Generalized Dynamic Concept Model as a Route to Construct Adaptive Autonomous Agents

Zsolt Kalmár[†‡] kalmar@obelix.iki.kfki.hu

Csaba Szepesvári[†,⋆] szepes@math.u-szeged.hu

András Lőrincz[†⋆] lorincz@obelix.iki.kfki.hu

[†]Department of Photophysics, Institute of Isotopes,
The Hungarian Academy of Sciences.
Budapest, P.O. Box 77, Hungary 1525

[‡]Department of Computer Science,
József Attila University of Szeged
Aradi vrt. tere 1. Szeged, Hungary 6720

[⋆]Bolyai Institute of Mathematics,
József Attila University of Szeged
Aradi vrt. tere 1. Szeged, Hungary 6720

Abstract

A model of adaptive autonomous agents, that (i) builds internal representation of events and event relations, (ii) utilizes activation spreading for building dynamic concepts and (iii) makes use of the winner-take-all paradigm to come to a decision is extended by introducing generalization into the model. The generalization reduces

[⋆]To whom correspondance should be addressed.

memory requirements and improves performance in unseen scenes as it is indicated by computer simulations.

1 INTRODUCTION

The principle goal of an autonomous agent is survival. Experiencing failure in satisfying this goal is, however, late for the system. The route to this failure should be blocked by a set of supplementary goals. These goals can (a) either initiate reflexes or (b) serve as the base of collection of experiences. The reflex system may be considered as a set of IF-THEN rules: having a given combination of inputs being above a given decision surface a trial is made to accomplish an action. The supplementary goal system and the experience collecting subsystem, on the other hand, may be put in the form of connected patterns, i.e., in the form of artificial neural networks. This point is important and may be justified as follows.

The agent's information about the world is restricted. In general, no state as detected by the agent, and no action executed by the agent can be repeated in a perfect fashion. Small details, detected or undetected will be different in each case. We shall refer to this point as the first reading of the Heraclit principle, or principle H_1. Moreover, in many cases, the differing details are left undetected since the sensory system provides restricted information about the world and thus the state detected by the agent and the action executed in that state lead to different results. Principle H_1 thus means that it is not possible to collect rules as a first step.

There is another reading of the Heraclit principle, however, that we shall call principle H_2. The original saying of Heraclit claims that "One cannot step into the same river twice". We can rephrase it for our purposes in the following way: "Although we know that it is different, although we know that the consequences might be different, nevertheless in most cases they are not, it is still possible to talk about the river and the action step." That is principle H_2 would say the following: "It is possible to capture essential long lasting concepts from the sensory information." Principle H_1 and H_2 will be called as principle H.

The route to solve the dilemma of H is first to collect experiences and collect chains of experiences and form rules at a later stage. The nature of the detection system and the world is, however, such that chains of experi-

ences form a network as it follows from principle H_1. That is, we need both systems: the collection of IF-THEN rules, i.e. a tree-like system with sparse connectivity and the collection of experiences, i.e. the network of events.

Our algorithm aims to connect rule-based systems and artificial neural networks for decision making under the following building principles: (i) The system is goal oriented. (ii) The system collects experiences and chains of experiences. (iii) The system intends to grind the experiences into the form of rules, extends the goal system and creates subgoals. (iv) The system should be memory saving. (v) The system should be of low time complexity.

The present work is an extension of the Dynamic Concept Model (DCM) [12, 5]. DCM constructs an internal representation of events and a network of events. Activation spreading (AS) creates dynamic concepts and decision making is accomplished via winner-take-all principles. As a second step generalization was introduced into the model. The present work provides new results on the autonomous agent with generalization, such as improvement of performance in unseen worlds.

2 ARCHITECTURE AND FUNCTION

DCM has the following subsystems: (i) the dimension reducing system, (ii) a prewired goal system, (iii) an operator or action system, (iv) a subsystem that builds up a dynamic internal representation of sensory events, (v) the algorithm that designs action plans, and (vi) a decision system (see Fig. 1). Details of the algorithm are given in [12].

The dimension reducing system Tremendous amount of sensory input information coming from the external world should be reduced to a set of 'high' or 'low' bits in order to build up an efficient internal representation. Systems considered here are categorizing systems of soft competition, such as the 'where system' [11, 4] with spatial filters that reduce the high resolution image to a few positions and thus we assume a grid system. The recognition of objects, the 'what' system, may be built by several methods, e.g. [2, 9, 8, 3, 7]. Both subsystems have bit string outputs and it is the task of the agent to find out the meaning of these bits. If preprocessing is not needed then the detector system itself may be considered as the input system.

The goal system The agent is equipped with a prewired goal subsystem. A 'goal' can be either 'high' or 'low'. If a goal is 'high' then the agent tries to set it 'low'. When it becomes 'low' then the actual event satisfies the goal and that is remembered. It is assumed that the goal system has its own sensors that set the goals 'high' or 'low'. When a goal is set 'low' we say that the goal was satisfied. It is worth mentioning that it is the point where the system may be connected to a knowledge based system: The goal system may initiate a set of actions, the "reflexes", i.e. the knowledge based system. In the present work such examples will not be considered.

Operator, or action system The system is equipped with a set of inherited operators, that allow to make macroscopic changes in the environment. It is assumed that these operators fit the grid of the 'where' system. Operators may be set 'high' or 'low', i.e., may be activated or deactivated.

Internal representation of events The system collects 'events' of the world. An event is composed of a given internal representation of the sensory input (I_1), the operator (O) applied in state I_1, and state I_2 that the system reached after executing operator O. The collection of these three entities shall be called triplet. During the course of life the agent experiences chains of triplets: one event is followed by another. We collect these in the form of a directed graph where nodes represent events, and arrows correspond to the flow of time. This representation reflects that operators are not mappings on the states but relations according to the H_2 principle. The representation is a doubly weighted directed graph (or a directed neural network), where weight of node $I_1 O I_2$ is upgraded to tell how reliably takes operator O from initial state I_1 to final state I_2. Weight of directed arrows between triplets $I_a O_1 I_b$ and $I_b O_2 I_c$ reflect how often operator O_1 is followed by operator O_2. The directed graph of events is called long term memory (LTM). At this point the system becomes general and may be reformulated as follows: The sensory system provides information in the form of 'high' and 'low' bits and operators of the system are capable of changing the sensory information. The dimension reducing input system may be considered as an illustrative example for cases where the sensory information is too large for direct evaluation.

The short term memory (STM) Memory saving requires the introduction of a temporary buffer, the STM. Events of the world as detected by the preprocessing units are memorized in a first-in-first-out finite length buffer. If the goal is reached then information contained by the STM is transcibed to the LTM. The LTM grows from the direction of events that reach the goal: In every case of success, i.e., when a goal was just satisfied, and if the chain of events in the STM does not exist in the LTM in full, then the existing part of the LTM chain is lengthened according to the STM. The cognitive space (CS) is made of the LTM and the STM.

Designing action plans and decision making When a goal becomes 'high' then activation is spread from triplets, that had satisfied that goal, opposite to the arrows of the directed graph. Spreading activation is monitored in nodes that have initial states identical to the present state. The highest activity node is chosen and activity is compared to a threshold. If the spread activity is below threshold then an action is chosen randomly. Otherwise the operator of the node is executed, the new state is experienced, and LTM is updated. Activation spreading is the means to reach low time complexity. It may be worth noting that activation spreading could have been implemented in the opposite way, that activation is spread from triplets that have initial states identical to the present internal representation. States with active goals could be the sinks of the activation spreading. If activation spreading that conserves the flow [6] is utilized then the node with the largest outgoing flow may be chosen in decision making. It has been shown [10] that activation spreading and also the network upgrading may be put in the form of adaptive real-time dynamic programming [1] and thus DCM can find the optimal policy for the class of Markovian Decision Problems.

3 EXPERIENCES WITH DCM

Computer experiments led to the following conclusions [12]: (i) The length of the STM should be kept small to avoid complicated representations and to save memory. Short STM helps to find optimal solutions. However, short STM restricts the length of the problems the system can solve. One solution that can overcome this problem is operator fusion where operators that

follow each other frequently may be unified into a new operator[1]. (ii) It is important to save rare events. In DCM a special form of postsynaptic learning was introduced for this purpose. (iii) The system works in probabilistic worlds. (iv) The system can work in conditionally probabilistic worlds, when the probability of the next state depends on previous states. (v) In the intermediate state of development cycles can arise. That and the safe discovery of interaction underlines the need for exploration. (vi) Information collected in a given world is given in a distributed form in the LTM. The representation is characteristic of the world and is of restricted use in another world. The root of this problem is that features of an unseen world will not fit previous experiences and the spreading activation will not have any route. Two methods come to mind to solve this problem: (a) Provide an inherited 'similarity measure' defined over the set of states that can group states. Predefined measure, however, restricts adaptivity. (b) Find a method that can extract the important features of the states for reaching the goal(s): the system needs generalization capabilities.

4 FROM RELATIONS TO MAPPINGS

It was noted that operators are just relations, not mappings on the space of states: Thus the result of an action is in general ambiguous. It is then attractive to transform the space of states in order to make operators mappings on the transformed space. If an operator becomes a mapping and the transformation has kept the learnt information to reach the goals then we say that 'all the important features are kept'. Important information then may be viewed es *'emerging concepts'* and the transformation as concept formation. This is in accord with principle H_2.

Our working hypothesis about the world and the agent is that 'emerging concepts' are *'quasi independent'*, i.e. the effect of operators may be estimated by considering just a few concepts in a given situation, or state. A concept C may be given as a subset of $\mathcal{I}$ ($\mathcal{I}$ is the space of all possible states.) In our case states are given as elements of the set $\{0,1\}^n$, where n is a fixed positive integer. Here, we restrict the set of possible concepts to Holland's Universe of $\mathcal{I}$, i.e., to the $\{0,1,*\}^n$ where string $''01*''$ denotes, e.g., the two element set $\{''010'',''011''\}$.

[1]Such method may be worked out, e.g. with the help of the genetic algorithm [13].

5 CONCEPT GENERATION

Concepts are important features to reach the goal and are thus formed from the direction of the goal. Creating an important feature to reach the goal may be viewed as the creation of a subgoal. That is, goals can be considered as the zeroth level concepts. A heuristic algorithm may be given as follows. First, a triplet that reaches goal g is viewed as IOC_g, where $C_g \subset \mathcal{I}$ is the set of states that satisfies goal g. Now, let us take two nodes (N_1, N_2) of the LTM that satisfy the following conditions: (i) they have the same operator, (ii) they have the same ancestor A on the directed graph, (iii) the ancestor is a generalized triplet of COC' form, where C is a non-trivial concept, and (iv) the ancestor is reliable (see Fig. 2). In this case a new node N is added to the LTM subject to all maintenance procedures detailed in [12]. New connection set, being the union of the connection sets of the old nodes, is generated to N. Initial concept of N is the 'union' of the respective initial concepts of the old nodes. 'Union' of concepts (and thus simple states too) is constructed by saving identical bits and replacing different bits by $''*''$. N has the joint operator of the old nodes. Final concept of N is the initial concept of the ancestor. The algorithm may start, if triplets that satisfy a goal are thought of as having an ancestor, the so called goal triplet. The described procedure is called generalization and is thought to proceed parallel with the base algorithm.

6 RESULTS ON GENERALIZATION

Agents and environments were simulated for the generalized DCM. The environment was a grid. As it appears to the agent at every grid point an obstacle or food may be present. Operators of the agent were 'up', 'down', 'turn left', 'turn right' and 'suck'. Time to time the goal 'eat' went high. The inherited rule is that the goal 'eat' goes 'low' if the agent eats a food, i.e. stops in front of the food and applies operator 'suck'. It is assumed that a sensor detects if its belly is filled and the negate of the output of the sensor is identified as the goal itself. The agents' sensors provide perfect information about a restricted part of the environment. The agent could see a 3 by 3 part of the grid with itself in the middle.

Comparisons were made between agents without generalization (w/oG)

and with generalization (wG) capabilities. Both agents were trained under the same circumstances. Performances, i.e., steps needed to reach the goal as well as memory requirements were compared. The agents could collect knowledge in different, randomly generated worlds. At the end of training in each world the agents were tested in a predefined test world (see Fig. 3). Parameters of the random worlds were the obstacle and food densities. The test was conducted in the most dense world. Figure 4 shows the performances of a random machine, and agents with and without generalization. Notice, that *the performance of the wG agent is near optimal* since on avarage 4 steps were needed to reach the goal in the test world. The less dense the training worlds are the larger the performance difference is between agents if enough time is given for training.

Other important differences may be found in the size and the structure of the LTMs (see Fig. 5). In the four step STM case the wG agent has, in essence, a six member memory on the top two levels. (The top level consists of events that reached the goal.) Below that level the branching ratio is high. On the other hand, the w/oG agent has a low branching ratio memory at every level with a larger number of memories at higher levels. The size of the memory is much smaller for the wG agent at levels where generalization was efficient.

The generalization replaces bits of the input by 'don't care' ($''*''$) signs. It may be asked how large part of the input could be generalized at a step. According to the computer simulations 'broad' generalizations, i.e., generalization of a large number of bits, are less efficient. When a larger number of bits can be generalized in one step then small generalizations are not made. This leads to overgeneralized triplets that cannot be further generalized without the loss of all the information. The situation leads to customs or 'superstitions' that tend to lengthen the average number of steps to reach the goal.

7 CONCLUSIONS

The starting point that neurons can be considered as building blocks of knowledge based systems as well as the basic units of neural networks and the observation that adaptive autonomous agents need a backing 'rule based system' in order to develop their own, were explored. The solution suggested

is to build internal representation of events and to apply activation spreading on the internal representation for creating dynamic concepts and for coming to decisions. This unique combination results in systems of low time complexity. In addition, there is a possibility for generalization allowing substantial decrease of memory requirements as well as functioning in unseen environments.

References

[1] A.G. Barto, S.J. Bradtke, and S.P. Singh. Real-time learning and control using asynchronous dynamic programming. Technical report 91-57, Computer Science Department, University of Massachusetts, 1991.

[2] G.A. Carpenter and S.A. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37:54–115, 1987.

[3] P. Földiák. Learning invariance from transformation sequences. *Neural Computation*, 3(2):194–200, 1991.

[4] T. Fomin and A. Lőrincz. Robustness of Hebbian and anti-Hebbian learning. *Neural Network World*, 4:699–717, 1994.

[5] Zs. Kalmár, Cs. Szepesvári, and A. Lőrincz. Generalization in an autonomous agent. In *Proceedings of IEEE WCCI ICNN'94*, Orlando, Florida, June 1994.

[6] G. Lei. A neural model with fluid properties for solving labyrinthian puzzle. *Biological Cybernetics*, 64(1):61–67, 1990.

[7] J. A. Marshall. Development of perceptual context-sensitivity in unsupervised neural networks: parsing, grouping, and segmentation. In *Proceedings of the Int. Joint Conf. on Neural Networks*, volume 3, pages 315–320, 1992.

[8] J. Rubner and K. Schulten. A self-organizing network for complete feature extraction. *Biological Cybernetics*, 62:193–199, 1990.

[9] J. Rubner and P. Tavan. A self-organizing network for principal component analysis. *Europhysics Letter*, 10:193–199, 1989.

[10] Cs. Szepesvári. Dynamic Concept Model learns optimal policies. In *Proceedings of IEEE WCCI ICNN'94*, Orlando, Florida, June 1994.

[11] Cs. Szepesvári, L. Balázs, and A. Lőrincz. Topology learning solved by extended objects: a neural network model. *Neural Computation*, 6(3):441–458, 1994.

[12] Cs. Szepesvári and András Lőrincz. Behavior of an adaptive self-organizing autonomous agent working with cues and competing concepts. *Adaptive Behavior*, 2(2):131–160, 1994.

[13] G.J. Tóth, Sz. Kovács, and A. Lőrincz. Genetic algorithm with alphabet optimization. *Biological Cybernetics*, 1995. in press.

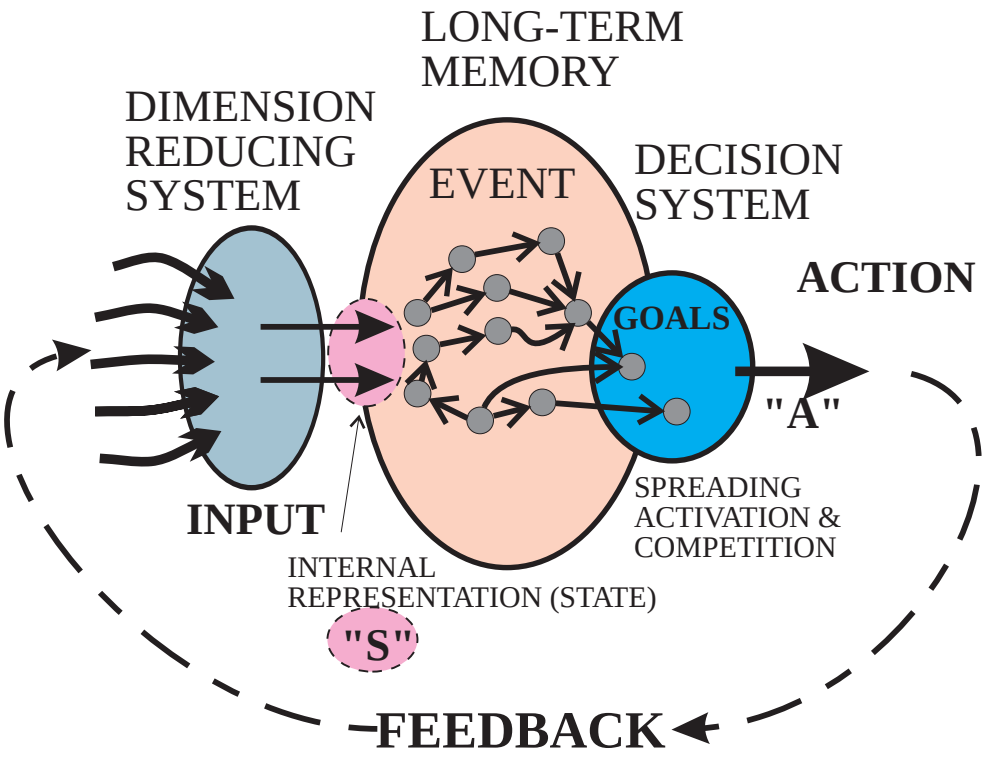

Figure 1: Architecture of the agent

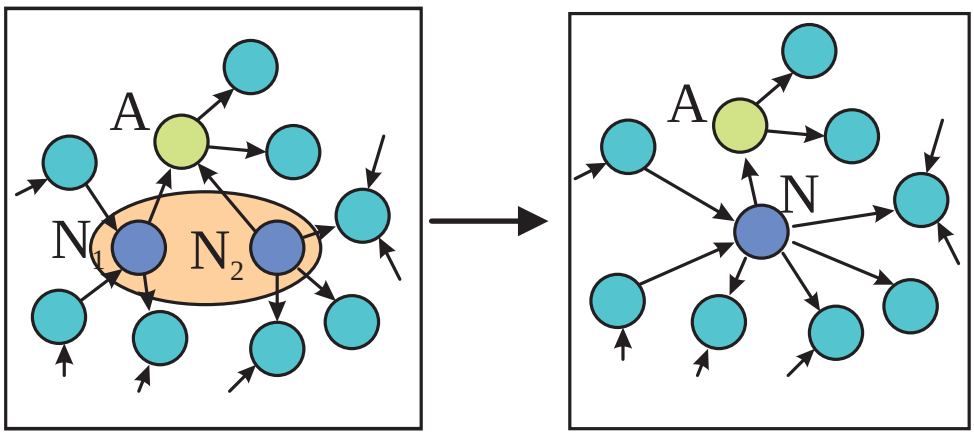

Figure 2: Generation of a new concept with generalization

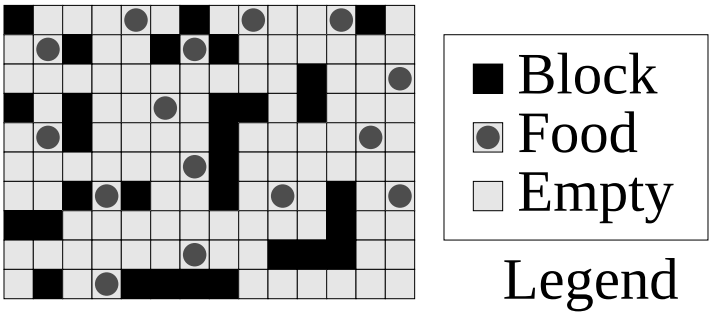

Figure 3: The test world

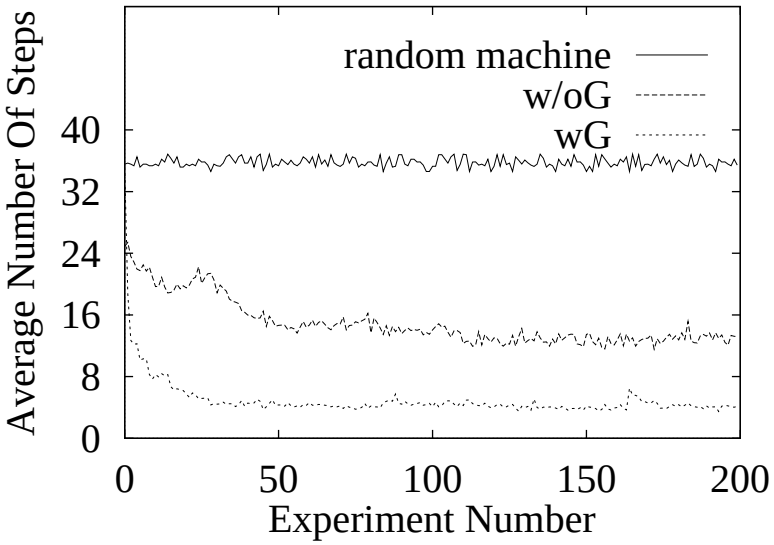

Figure 4: Performances of agents

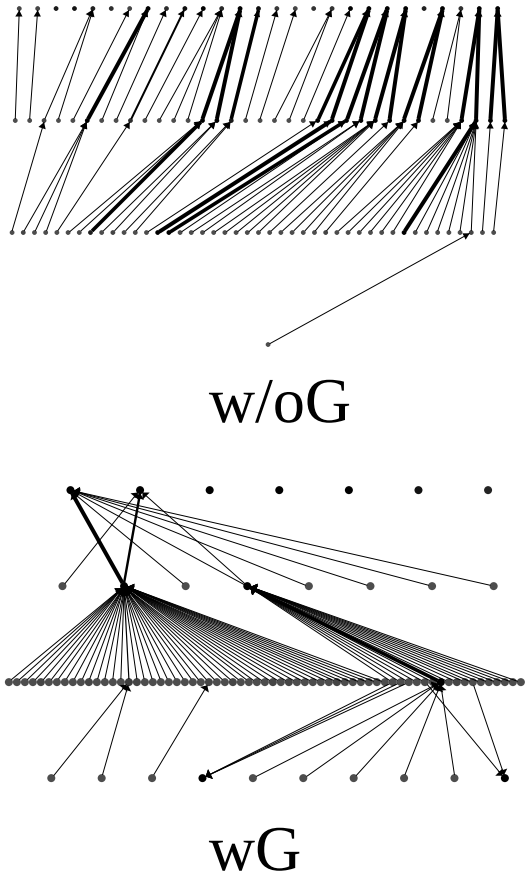

Figure 5: Memory structure of agents without and with generalization