Compact Crossbars of Multi-Purpose Binders for Neuro-Symbolic Computation

Work in Progress Report

Gadi Pinkas
Center for Academic Studies, Israel
gadip@mla.ac.il

Priscila M. V. Lima
Universidade Federal Rural do Rio de Janeiro
Seropédica, Brazil
priscilamvl@ufrrj.br

Shimon Cohen
Center for Academic Studies, Israel
shamon51@gmail.com

Abstract

We present a compact – yet expressive – Multi-purpose, distributed binding mechanism, which is useful for encoding complex symbolic knowledge and computation, using Artificial Neural Networks (ANNs) or using Satisfiability (SAT) solvers. The technique is demonstrated by encoding unrestricted First Order Logic (FOL) unification problems as Weighted Max SAT problems and then translating the later into ANNs (or learning them). It is capable of capturing the full expressive power of FOL, and of economically encoding a large Knowledge Base either as long term synapses or as clamped units in Working Memory. Given a goal, the mechanism is capable of retrieving from the synaptic knowledge just what is needed, while creating novel, compound structures in the Working Memory. Two levels of size reduction are shown. First, we build a Working Memory, using a pool of multi-purpose binders, based on the assumption that the number of bindings that are actually needed is far less than the number of all theoretically possible bindings. The second level of compactness is due to the fact that, in many symbolic representations, when two objects are bound, there is a many-to-one relationship between them. This happens because, frequently, either only one value is pointed by variable or only one variable point to a value. A crossbar binding network of $n \times k$ units with such restriction, can be transformed into an equivalent neural structure of size $O(n \log(k))$. We show that, for performing unrestricted FOL unifications, the Working Memory created is only log dependent on the KB size; i.e., $O(n \log(k))$. The variable binding technique described is inherently fault tolerant as there are no fatal failures, when some random neurons become faulty and the ability to cope with complex structures decays gracefully. Processing is distributed and there is no need for a central control even to allocate binders. The mechanism is general, and can further be used for other applications, such as language processing, FOL inference and planning.

1 Introduction

1.1 The Binding Problem

Human cognition is capable of producing combinatorial structures. The general binding problem concerns how items that are encoded in distinct circuits of a massively parallel computing device (such as the brain or ANN) can be combined in complex ways for perception, reasoning or for action [Feldman 2010]. Consider for example, a planning problem, where the task is to pick up an object and move it from its current position to another place. In order to meet a goal, a "brain"-like device, must be able to represent the object, its properties, its position and the ways to manipulate it, in such a way that the goal is achieved. The object and its properties must be bound together, and this rather complex structure should also be used in conjunction with other entities and rules, such as the action consequences (e.g., moving X from Y to Z clears position Y while occupying position Z). In another example, consider the sentence: "Sally ate": In language processing, the verb "EAT" is a predicate with at least two roles - EAT("Sally",X). The noun "Sally" should be bound to the first role, while an existentially quantified variable (representing "something") should be bound to the second role. Once we get the information that "Sally ate salad", and knowing the rule: EAT(Y,X)⇒ DIGESTED(X) we should reason that "the salad is digested". In order to do that, we must bind the variable X to the noun "salad", while X must be bounded to both EAT(,X) and DIGESTED(X).

1.2 Connectionism and Variable Binding

During the years, connectionist systems have been criticized for "Propositional Fixation" [McCarthy 1988]. In [Fodor, Phylyshyn 1988] connectionism was criticized for lacking abilities to construct combinatorial representations and for performing processes that are sensitive to complex structure. Exactly how compositionality can occur is a fundamental question in cognitive science and the binding aspect of it has been identified as a key to any neural theory of language [Jackendoff 2002]. Several attempts have been made to approach the variable binding problem in a connectionist

framework [Shastri, Ajjanagadde 1993], [Browne, Sun 2000], [Zimmer et al. 2006], [Van der Velde, Kamps, Kamps 2006], [Barret et al. 2008], [Velik 2010]; yet, virtually all these suggestions, have limitations, related to either limited expressiveness, size and memory requirements, central control demands, lossy information, etc.

For example, compositionality can be provided using Hollographic Reduced Represenations [Plate 1995]; however, the convolution operation used, is lossy and errors are introduced as structures become more complex or as more operations are done. The BlackBoard Architecture [Van der Velde, Kamps, Kamps 2006] can form complex structures but does not manipulate those structures to perform cognition. Shastri's temporal binding has only limited FOL expressiveness and no mechanism for allocating temporal binders. Finally, all the above systems need neurons in numbers that is at best linear in the KB; while some use much more neurons than that.[1] For FOL compositionality in ANNs see [Ballard 1986], [Pinkas 1992], [Shastri 1999], [Lima 2000], [Garcez, Lamb 2006]. For partial-FOL encodings in Satisfiability, see [Domingos 2008], [Clark et al. 2001].

The ability to represent combinatorial structures and reasoning with them, still presents challenges to theories of neurocognition [Marcus 2001], while the variable binding problem is fundamental to such ability [Feldman 2010].

1.3 Unification

In conventional computing, unification is a key operation for realizing inference, reasoning, planning and language processing. It is the main vehicle for conventional symbolic systems to match rules with facts, or rules with other rules. In unification, two or more distinct hierarchical entities (terms) are merged, to produce a single, unified, tree-like structure. This unified structure adheres to the constraints of both the original entities. Formally, unification is an operation which produces from two or more logic terms, a set of substitutions, which either identifies the terms or makes the terms equal modulo some equational theory. For connectionist approaches to unification see [Hölldobler 1990], [Weber 1992], [Komendantskaya 2010].

For easiness of reading, we have chosen to demonstrate our compact variable binding mechanism on the more fundamental unification function, rather than on full FOL inference.

1.4 Artificial Neural Networks and SAT

ANNs may be seen as constraint satisfaction networks, where neuron-units stand for Boolean variables, and where the synapse weights represent constraints imposed on the variables. Any ANN may be seen as such a constraint network; yet, for ANNs with symmetric weights (e.g. Hopfield, Boltzmann Machines, MFT) a simple conversion has been shown for translating any Weighted MAX SAT problem into symmetric ANN and vice-versa [Pinkas, 1991]. Any such SAT problem could be compiled into an ANN, which performs stochastic gradient descent on an energy function that basically counts the number of unsatisfied logical constraints. The size of the generated network is linear in the size of the original formula, though additional hidden units may be required. In addition to compilation, the logical constraints of a network could be PAC learnt using Hebbian-like rule [Pinkas 1995], thus, for small-size constraints, a network can efficiently learn its weights and structure from a training set that is composed of the satisfying models. The performance efficiency of this neural mechanism can be attributed to the similarities of symmetric ANNs to stochastic local search algorithms, such as WALKSAT [Kautz et al 2004]. Due to the tight relationship between ANNs and Weighted Max SAT, our methodology is to specify an ANN designed for certain symbolic computation (e.g. unification), using a set of Boolean variables (the visible units) and a set of constraints; i.e., Boolean formulae designed for restricting the values of the visible units. The constraints specified are used to force the visible units to converge to a valid solution that satisfies as many (weighted) formulae as possible. We have written a compiler that translates such specifications into either weighted CNF (for Weighted Max SAT Solvers) or for ANN with symmetric weights.

We believe that our fault tolerant mechanism and methods for dynamically forming recursive structures will scale and be useful for both the engineering of massively parallel devices, and for modeling of high-level cognitive processes.

2 Improving CrossBar Binding

The simplest, most naïve binding techniques is CrossBar binding. The term was mentioned in [Barrett et al. 2008], yet it was intuitively used by many connectionist systems in the past [Ballard 1986], [Anandan 1989] and in many SAT reductions; e.g., [Kautz, at el 2006]. Formally, we define crossbar binding as a Boolean matrix representation of a relation between 2 sets of items, using Characteristic Matrix of the relation; i.e., if A contains m objects and B contains n objects, then the characteristic matrix R has m lines and n columns, containing $m \times n$ Boolean variables (neurons). We say that *item i is bound to item j iff R(i,j)=1*. In this naïve, binding mechanism, a neuron should be allocated for each possible binding, and all theoretic combinations of two items must be pre-enumerated as rows and columns of the matrix. A crossbar matrix, that needs to represent a complex tree or a graph, must bind together not just simple constituents, but all the compounded entities representing partial trees (or sub-graphs). It is possible to represent a FOL KB this way at the cost of using an enormous number of neurons, and with an extremely localist approach. Even more frustrating is the fact that this technique will not be suitable for dynamically creating novel, nested structures upon demand. The number of theoretic bindings, for all possible tree

[1] The BlackBoard architecture uses billions of neurons to represent thousands of atomic concepts; HRR Production systems [Stewart, Elliasmith 2008] needs about one million neurons.

structures, grows exponentially with the number of constituent items and must be computed in advance.
We improve this simplistic binding mechanism in several steps:

2.1 Using Binders as "pointers" to form Graphs

First, we introduce[2] a special kind of entities called General Purpose Binders (GPBs). GPBs are similar to pointers except for the fact that a single GPB can point to several objects, as the crossbar paradigm permits implementation of arbitrary relations between binders and objects. In the special case where binders point to binders, arbitrary directed graphs can be built. In this scenario, we can interpret each GPB as a node in the graph, and the crossbar, as specifying the arcs of the graph (adjacency matrix). In such graph interpretation, each node may be labeled using a labeling crossbar, that ties together binders, with symbols such as, predicates, functions or constants in FOL. Arcs can also be labeled, as the binder-to-binder crossbar, may have a third dimension which relates one or more labels to each arc. This enables the formation of arbitrary complex graph structures, that can be used to represent language constituents and in particular, FOL terms, predicates, literals and clauses. Unlike in the naïve crossbar approach, unrestricted graphs can be built directly out of simple constituents, with GPB as the mechanism for gluing them together.

Because the binders are general-purpose entities, we can construct a working memory out of a pool of such binders. As long as GPBs remain unallocated, they can be used for dynamic creation of novel, goal oriented structures. To do so, the "right" constraints should be embedded in the synapses, forcing binders first to be allocated and then to assume a desired structure for solving the goal. These constraints, stored at the synaptic weights, are the driving force that causes the visible units to converge to the needed graph-like structures.

Using this technique, we show that arbitrary KB of size k, can be encoded in a working Memory (WM) with O(k) binders and with a total size of $O(k^2)$. Unfortunately, when the KB tends to grow, the WM and the set of constraints may become too large for the mechanism to be used in real applications. [3]

2.2 Using a pool of binders "As Needed"

Luckily, we can reduce that size requirement, drastically, as we can assume that, at a certain time, only few binders are actually needed for the processing of a given goal. This is supported by cognitive studies [Cowan 1981] and constitutes a common assumption of several connectionist systems [Shastri, Ajjanagadde 1993], [Barrett et al 2008]. We therefore can design a Working Memory of neural units, which uses only a pool of General Purpose Binders, labeled and nested within each other; i.e., a small set of binders, for representing only those graphs that are actually needed for computing the goal. It turns out that this approach is consistent with cognitive theories, where a large KB is stored in synapses (long term memory); and a smaller size working memory is used for retrieving only few KB items at a time. Only those items that are necessary to the process[4] get to be retrieved from the synaptic KB. For example, if our purpose is to find a plan for a goal, expressed in FOL, we need to design the WM with enough binders to represent a valid plan. We retrieve the facts and rules of the world from that KB only if they are required by the plan we desire to make.

To implement a pool of binders for FOL unification, the WM should contain three crossbar matrices: One for labeling nodes by symbols (predicates, functions, constants). The second is for nesting of the nodes in Graphs and labeling the arcs according to slots of the predicates and functions. The third crossbar is for retrieving items from the long term memory where the KB is stored (e.g., terms, literals or clauses). This third matrix ties a binder to a KB item and triggers the constraints of that item to be activated so that the binder node is forced to assume the structure of the KB item retrieved. The mechanism starts working as goal activated constraints cause some binders to be tied to KB items and activate some KB constraints. Those constraints, in turn, activate other constraints, till the WM converges to a valid solution. When we implement unification problems, the size of the WM is O(n ×k) where n is the maximal number of nodes in a solution; k is the size of the KB and n<<k. This constitutes a drastic improvement, as the WM size is linear in the size of the KB, instead of being quadratic.[5] Actually, we can do even better:

2.3 Crossbars with n*log(k) size complexity

In the next size improvement, we further reduce the size of many crossbar matrices from O(n*k) to O(n* log(k)). Thus, in our unification example, a WM of O(n* log (k)) is created, where n is the maximal size of a unification tree and k is the size of the KB. This means that the WM size is only log dependent[6] on the KB size; rather than linearly as in previous section.

The key to this log-reduction, is the fact that frequently, binding relationships have *many-to-one* or one-to-many restrictions. For example, the crossbar matrix for node labeling, allows for a binder to point only to a single symbol (whereas many binders could point to the same symbol). This *many-to-one* relationship causes the rows of the crossbar labeling matrix to be *Winner-Takes-All* (WTA) arrays, where only one neural unit (if any) may fire. Normally, we need mutual exclusion constraints to force the rows of the matrix to be either all-zeros or have a single variable set to one. In such a scenario, however, we can replace each WTA

[2] The method was suggested in [Pinkas 1992] and used in [Lima 2000], [Lima 2007] for clamping a KB in Working Memory.

[3] $O(k^3)$ constraints are needed for unification in this paradigm.

[4] When an item is already in WM, no retrieving is needed.

[5] The number of *constraints* needed for unification is $O(n^2k)$; linear in the KB size, when n<<k.

[6] Even, if occurs check is used, the WM size is still linear in the KB size when n<<k.

line (with k-variables), with a much smaller size line of only O(log(k)) variables. Each such line of log(k) variables (neurons), represents an index (or a signature) to the target label. Therefore, if a binder may point to just a single object (out of k possible objects), we may use only log(k) bit signatures. Fig 1 illustrates, how one binder with WTA line that points to object 6 (out of 15 objects) is reduced to only 4 bits LOG WTA array, representing the signature of that item. This signature, once it emerges in a binder's row, activates a set of constraints associated with the bounded object. These constraints force the binder to get the retrieved item's structure and may cause a chain reaction of more constraints, retrieving more KB items and so forth.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

1	2	3	4
0	1	1	0

Figure 1. On top: A Standard WTA pointing to the 6th object; Bellow: a LOG WTA array with a binary value of 6 representing the 6th object's signature '0110'.

It should be noted that, once a LOG WTA encoding is used instead of the standard WTA, the constraints imposed on WM might need to be adjusted. [7]

3 Fault Tolerance

The variable binding mechanism suggested and its application to unification are inherently fault tolerant, if each variable is allocated a processing unit (a neuron). Small random damage to the neurons does not radically affect the unification process (if at all). For example, if a single neuron related to a binder, in one of the crossbar matrices, becomes faulty and stops firing, then the binder cannot point to a certain symbol; however, other binders from the pool can be used for pointing to that symbol if such is needed. In the meantime, this "faulty" binder may still be used, as it can be allocated to point to other symbols. Even if the faulty neuron starts firing constantly, it may still participate in the process if the symbol that is pointed by that "faulty" binder happens to be needed. The binder will simply not be used, if that symbol is irrelevant to the goal. If the damage to the WM neurons is more widespread, so that a binder cannot take part in the process, then this binder will not be allocated, and therefore will not be used in the graph construction. This may shorten the number of available GPB nodes in the largest graph but will not destroy the ability of the WTA to unify less complex terms (shallower trees).[8]

4 Conclusions

We have shown a general purpose binding mechanism that uses a pool of general purpose binders, and allocates them to KB items, only when they are necessary for achieving the goal. A large KB may be stored in long term connections rather than in the Working Memory. KB constraints are activated only upon need, and only if they are supportive for achieving the goal. We then showed that further log reduction is possible if the binding represents a many-to-one relationship. The size of a crossbar matrix is then reduced from O(n*k) to O(n*log(k)) and the number of constraints is also reduced.[9] We demonstrated the use of the suggested binding technique in ANN that performs FOL unification with size[10] that is O(n×log(k)). The mechanism is distributed since there is no central control and even binder allocation is done in a totally distributed way. It is also inherently robust, as no fatal failures occur when neurons "die". We have performed initial experiments with the GPB pool mechanism (without the LOG WTA reduction), these experiments indicate the feasibility of the approach on rather complex unification tasks including multi-instance parallel-unification and recursive occurs checking. LOG WTA and fault tolerance experiments are the subject of ongoing work. The mechanism described is general and can further be used for other applications such as: language processing, FOL inference and planning. We are working on extending the techniques, for full FOL inference and conjecture that these techniques will also improve other SAT encodings that use crossbar-like bindings, e.g. as in [Kautz, et al 2006].

References

[Ballard 1986] D. H. Ballard. Parallel logical inference and energy minimization. In Proceedings of the AAAI National Conference on Artificial Intelligence, pages 203–208, 1986.

[Barrett et al 2008] Barrett L, Feldman JA, Mac Dermed L. A (somewhat) new solution to the binding problem. Neural Computation, 20: 2361-237, 2008.

[Brown, Sun 2000] A. Browne and R. Sun. Connectionist variable binding. Springer, Heidelberg, 2000.

[Clarke et.al 2001] E. Clark, A. Bier, R. Raimi, Y. Zhu, Bounded Model Checking Using Satisfiability Solving, in Formal Methods in System Design archive, Volume 19 Issue 1, July 2001.

[7] E.g., mutual exclusion constraints - for enforcing WTA are eliminated. Long OR constraints of O(k) size, become only of log(k) length.

[8] This property may help in supporting neuro-linguistic theories that relate certain symptoms of aphasia, with loosing abilities for processing deep structures; e.g. deep syntactic tree pruning in Agrammatism and Broca's Aphasia [Friedman 2002].

[9] Note, that some constraints may grow in size complexity; while others may shrink.

[10] $O(n^2 + \log(k))$ with occurs check.

[Cowan 2001] N Cowan, The magical number of 4 in short term memory: A reconsideration of mental storage capacity, Behavioral and Brain Sciences, 1(24), 2001.

[Domingos 2008] P. Domingos, Markov logic: a unifying language for knowledge and information management. CIKM 2008:519.

[Feldman 2010] J. Feldman, The Binding Problem(s), http://www.computational-logic.org/content/events/iccl-ss-2010/slides/feldman/papers/Binding8.pdf, accessed on May 2010.

[Fodor, Phylyshyn 1988] J. A. Fodor and Z. W. Phylyshyn, Connectionism and cognitive architecture: A critical analysis. In Pinker and Mehler (eds): Connectionism and Symbols, 3-71, MIT Press, 1988.

[Friedman 2002] Friedmann, N.Syntactic tree pruning and question production in agrammatism. Brain and Language, 83, 117-120

[Garcez, Lamb, 2006] A. S. d'Avila Garcez and L. C. Lamb. A Connectionist Computational Model for Epistemic and Temporal Reasoning. Neural Computation 18(7):1711-1738, MIT Press, July 2006.

[[Garcez, Lamb 2006] A. S. d'Avila Garcez and L. C. Lamb. A Connectionist Computational Model for Epistemic and Temporal Reasoning. Neural Computation 18(7):1711-1738, MIT Press, July 2006.

[Hölldobler 1990] S. Hölldobler, A Connectionist Unification Algorithm. International Computer Science Institute, Berkeley, TR-90-012: 1990.

[Jackendoff 2002] Jackendoff, Ray (2002). Foundations of Language: Brain, Meaning, Grammar, Evolution. Oxford: Oxford University Press. pp. 477.

[Kautz et al 2004] Henry Kautz, Bart Selman, & David McAllester, Walksat in the 2004 SAT Competition, International Conference on Theory and Applications of Satisfiability Testing, Vancouver, 2004

[Kautz at el 2006] Henry Kautz, Bart Selman, and Joerg Hoffmann. SatPlan: Planning as Satisfiability, Abstracts of the 5th International Planning Competition, 2006

[Komendantskaya 2010] E. Komendantskaya, Unification neural networks: unification by error-correction learning, Logic Jnl IGPL, 2010

[Lima, 2000] P. M. V. Lima: Resolution-Based Inference on Artificial Neural Networks. Ph.D. Thesis, Department of Computing. Imperial College London, UK (2000).

[Lima et al 2007] P. M. V. Lima, M. Mariela, M. Morveli-Espinoza and F. M. G. França, Logic as Energy: A SAT-Based Approach, Advances in Brain, Vision, and Artificial Intelligence Lecture Notes in Computer Science, 2007.

[Marcus 2001] G. F. Marcus, The algebraic mind. Cambridge, MA: MIT Press, 2001.

[McCarthy 1988] J. McCarthy, Epistemological challenges for connectionism. Behavioral and Brain Sciences, 11:44, 1988.

[Pinkas 1991] G. Pinkas, "Symmetric neural networks and logic satisfiability," Neural Computation 3, no. 2, pp.282-291, 1991.

[Pinkas 1992] G. Pinkas, "Constructing syntactic proofs in symmetric networks" in Advances in Neural Information Processing Systems – 4 (NIPS-91), pp.217-224, 1992.

[Pinkas 1995] G. Pinkas, "Reasoning, non-monotonicity and learning in connectionist networks that capture propositional knowledge", Artificial Intelligence Journal 77, (AIJ) pp. 203-247, 1995.

[Plate 1995] T. Plate Holographic Reduced Representations, IEEE Trans. On Neural Networks 6(3), 623-641

[Shastri, Ajjanagadde 1993] L. Shastri L. and V. Ajjanagadde, From associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. Behavioral and Brain Sciences 1993;16(3):417-494.

[Stewart, Eliasmith 2008] T.C. Stewart, C. Eliasmith, Building Production Systems with Realistic Spiking Neurons", Cognitive Science Conference. Washington, DC. August, 2008

[Velik 2010] R. Velik, From single neuron-firing to consciousness- towards the true solution of the binding problem, Neuroscience Behavioral Rev., 2010, 34(7): pp. 993-1001.

[Weber 1992] V Weber, Connectionist Unification with a distributed Representation, In IJCNN-92

[Zimmer et. Al. 2006], HD Zimmer, A. Mecklinger, U. Lindenberger (eds), Handbook of binding and memory - Perspectives from cognitive neuroscience. Oxford University Press 2006.