal{S}[\phi] : \mathcal{S}[X] \to \mathcal{S}[Y]$.

The function $\mathcal{S}[\phi]$ should satisfy the following functorial properties:

1. for all bijections $\phi : X \to Y$, $\pi : Y \to Z$, we have $\mathcal{S}[\pi \circ \phi] = \mathcal{S}[\pi] \circ \mathcal{S}[\phi]$
2. for the identity map $1_X : X \to X$, we have $\mathcal{S}[1_X] = 1_{\mathcal{S}[X]}$.

The elements of $\mathcal{S}[X]$ are called *$\mathcal{S}$-structures* on X and the function $\mathcal{S}[\phi]$ describes the *transport* of $\mathcal{S}$-structures along ϕ. Note that $\mathcal{S}[\phi]$ is a bijection by definition of the functorial property of $\mathcal{S}$. Thus a species of structures is a mapping $\mathcal{S} : X \to \mathcal{S}[X]$ where $\mathcal{S}[X]$ is a finite set consisting of elements $\sigma \in S[X]$ that can be expressed in terms of the labels $i \in X$ only.

Example 1 *Let X be a finite set.*

1. *Species $\mathcal{P}[X]$ of all subsets of X: $\mathcal{P}[X] = \{Y \ : \ Y \subseteq X\}$*
2. *Species $\mathcal{P}^{[k]}[X]$ of all k-element subsets of X: $\mathcal{P}^{[k]}[X] = \{Y \subseteq X \ : \ |Y| = k\}$*
3. *Species $\mathcal{G}[X]$ of* simple graphs*: $\mathcal{G}[X] = \{(\sigma, X) \ : \ \sigma \subseteq \mathcal{P}^{[2]}[X]\}$*
4. *Species $\mathcal{P}er[X]$ of all permutations: $\mathcal{P}er[X] := \{\pi \mid \pi : X \to X, \textit{bijectively}\}$*

Induced substructures: Let $Y \subseteq X$ a subset of X and $\sigma \in \mathcal{S}[X]$ be a structure. By $\sigma|_Y$ we denote the *restriction* of σ to the ground set Y, if $\sigma|_Y$ exists and is a member of $\mathcal{S}[Y]$. In this case we call $\sigma|_Y$ an *induced substructure* of σ induced by the subset Y. A species $\mathcal{S}[X]$ is said to be *irreducible*, if there exists a subset $Y \subseteq X$ and a structure $\sigma \in \mathcal{S}[X]$ such that $\sigma|_Y \notin \mathcal{S}[Y]$. Otherwise, we call $\mathcal{S}[X]$ *reducible*. Similarly, the elements of an irreducible (reducible) species are called irreducible (reducible) structures.

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 107-112

Example 2 *Let $Y \subseteq X$.*

1. *$\mathcal{P}[X]$ is reducible. Any $\sigma \in \mathcal{P}[X]$ is a subset $\sigma \subseteq X$. For any subset $Y \subseteq X$ the restriction $\sigma|_Y = \sigma \cap Y$ is a subset of Y and therefore an element of $\mathcal{P}[Y]$.*
2. *$\mathcal{P}^{[k]}[X]$ is irreducible. The restriction of a k-element subset $\sigma \subseteq X$ to any subset Y with $|Y| < k$ is a subset with less than k elements.*
3. *$\mathcal{G}[X]$ is reducible. Let (σ, X) be a graph and $Y \subseteq X$. The restriction $\sigma|_Y = \sigma \cap \mathcal{P}^{[2]}[Y]$ yields the induced subgraph $(\sigma|_Y, Y) \in \mathcal{S}[Y]$.*
4. *$\mathcal{P}er[X]$ is irreducible. Let $i \in X \setminus Y$. Then the restriction of any permutation $\sigma \in \mathcal{P}er[X]$ mapping an element $j \in Y$ to i does not exist.*

Isomorphisms: Let X and Y be finite sets. An *isomorphism* ϕ from $\sigma \in \mathcal{S}[X]$ to $\tau \in \mathcal{S}[Y]$ is a bijection $\phi : X \to Y$, $i \mapsto i^\phi$ such that $\mathcal{S}[\phi](\sigma) = \tau$. In this case we call σ and τ *isomorphic* ($\sigma \cong \tau$). An automorphism of σ is an isomorphism from σ to σ. The set $\mathrm{Aut}(\sigma)$ of all automorphisms of σ forms a group called the *automorphism group* of σ.

Modular product: Let $\sigma \in \mathcal{S}[X]$ be a reducible structure and $\boldsymbol{i} = (i_1, i_2)$ and $\boldsymbol{j} = (j_1, j_2)$ elements of $\mathcal{P}^{[2]}[X]$. The modular product of $\boldsymbol{i} \diamond \boldsymbol{j}$ is defined to be

$$\boldsymbol{i} \diamond \boldsymbol{j} = \begin{cases} 1 & : \text{ if } \sigma_1 \cong \sigma_2 \\ -1 & : \text{ otherwise} \end{cases}$$

where $\sigma_1 = \sigma|_{\{i_1, j_1\}}$ and $\sigma_2 = \sigma|_{\{i_2, j_2\}}$ are isomorphic substructures of σ. We call $\boldsymbol{i}$ and $\boldsymbol{j}$ *compatible*, if $\boldsymbol{i} \diamond \boldsymbol{j} = 1$ and *incompatible* otherwise.

The species graph: Let $(\sigma, X) \in \mathcal{G}[X]$ be a graph structure. The elements of X and σ are called *vertices* and *edges*, respectively. A graph $(\tau, Y) \in \mathcal{G}[Y]$ is an *induced substructure* of $(\sigma, X) \in \mathcal{G}[X]$, if $Y \subseteq X$ and $\tau = \sigma \cap \mathcal{P}^{[2]}[Y]$. A *clique* C of a graph (σ, X) is a subset $C \subseteq X$, such that $\mathcal{P}^{[2]}[C] \subseteq \sigma$. A *maximum clique* is a clique with maximum number of vertices. A *maximal clique* is a clique which is not contained in any larger clique. By $\mathcal{G}_w[X]$ we denote the species of *weighted graphs*. A weighted graph (σ, X) is a graph where each edge of σ is weighted by a real valued scalar. In the following we simply write σ instead of (σ, X), if the ground set X is known.

3 An Algebraic Associative Memory

The algebraic associative memory stores the automorphism group of a given labeled reducible structure σ. Since the pattern to be stored are unknown in general, the associative memory stores the automorphisms of σ implicitly by memorizing σ itself. After learning, the memory provides an unlabeled presentation of the labeled structure σ, containing the automorphism group $\mathrm{Aut}(\sigma)$. The primary function of an associative memory is to retrieve the memorized patterns (automorphisms) in response to a corrupted input pattern.

For the following mathematical analysis let $|X| = n$ and $N = n^2$. The patterns are binary relations $\varrho \subseteq X^2$ encoded as N-dimensional binary vectors $\boldsymbol{x}(\varrho) \in \{0, 1\}^N$. The components $x_{\boldsymbol{i}}(\varrho)$ of $\boldsymbol{x}(\varrho)$ are indexed by $\boldsymbol{i} = (i_1, i_2) \in X^2$,

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 107-112

where $x_{\boldsymbol{i}} = 1$ if and only if $\boldsymbol{i} = (i_1, i_2) \in \varrho$. We say a binary relation ϱ maps i to j, if $(i,j) \in \varrho$. For example an automorphism $\pi \in \mathrm{Aut}(\sigma)$ is a binary relation encoded as a vector $\boldsymbol{x}(\pi)$ with components

$$x_{\boldsymbol{i}}(\pi) = \begin{cases} 1 & : \text{ if } \boldsymbol{i} = (i, i^{\pi}) \\ 0 & : \text{ otherwise.} \end{cases} \tag{1}$$

Storing Phase. The essence of an associative memory is to map the patterns onto the stable equilibrium points of the underlying dynamic system. We apply this principle to store the unknown automorphism group of a structure σ by storing σ itself.

The topology of an associative memory is best described in terms of the theory of species. The *association structure* A_σ of a structure $\sigma \in \mathcal{S}[X]$ is a complete weighted graph structure on $\mathcal{P}^{[2]}[X]$ with weights $w_{\boldsymbol{ij}}$ between vertices $\boldsymbol{i}$ and $\boldsymbol{j}$. To store the automorphisms of σ, the weights are adjusted according to a similar form of the generalized Hebb rule for the one pattern problem

$$w_{\boldsymbol{ij}} = \eta_{\boldsymbol{ij}} \cdot (\boldsymbol{i} \diamond \boldsymbol{j}) \tag{2}$$

where $\eta_{\boldsymbol{ij}} > 0$ depends on the type of connection. If $\boldsymbol{i}$ and $\boldsymbol{j}$ are compatible (incompatible), then their connection type is *excitatory* (*inhibitory*) with synaptic weight $w_{\boldsymbol{ij}} = \eta_{\boldsymbol{ij}} = w_E > 0$ ($w_{\boldsymbol{ij}} = -\eta_{\boldsymbol{ij}} = -w_I < 0$). After learning, an unlabeled presentation of σ is stored in the associative memory. In the next part we justify our approach and show that the memory indeed stores the automorphism group $\mathrm{Aut}(\sigma)$ by applying learning rule (2).

Retrieval phase. During retrieval the goal of the associative memory is to recall the *closest* automorphism $\pi \in \mathrm{Aut}(\sigma)$ encoded as $\boldsymbol{x}(\pi)$ given a corrupted input pattern $\boldsymbol{x}(\varrho)$, which represents a binary relation $\varrho \subseteq X^2$. Typical application scenarios would be to test whether a given permutation of X is an automorphism of σ, or trying to extend partial permutations defined on a subset $Y \subseteq X$ to an automorphism. More generally, besides automorphisms, an input pattern can represent ambiguous, deformed, distorted, or incomplete version of automorphisms or mixtures of those versions. Ambiguities are binary relations $\varrho \in X^2$, which map at least one element $i \in X$ to distinct elements of X. Deformations are relations ϱ, which correspond to non-injective mappings on X. An incompletion is a synonym for a partial permutation.

Given an input pattern, we map the problem of retrieving the closest automorphism of a structure to the problem of finding a maximum clique in the *positive association structure* A_σ^+ of σ. The structure A_σ^+ is a simple graph on $\mathcal{P}^{[2]}[X]$ with $\{\boldsymbol{i}, \boldsymbol{j}\} \in A_\sigma^+$ if and only if $\boldsymbol{i} \diamond \boldsymbol{j} = 1$. Hebb's rule establishes a basis for an an associative memory such that its attractors corresponding to the global minima of the underlying energy function are in 1-1 correspondence to the automorphism of σ. Theorem 1 shows that the maximum cliques of the positive associative structure A_σ^+ are in 1-1 correspondence to the automorphisms of σ. Examples of dynamical rules such that the maximum cliques indeed correspond to the global minima of the underlying energy function

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 107-112

are shown in [3], [4]. This maps the retrieval of the fundamental memories to the problem of finding a maximum clique in A_σ^+.

Theorem 1 *Let A_σ be the associative memory of a reducible structure $\sigma \in \mathcal{S}[X]$. Then there exists a bijection*

$$\chi : Aut(\sigma) \to \mathcal{MC}(A_\sigma^+)$$

from the automorphism group $Aut(\sigma)$ to the set of all maximum cliques in A_σ^+.

Proof: Let $\pi \in \mathrm{Aut}(\sigma)$. Define $\chi(\pi) = C_\pi = \{(i, i^\pi) \ : \ i \in X\}$. First we show that C_π is a clique of A_σ^+ with n vertices. Let $\boldsymbol{i}, \boldsymbol{j} \in C_\pi$ with $\boldsymbol{i} = (i_1, i_2)$ and $\boldsymbol{j} = (j_1, j_2)$. By definition of χ, we have $i_1^\pi = i_2$ and $j_1^\pi = j_2$. Then the induced substructures $\sigma|_{\{i_1,j_1\}}$ and $\sigma|_{\{i_2,j_2\}}$ are isomorphic. By definition of the modular product we get $\boldsymbol{i} \diamond \boldsymbol{j} = 1$. This yields $\{\boldsymbol{i}, \boldsymbol{j}\} \in A_\sigma^+$. Thus C_π is a clique. In addition, C_π consists of n vertices, since π is bijective and by definition of χ.

Next we show that C_π is a maximum clique. Assume that there exists a clique C with $|C| > n$. Any vertex $\boldsymbol{i} \in C$ is of the form $\boldsymbol{i} = (i_1, i_2)$ with $i_1, i_2 \in X$. Since $|X| = n$ and $|C| \geq n + 1$, there exist at least two vertices $\boldsymbol{i}, \boldsymbol{j} \in C$ with $i_1 = j_1$ and $i_2 \neq j_2$. Then the induced substructures $\sigma|_{\{i_1,j_1\}} = \sigma|_{\{i_1\}}$ and $\sigma|_{\{i_2,j_2\}}$ are not isomorphic. Hence, $\boldsymbol{i} \diamond \boldsymbol{j} = -1$ and $\{\boldsymbol{i}, \boldsymbol{j}\} \notin A_\sigma^+$. This contradicts our assumption that C is a clique.

So far we have shown that χ maps automorphisms to maximum cliques. Furthermore by definition χ is well defined. Thus it is left to show that χ is bijective. First we prove that χ is an injection. Let $\pi \neq \phi$ distinct automorphisms of σ. Then there exists an element $i \in X$ with $i^\pi \neq i^\phi$. Thus $(i, i^\pi) \in C_\pi \setminus C_\phi$. Hence, χ is injective. To show that χ is surjective let C be a maximum clique. Then the modular product of $\boldsymbol{i}$ and $\boldsymbol{j}$ is 1 for any pair of vertices $\boldsymbol{i}, \boldsymbol{j} \in C$. *Lifting back* the definition of the modular product yields an automorphism induced by C. This shows that χ is surjective. □

Let A_σ be the associative memory of σ and $\boldsymbol{x}$ be a corrupted input pattern. To retrieve the closest automorphism from the memory we map that problem to the problem of finding a maximum clique given that $\boldsymbol{x}$ is imposed as initial activation onto the associative memory. Theorem 1 justifies this approach. A standard procedure to solve the maximum clique problem by means of minimizing the energy function of the associative memory according to the Hopfield & Tank approach [2] proceeds as follows: After the initial activation is imposed on the network, the memory evolves in accordance to a dynamical rule as given in [3] or [4]

$$x_{\boldsymbol{i}}(t+1) = x_{\boldsymbol{i}}(t) + \ w_E \cdot \sum_{\boldsymbol{j} \in N(\boldsymbol{i})} o_{\boldsymbol{j}}(t) - \ w_I \cdot \sum_{\boldsymbol{j} \notin N(\boldsymbol{i})} o_{\boldsymbol{j}}(t) \tag{3}$$

where $x_{\boldsymbol{i}}(t)$ denotes the activity of unit $\boldsymbol{i} \in \mathcal{P}^{[2]}[X]$. $N(\boldsymbol{i})$ is the set of all vertices $\boldsymbol{j}$ adjacent to vertex $\boldsymbol{i}$. The output function $o_{\boldsymbol{i}}(t)$ of unit $\boldsymbol{i}$ is a non-decreasing function applied on its activation $x_i(t)$. During evolution of the memory any unit is excited by all active units with which it can form a clique and inhibits all other units. After convergence the stable state corresponds to a maximal clique of X. The size of a maximal clique can be read out by counting

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 107-112

the units with output $o_i(t) = 1$.

Note, that the dynamical rule (3) as stated in different versions in [4], [3] ensure convergence of the system to a stable state corresponding to a maximal clique where the maximum cliques correspond to the global and the maximal cliques to the local minima of the underlying energy function. Thus the output of the associative memory corresponds either to the automorphism closest to the input pattern if the retrieved clique C is a maximum clique or to a spurious state if C is a maximal but not a maximum clique.

The spurious states give us some additional information about the memorized structure. The maximal cliques of A_σ^+ are in 1-1 correspondence to the maximal isomorphisms between induced substructures of σ. An isomorphism ϕ between induced substructures is induced by a partial permutation on the set X. We call the isomorphism ϕ maximal, if the associated partial permutation can not be extended to an automorphism of σ. This statement is summarized in Theorem 2. The proof of Theorem 2 is similar to the proof of Theorem 1.

Theorem 2 *Let A_σ be the associative memory of a reducible structure $\sigma \in \mathcal{S}[X]$. Then the maximal cliques of A_σ^+ are in 1-1 correspondence to the maximal isomorphisms between induced substructures of σ.*

4 Conclusion

We presented an associative memory model for storing the automorphism group of any discrete reducible structure. The proposed model carries an algebraic structure and thus links the neurodynamic energy minimization process to the theory of permutation groups. Retrieval of *relevant* memorized patterns combined with basic results from group theory may yield an effective way to compute the automorphism partition or the automorphism group.

References

[1] F. Bergeron, G. Labelle, and P. Leroux. *Combinatorial species and tree-like structures.* Encyclopedia of Mathematics and its Applications, 67. Cambridge University Press, 1998.

[2] J.J. Hopfield and D.W. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52:141–152, 1985.

[3] A. Jagota. Approximating maximum clique with a Hopfield network. *IEEE Trans. Neural Networks*, 6:724–735, 1995.

[4] B.J. Jain and F. Wysotzki. Fast winner-takes-all networks for the maximum clique problem. In *25th German Conference on Artificial Intelligence*, pages 163–173. Springer, 2002.

[5] M. Klin, Ch. Rücker, G. Rücker, and G. Tinhofer. Algebraic Combinatorics in Mathematical Chemistry. Methods and Algorithms. I. Permutation Groups and Coherent (Cellular) Algebras. *Communications in mathematical and in computer chemistry*, pages 1–138, 1999.