A Current-Mode Analog Circuit for Reinforcement Learning Problems

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td><a href="http://dx.doi.org/10.1109/ISCAS.2007.378410">http://dx.doi.org/10.1109/ISCAS.2007.378410</a></td>
</tr>
<tr>
<td>Publisher</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>Version</td>
<td>Final published version</td>
</tr>
<tr>
<td>Accessed</td>
<td>Mon Feb 04 02:13:02 EST 2019</td>
</tr>
<tr>
<td>Citable Link</td>
<td><a href="http://hdl.handle.net/1721.1/52597">http://hdl.handle.net/1721.1/52597</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>Article is made available in accordance with the publisher’s policy and may be subject to US copyright law. Please refer to the publisher’s site for terms of use.</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td></td>
</tr>
</tbody>
</table>
A Current-Mode Analog Circuit for Reinforcement Learning Problems

Terrence S.T. Mak, K.P. Lam, H.S. Ng
Department of Systems Engineering & Engineering Management
The Chinese University of Hong Kong, Shatin, Hong Kong

G. Rachmuth, C.-S. Poon
Harvard-MIT Division of Health Sciences and Technology
MIT, Cambridge, MA 02139, USA

Abstract—Reinforcement learning is important for machine-intelligence and neurophysiological modelling applications to provide time-critical decision making. Analog circuit implementation has been demonstrated as a powerful computational platform for power-efficient, bio-implantable and real-time applications. In this paper, we present a current-mode analog circuit design for solving reinforcement learning problem with a simple and efficient computational network architecture. Our design has been fabricated and a new procedure to validate the fabricated reinforcement learning circuit will also be presented. This work provides a preliminary study for future biomedical application using CMOS VLSI reinforcement learning model.

I. INTRODUCTION

Reinforcement Learning is a putative model of learning rewarding and punishing predictions based on environmental stimuli [16], [3], [7], [2]. The learning occurs through updating expectations of the reward in proportion to prediction error, such that across trials, the expected reward converges to the actual reward. Reinforcement learning has been a proven powerful optimization and learning algorithm with abundant applications across a number of engineering and machine intelligence domains [8]. For example, in autonomous robots route planning, reinforcement learning is implemented in a software-based computer to evaluate the environment and make real-time decisions. Such computation is highly computational intensive and often requires numerical approximation techniques to improve computational speed. A hardware-based reinforcement learning system would circumvent the limitations of software-based reinforcement learning implementation.

Besides, reinforcement learning has also been thought as a general modeling methodology for natural rewards systems based on the observations about the ability of animals to learn appropriate actions in response to particular stimuli on the basis of associated reward. Recent neurophysiological evidence described the infusion of biological findings [11], [13], [4], [9] in dopamine function with quantitative models derived from theories of reinforcement learning. Also, in [12], Redish proposed a computational TDRL (temporal-difference reinforcement learning) model from a theoretic view-point, in which a noncompensable drug-induced dopamine as biased rewards signals would result in an over-selects actions leading to drug receipt. These findings suggest that hardware reinforcement learning and bio-implantable system, which are comprising properties of highly portable and extreme low-power consumption, are required for future neuro-prosthetic biomedical applications.

Analog circuit implementation has been demonstrated as a powerful computational platform for power-efficient, bio-implantable and real-time applications. In this paper, we present a current-mode analog circuit design for solving reinforcement learning problem with a simple and efficient computational network architecture. Our design has been fabricated and a new procedure to validate the fabricated reinforcement learning circuit will also be presented.

The paper is organized as follows: in section II, the background of reinforcement learning problem and its formulation are presented. In section III, the current-mode circuit design and simulation results are presented. Fabricated chip design and testing results are presented in section IV.

II. BACKGROUND

In general, a model of an environment, which defines the reward functions and transition probabilities between states, may not be always available and therefore leads to the two alternatives to proceed, the model-free and the model-based reinforcement learning. For the model-free approach, such as TDRL, a model of environment is not known in advance. Throughout a number of trials and interactions against the environment, optimal expected rewards and policies can be approximated. Although the model-free approach does not require prior information about the environment, this model is usually slow to give convergence [15] and requires a large number of interactions with the environment which may be costly.

In contrast, the model-based Bellman’s equation gives the exact solution and requires no learning because it assumes full knowledge of the model of environment, which can be obtained by probabilistic model estimation and identification. For accelerated convergence under certain circumstances, a hybrid solution is also viable by a combination of model-based and model-free approaches. For instance, optimal cost estimates can first be obtained from the Bellman equation using approximate probabilistic information and rewards of the environment; these estimates are then used in iterative TDRL-type learning to give more precise converged values.
The objective of the model-based reinforcement learning is to find a set of actions that can maximize the received reward in a long run based on the given model. Particularly, it can be formulated as a shortest path problem, in which the objective of the formulation is to find a set of actions that lead to the shortest path under a deterministic environment. Consider an $N$-node single-destination shortest path problem, where the model is defined as a set $S$ of nodes $\{s_i, i = 1, ..., N-1\}$ have possible actions $\{a\}$ in the set $A(s_i)$ to nodes $\{s_j, j \neq i\}$ with costs $C_{s_is_j}$ and $s_N = K$ is the destination node. The Bellman-Ford algorithm [1], [6] defines a recursive procedure in step $k$, to find the new estimate $\hat{V}^{(k)}(s_i)$ for the expected shortest path cost from $s_i$ to the destination $K$ using the previous estimates in step $k-1$, known as dynamic programming.

$$\hat{V}^{(k)}(s_i) = \min_{a \in A(s_i)} \{C_{s_is_j}^{a} + \hat{V}^{(k-1)}(s_j)\}$$  \hspace{1cm} (1)

A continuous-time extension of the recursive Bellman-Ford algorithm can then be formulated from Eq.(1), to give

$$\frac{dV_{s_i}(t)}{dt} = -\lambda_{s_i} V_{s_i}(t) + \min_{a \in A(s_i)} \{C_{s_is_j}^{a} + V_{s_j}(t)\}$$  \hspace{1cm} (2)

where $V_{s_i}(t)$ represents the approximated $\hat{V}$ in the continuous time, and $\lambda_{s_i}$ is a first-order lag constant which is implementation-dependent. Notice that when $dV_{s_i}(t)/dt$ is approximated by the discrete-time finite difference $V_{s_i}(t+1) - V_{s_i}(t)$ and $\lambda_{s_i} = 1$, Eq.(2) reduces back to Eq.(1). Eq.(2) provides a natural setting for parallel computational circuits, without the need of imposing sequential constraints as in the traditional digital Bellman-Ford equation (Eq.(1)).

### III. IMPLEMENTATION AND RESULTS

#### A. Current-Mode Implementation

Analog voltage-mode implementation using the binary relation inference network (BRIN) [10] for solving the dynamic programming was previously reported. In line with the previous work, with a simpler and more effective current-mode design using only loser-take-all (LTA or min) circuits [5], [14] previously developed for neural-fuzzy systems, to solve the shortest path problem.

The right hand side of Eq.(1), in which a minimum operation is performed over all possible actions from node $s_i$, provides the necessary framework for using the LTA in a dynamic programming unit for solving an $N$-node directed graph problem (e.g., a reference 8-node graph is shown in Fig. 1(a)). If the cardinality of $A(s_i)$ (or the out-degree of node $s_i$) is $n_{s_i}$, then a straightforward current-mode realization of Eq.(2) is to have an LTA (or min) circuit over its $n_{s_i}$ input sites, by simply joining $C_{s_is_j}$ and $V_{s_j}(t)$ at each site. As there are $N-1$ shortest path costs for $\{V_{s_i}(t), s \neq K\}$, a current-mode realization of an $(N-1)$ LTA-circuit array is depicted in Fig. 1(b) for an $N = 4$ directed graph problem (where the input sites of each LTA are limited to $n_{s_i} = 4$ in our chip implementation). As a simple illustrative example of an LTA unit for realizing $V_{s_i}(t)$, node $J$ has two outgoing paths: a direct one to the destination $K$ with cost $d_{JK} = x$, and another one with cost $d_{JH} = y$ to node $H$. Notice that for each of the other two remaining sites, a high current value ‘infin’ (e.g. the upper limit of the operation range) is required to maintain the correct min function of the LTA.

A reasonable LTA-circuit array size was considered for our proof-of-concept chip implementation. Fig. 1(a) is an 8-node directed graph, for which the shortest paths from the nodes $J, H, I, G, E, D, C$ to the destination node $K$ can be readily obtained from a connected array of 7 LTA-circuit units. The graph also consists of a 4-node subgraph $\{K, J, H, I\}$ which provides a simpler evaluation platform for 3 LTA units (shown in Fig. 1(b)). For simplicity and without loss of generality, we chose to take a systematic approach in considering two cost variables ($x$ and $y$) irrespective of the graph size. The idea is to consider a class of problems using $x$ for all the feedforward paths (pre-wired within the chip), while providing some $y$ feedback loops to investigate the decision changes made by the LTA units for verification.

A conventional 2-stage binary-tree circuit design (Fig. 2) for a 4-input LTA was used for our current-mode chip. Vittoz’s [17] 2-input minimizer combines NMOS and PMOS transistors to carry out addition and subtraction of replicas of two currents, $A$ and $B$. $A - B$ is then used in a 2-transistor arrangement to give $\max(A - B, 0)$ (hence blocking any negative input to 0). The output $\min(A, B)$ is obtained by subtracting $A$ from $\max(A - B, 0)$. Our initial design consists of two Vittoz’s 2-input minimizers in the first stage, and the second stage is a 2-input minimizer acting on the two outputs from the first stage. More efficient single-stage, high-precision designs for $n_{s_i}$ inputs [5], [14] may further enhance performance. Each of the input assumes the joining of proper

![Fig. 1. (a) An $N = 1$ LTA-circuit array for solving deterministic shortest path problem ($N = 10$). (b) Site function and unit function of dynamic programming unit.](image-url)
current sources representing $C_{s_1s_2}$ and $V_{s_1}(t)$. In current-mode design, an adequate number of current outputs are readily replicated using current mirrors for network connectivity and observability.

B. Results

Fig. 3 shows the simulation results for solving the 10-node shortest path problem by using the circuit design presented at previous section. The evolution of current outputs from each computational units were shown at the upper figure while the average relative error of the computation was shown in the lower figure. The circuit converges rapidly, as the relative error decreased to less than 10% in 100 ns.

![Fig. 3. Simulation results of the circuit for solving the 8-node shortest path problem in Fig. 1(a). (upper) Evolution of current outputs from different computational units. (lower) Average relative error of the analog circuit computation and the lines are the standard deviations.](image)

IV. CHIP IMPLEMENTATION AND TESTING

Our CMOS chip was fabricated by MOSIS using the AMI 1.5 μm process on a standard 4.6mm x 4.6mm die with 116 pins. For chip testing, a large number of voltage-controlled current sources were built in-house using matched 100 KΩ / 10KΩ resistor pairs for the differential inputs of an operational amplifier (LM324). In dealing with the 4-node subgraph problem, a total of 12 current sources (3x, 2y, and 7 ‘infin’) were used; while the 8-node graph problem required 28 current sources (11x, 4y, and 13 ‘infin’). Current measurements in the μA range were made using a transimpedance CAM (configurable analog module) of the Anadigm FPAA (field programmable analog array). Appropriate calibrations were also performed on both the current-measurement and the current-source setup using an HP 3245A universal source.

![Fig. 4. Dynamic test of LTA.](image)

Dynamic tests were performed on individual 4-input LTA units of our fabricated chip using a varying input on one site, while keeping the other site inputs constant. A typical example is provided in Fig. 1(b) for LTA unit $J$, showing a case for the varying Input 1 and the measured $J$ output: Input 2 is 30 μA, Inputs 3 and 4 are voltage-controlled at 2.3V (to give 11.3329 μA). The LTA-circuit performs well, giving the minimum output as those Input 1 that are smaller than Inputs 3 and 4. The output $J$ is shown to behave like a first-order lag (including that of the lag of the current-measurement device) with respect to step changes. The dynamic tests are useful for selecting the functional LTA units of the fabricated chip for subsequent network connection; and more detailed timing measurements can be performed to deduce the computational delay of the circuits.

Static tests were performed on two graph problems specified by Fig. 1: a smaller 4-node graph with a connected 3 LTA-circuit array for \{J, I, H\}, and an 8-node graph with a 7 LTA-circuit array for \{J, I, H, G, E, D, C\}. In each static test, the $y$-inputs and ‘infin’-inputs were fixed while the $x$-inputs vary over a range of values. The various measurable LTA unit outputs were then measured at a set of $x$-input values, and were compared with that obtained from theoretical analysis and prediction. For the 4-node subgraph \{K, J, H, I\} of Fig. 1(b), the $J$ and $I$ outputs are expected to be $\min\{x, 2y\}$ and $\min\{2x, x + 2y\}$, respectively. A very nice experimental curve is obtained in Fig. 5, showing that both $J$ and $I$ are able to make the necessary decision changes at around 1 μA (which matches well with the value of 2 times the $y$ current setting at 0.5011 μA). The curve slopes are also useful to convey information, i.e., $J$ becomes flat after $x > 2y$ because its
minimum path cost is $2y$, and $I$’s slope switches from 2 to 1 because the minimum path costs for $I$ are $2x$ (for $x \leq 2y$) and $x + 2y$ (for $x > 2y$), respectively.

Referring to the 8-node graph of Fig. 1(a), the expected outputs of $J$ and $I$ are obtained as $\min\{x, 2y\}$ and $\min\{2x, x + 2y, 2y\}$, respectively. Fig. 6 shows the output curves for nodes $J, I, E, D$ and $C$; and it is interesting to compare the $J$ and $I$ curves of Fig. 5 with 6. Because of an additional $y$ feedback loop at node $I$, $I$ makes the correct decision change near $x = y$, and $J$ makes the change near $x = 2y$. They both have a minimum path cost close to $2y$, but for two different paths, i.e., $I \rightarrow G \rightarrow K$ versus $J \rightarrow H \rightarrow K$. These results are as expected (within the prescribed error tolerance for analog circuits) from the problem specification and validate the network performance. The expected output values of the other nodes can readily be obtained from further analysis, e.g., $\min\{3x, x + y\}$ for $E$, and $\min\{4x, x + y\}$ for $D$. The curves for $E, D,$ and $C$ show the expected rise with a slope of 1 for large $x$; but their initial slopes for small $x$ are less than the predicted values due to measurement errors and transistor mismatch.

V. CONCLUSIONS

We have developed a novel analog VLSI chip for computing shortest paths using current-mode techniques, which alleviate the need of ‘addition’ circuitry and reduce the computational delay associated with previous voltage-mode designs. New procedures for dynamic and static tests were proposed and successfully demonstrated to validate such type of dynamic programming circuits. The proposed design could be used for future biological modeling of reinforcement learning and time-critical biomedical applications.

Acknowledgement: The support of MOSIS educational program (research) for chip fabrication is gratefully acknowledged. T. Mak acknowledged the support of Croucher Foundation. G. Rachmuth was a recipient of the US National Defense Science and Engineering Graduate Fellowship. C.-S. Poon acknowledged the support of NIH grant EB005460.

REFERENCES