ASIAN JOURNAL OF PHARMACEUTICAL AND CLINICAL RESEARCH

INNOVARE ACADEMIC SCIENCES
Knowledge to Innovation

Special Issue (April)

Online - 2455-3891
Print - 0974-2441

Full Proceeding Paper

MATHEMATICAL ARCHITECTURE OF MICROSERVICES FOR GEOGRAPHIC INFORMATION SYSTEM BASED HEALTH MANAGEMENT SYSTEM

ANKUSH RAI*, JAGADEESH KANNAN R

School of Computing Science and Engineering, VIT University, Chennai, Tamil Nadu, India. Email: ankushressci@gmail.com

Received: 28 December 2016, Revised and Accepted: 10 May 2017

ABSTRACT

Objective: Despite the wide adoption of distributed computing with several webs standards and cloud technologies; the building of city-wide health management system for smart city platform is a daunting task.

Methods: Owing to the limitations in sparse learning of disease outbreak and its dynamic nature. As it would require the development of a scalable, distributed and evolving architecture on the web; where a sparse machine learning based algorithm will enable authorities collaborate in preventing, controlling, and responding to a specific disease outbreak and its time factor analysis. In this work, a mathematical system model is presented for sparse learning in geographic information system based architecture to support continuity of microservices in health management setting.

Results and Conclusion: The model can continually cope with the transformation in architecture to match with the system goals of microservices and anticipate the evolutionary aspects of the architecture configuration.

Keywords: Microservices, Geographic information system, Health management, Smart city.

INTRODUCTION

The main problem in establishing a scalable coordination between distributed microservices is to solve the issue of the high dimensional semantic decision [1,5]. All autonomous industrial services working collaboratively through distributed microservice platform (DMP) will require a close loop iteration; which is divided into three essential steps:

i. Distributed sensing from the environment [3].
ii. Performing local computation of the sensed data.
iii. Fusion the computed data from several distributed settings to perform global actions and communicating with other end to end devices [4].

Industry 4.0 supports the integration of manufacturing system on a global scale with several advances to achieve simultaneous communication, computation and microservice in manufacturing domain [1]. From a system level point of view, DMP is crucial to achieve a scalable holonic system; wherein small subsystems will be programmed in such a way that it will not only sustain the manufacturing on itself but additionally will be able to collaborate with other subsystem to achieve a global scale industrial application. Such subsystem microservice should avail overall coordination between industrial services.This requires enhancing machine level decision-making process, adaptive sharing of resources, developing a matrix of specificity of actions as per the varying context [4-7]. This would reduce the cost of product manufacturing, manufacturing life cycle, resource utilization [2,8]. Since the efficiency of the scalable services is heavily dependent on the collaborative process of disseminating data and its contextual analysis. Thus, weaving such a cyber-physical system is computationally expensive in high dimensional relationship between sensed data and its associated microservice actions [9,10]. Furthermore, the rising trend of on-demand dispatch of microservice is a predicament owing to its complexity in modeling. A good solution is to integrate the problem of modeling scalable system and its distributed microservice features within the same framework.

METHODS

In this section, we present an algorithmic framework to achieve scalable distributed learning and decision-making to model global coordination between industrial services. This self-modeling approach will enable high learning rate for high-dimensional on demand microservice. The system is tested and validated in a virtual manufacturing setting.

The first step is to define the synchronous machine model of distributed systems with its weighted sensed data x_{ij} and microservice action sets y_{ki}. Here, we are using membrane computing-based model of neural system to define the correlation between the sensed data and the microservice action such that the final sets derivable would be optimized and weighted to achieve optimization for high dimensional decision space, then in the next step, it shall be forwarded to semantically filter out optimal policy (i.e., correlated state action pair) with higher reward through the help of distributed reinforcement learning.

Now, for the first step, let us supposed that we have a total of excitatory neurons (E_N) and inhibitory neurons (I_N) where they are in the ratio of $I_N=0.2\times E_N$. Now, that we need to find an evolutionary Hodgkin-Huxley equation for self-managing neurons. Therefore, to model the phenomenon of building the learning model for biological neurons, we require to merge the properties of artificial neural network with the biological neurons. Where the sequence of inputs of the firing neurons is affects the other subsequent sequence and consequently synapse formation before giving a unitary idea of stimuli. Thus, W_{ij} be the weightage for the connection strength from neuron i to neuron j, similarly W^{IE}, W^{EE}, and W^{EI} represents weightage for inhibitory to excitatory connections, excitatory to excitatory, and excitatory to inhibitory connections, respectively. The W^{EE} and W^{EI} are initialized as sparse random matrices with the range of connection probabilities between the value of 0.1 and 0.2. Initially, the W_{IE} connections are meant to freeze at their random initial values which are depicted from uniform distribution, latter followed by normalization [13,14]. Altogether, the sum of connections entering a neuron is in a sequence of 1 and 0; thereby the binary vectors is given by $x(E_N) \in \{0, 1\}$ and

$y(I_N) \in \{0, 1\}$ for the excitatory and inhibitory neural activity at time step t, respectively. Hence, the sequencization of the network states at time step t+1 is equivalent to:

$$x_{ij}(t+1)=\theta\left(\sum_{j=1}^{E_N} W_{ij}^{EE}(t)x_{ji}(t)-\sum_{k=1}^{I_N} W_{ik}^{EI}(t)y_{ki}(t)-T_{ij}^{E}(t)+\xi E_i(t)\right)$$

$$y_{ki}(t+1)=\theta\left(\sum_{j=1}^{N^E} W_{ij}^{IE}x_j(t)-T_i^I(t)+\xi I_i(t)\right)$$

As the network equation continues to evolve x_{ij} and y_{ki}, the synaptic weights is given as:

$$\Delta W_{ij}^{EE}(t)=\eta\left(x_{ij}(t)x_{ji}(t-1)-x_{ij}(t-1)x_{ji}(t)\right)$$

$$\Delta W_{ij}^{EI}(t)=-(1-\eta)y_{ji}(t-1)\left(1-x_{ji}(t)\left(1+\frac{1}{\mu_{ji}}\right)\right)$$

ϴ is the Heaviside step function; TE and TI are the threshold values for excitatory and I_N, where it is initially drawn from the uniform distribution within the interval $[0,T_{max}^{E}]$ and $[0,T_{max}^{1}]$. ξE_i (t) and ξI_i (t) are white Gaussian noise processes with $\mu_\xi \in [0.01, 0.05]$. Here, one-time step corresponds roughly to the duration of window of the spike time dependent plasticity. η is the learning rate. Now, that the threshold value of the E_N in response for a sequence of activated neurons is made a pass through the previously generated targeted sequence code blocks of firing neuron states $S_{ij}^{code\ block}$; which is determined by the adaptation rate ηAD as:

$$T_{ij}^{E}(t+1)=T_{ij}^{E}(t)+\eta_{AD}\left(x_{ji}(t)-S_{ij}^{code\ block}\right)$$

The inhibitory spike-timing dependent plasticity ji rule regulates the weights backward from inhibitory to E_N which stabilizes the amount of excitatory and inhibitory to drive sensory information through the excitatory microservice neurons. Therefore, the evolutionary dynamics of the neuronal membrane potential that mediates the excitatory and inhibitory sequences through the network of membranes is governed by the above-deduced equation.

Following the above step, the generated data need be forwarded to semantically filter out optimal policy (i.e., correlated state action pair) with higher reward through the help of following distributed reinforcement learning. A policy P is memory-less technique, i.e., it primarily depends only on the current state and not onto its history. Thus, a deterministic strategy P assigns each state a unique action. While taking after a strategy P, we perform at time t action a_t at state s_t and observe a reward r_t (distributed according to R_{MDP} [s,a]). and the next state s_{t+1} (dispersed according to $P_{S_t,S_{t+1}}^{MDP}(a_t)$). We consolidate the sequences of rewards to a single value called the return, and our goal is to maximize it. Hence, we concentrate our work to focus on discounted return, which has a parameter γ∈(0,1), and the discounted return of policy P is:

$$V_{MDP}^{P}=\sum_{t=0}^{\infty}\gamma^t r_t$$

Where r_t is the reward observed at time t. Since all the rewards are bounded by R_{max} the discounted return is limited by:

$$V_{Max}=\frac{R_{Max}}{1-\gamma}$$

For a sequence of pairs for state and action, let the covering time, denoted by C', be an upper limit on the number of state-action pairs beginning from any pair, until all state-action appears in the sequential arrangement.

$$\delta_i(x,y)=\frac{\sum(I_{xy}^{i}(t)-\overline{I_{xy}^{i}(t)})^2}{T}$$

Where, $I_{xy}^{i}(t)$ is the consequent frame with the location in the form of (x, y) for the frame at time t, $I_{xy}^{i}(t)$ averaged over information of all $I_{xy}^{i}(t)$ value for time t. Hence, STD for the frame comprising of local decision space (δ_i^0) and its multiscale decision space (δ_i^S) can be mapped as the quantitative information about its trajectory in continuous frame sequence can be derived from:

$$s_i=\sum(I_{xy}(t)-I_{xy}(t-1))^2$$

Now, we need to derive the symmetry of the semantic action table and its multi-scale decision space to optimize the classification process. A data point from the previous steps can be represented as an element with r items for which there is a sequence of m number of frames given by $X\in\{X_{x_0}{}^{(1)},X_{x_0}{}^{(2)},\&,X_{x_0}{}^{(n)}\}$. Here, X is the set of possible values of a frame. A frame could be a sequence of state-action pairs with a reward value. Content frames may overlap spatially, temporally, or both. Here, overlapping time windows is 2 seconds long and starts every 185 ms; with overlap of 15/16 are used as frames. Let us suppose that C, B, and Y be matrix of filtered output, Y be the matrix of filters for stimulant variable and response variable for each X, such that C=XBY. Then, C is a super frame of B. The length of a frame S is equivalent to the total number of frames in it and is denoted by |S|.

Now,

$$X_{x_0}=\begin{bmatrix} X_{x_0}{}^{(1)} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & X_{x_0}{}^{(r)} \end{bmatrix}, X_{x_0}{}^{(j)}=\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, j=1,2,...r$$

Where X_{x_0} is the covariance map from X which asserts to the association formed between the frames S(t) with that of stimulant and response variable. X_i is the position of the input record. C_l is the cluster value which contains various values from 1 to l.

Now, at each step we calculate:

$$S(t)=\sum_{i=-\infty}^{+\infty}\sum_{k=1}^{N_{SC}}C_{li}S_l(t-iT_s)$$

$$S_l(t)=\prod(t)e^{j2\pi f_l t}$$

$$\prod(t)=\begin{cases}1,0<t\le T_s\\0,t\le 0,t>T_s\end{cases}$$

Where C_{li} is the ith information symbol at the lth subcarrier (when output of one iteration is propagated to the input of the other), T_s is the symbol period, S_l is the waveform for the lth subcarrier, N_{SC} is the number of subcarriers (number of matching iterations), f_l is the frequency of the subcarrier, and π(t) is the pulse shaping function. Following this process to complete the dataset in all records. Thus, the dynamics of the equation for a computational job using is computed as shown below: At time t, standard deviation is requested by every robotic device. For synchrony between industrial robots, we first give the outcomes for the synchronous Q-learning algorithm, where we overhaul every one of the sections of the Q capacity at every time step, i.e., the redesigns are synchronous. Let Q_T be the value of the synchronous Q-learning algorithm using polynomial learning rate at time T. Then with probability at least 1–δ, we have that $\|Q_T - Q^*\| \leq \varepsilon$, given that

$$T = \Omega\left\{\left(\frac{V_{\max}^2 \ln\left(\frac{|S||A|V_{\max}}{\delta.\frac{1}{1-\gamma}.\varepsilon}\right)}{\left(\frac{1}{1-\gamma}\right)^2 \varepsilon^2}\right)^{\frac{1}{\omega}} + \left(\frac{\ln\left(\frac{V_{\max}}{\varepsilon}\right)}{\left(\frac{1}{1-\gamma}\right)}\right)^{\frac{1}{1-\omega}}\right\}$$

$$Q_{C+1}(s,a) = \begin{cases} Q_C(s_i) + \alpha_t\left(S_l(t)\right)\left(R_{MDP}(s,a) + \gamma \max\limits_{b \in U(s')} Q_C\left(s',b\right)\right) & \text{if } s_i \text{ is validated} \\ Q_C(s,a), \text{ otherwise} \end{cases}$$

The above bound is somewhat complicated. To simplify, assume that ω is a constant and consider first only its dependence on T_{ij}^E. This gives us linear time complexity for the synchronous learning rate. Where symmetry breakdown allows us to ease the problem of extracting semantic rule by looking for the inter-correlation between symmetry of the state pairs and the symmetry. Hence, the relationship between it can be learned in one shot for rule generation, which is given as:

$$x(i,p) \leftarrow \left(\sum_{t=t(i,j)+1}^{t(i,j+1)-1} F_j(t)\right) - z(i,j) - w(i)$$

Here, x(i, p) is an indicator to the event that the solution is in state i during the pth phase of feature instance and n_i be the number of phases of state i. Thus, forming a dynamic sequence. The nodal degree distribution was fat-tailed with high-degree hub nodes to be located in the above-mentioned excitatory neural network using a sequence of information to excite the necessary regions and assess the information in an associative form. This enables several services all at once to not only learn but also it enables it to embark the cross relationship between various data for prediction or simulation based logical conclusion; herein, the processing is done over neural net-based shell environment. Computationally, this topology was embedded parsimoniously, in terms of the connection distance between co-activated nodes. Most connections or edges were separated by short sequence of excitatory data, significantly shorter than random networks; the parallel reinforcement learning equation is given as based on Instance of Window's Workspace W (b), Instance of machine's end U and the filtered action sets is given by AS_i with matrix model of tree of actions M_x. Compute the pointing correlation state P as:

$$P = \frac{1}{L_N} \sum_{p_i}^{L_W - 1} \left[\sum_{p_2}^{L_U - 1} S_{p_1,p_2}(t_i, f_1, f_2)\right]\left[\sum_{p_2}^{L_U - 1} S'_{p_1,p_2}(t_i, f_1, f_2)\right]$$

Where, L_N are the universal set of level for the microservice actions, p_i and p_2 are the adjoint sequence pairs with the levels L_W and L_U respectively, S_{p_1,p_2} and S'_{p_1,p_2} are the sets of sequence density constraint layout for the action sets positioning with its patterning saved in levels and between its intersection of adjoint pairs and the superpositioned pair density layout of differing state at the service's instance of the frame U. Furthermore, t_i is the collection of patterns for the weighted superposed state P_c (initially its value is set to 0), f_1,f_2 are the two delay frames with a minimal time delays t_i [11-13]. Thus, we calculate the tree of action based on continuous feedback loop:

$$M_x = \begin{pmatrix} t_1 \begin{bmatrix} P_1 \\ P_4 \\ P_8 \end{bmatrix} = AS_1 \\ t_2 \begin{bmatrix} P_3 \\ P_9 \\ P_6 \end{bmatrix} = AS_2 \\ t_3 \begin{bmatrix} P_2 \\ P_5 \\ P_7 \end{bmatrix} = AS_3 \\ \vdots \\ t_i \begin{bmatrix} P_0 \\ P_5 \\ P_c \end{bmatrix} = AS_i \end{pmatrix}$$

Where AS_i is the automated classified action sets. Again, to optimize the above-derived sequence of blocks we use membrane computing to carter distributed services with parallel Q-learning from several agents as mentioned below:

Here, C_{tar} is the desired target output and C_{out} is the actual network output. The value of C_{out} is determined as: $C_{out} = \left[Q_2^{(1)} Q_2^{(2)} \ldots . Q_2^{(N)}\right]$ where $Q_2^{(1)}, Q_2^{(2)}, \ldots, Q_2^{(N)}$ are the network outputs of each agent using reinforcement learning. The individual network outputs can be computed as:

$$Q_2^{(1)} = \sum_{r=1}^{N_g} w_{2r1} Q_1(r)$$

$$Q_1(r) = \frac{1}{1 + \exp(-w_{1r1}.C_{in})}$$

Where w_{2rl} is the weight of the connection from the 2rth input element to the 1th hidden unit. The above equation is a distributed function of several intermittent output layer and hidden layer, respectively. Adjusting the weights of all neurons by w = w+Δw, where Δw is the change in weight estimated as: $\Delta w = \gamma.Y_2.BP_{err}$, where γ is the learning rate. In general, the value of learning rate is between 0.2 and 0.5.

RESULTS AND DISCUSSION

Our methodology depends on distributed machine learning, sharing for networks and backings a wide range of operations in the middle of independence and collaboration with least suppositions on system availability. In particular, we added to a distributed derivation framework in view digital predicates that can catch the association with the physical world. In the fundamental distributed computing model, realities and objectives are spoken to as learning that can be shared craftily and aide the distributed thinking procedure. The response of chain growth in training was remained stochastic, but in our mathematical model of internetworked neurons, we have found that repeated simulations for training neurons changes the weights of the synaptic distribution and consequently forms a stable and strong connectivity within synaptic chain. Thus, the selection of postsynaptic

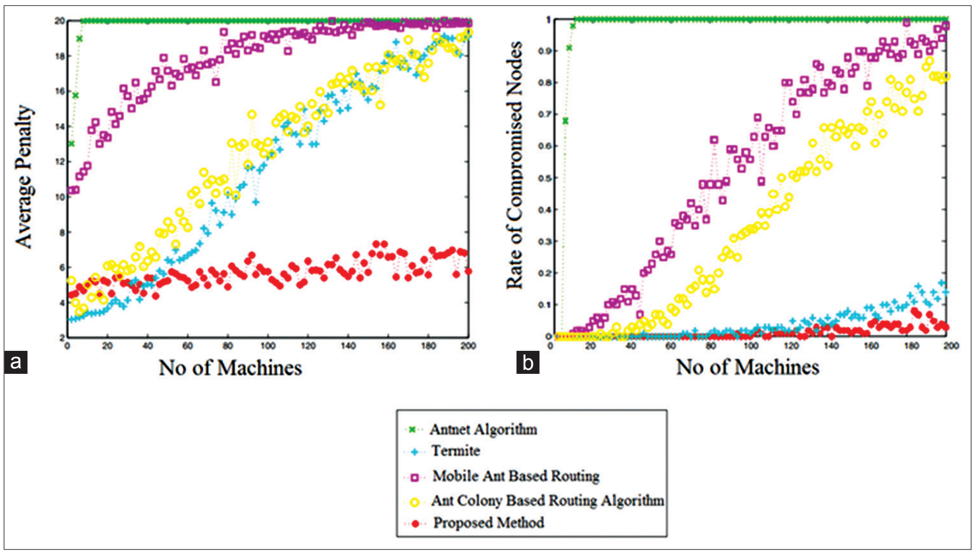

Fig. 1: Performance results and comparison. (a) Performance analysis of distributed learning where average penalty is taken as metric to determine the encounter of failure, (b) scalability analysis for the presented system

targets is crucial for the formation of loop of chains that stops its growth for the similar stimulation but keep on adjusting weights with chain growth emanating from the training neurons. Due to the spike time dependency plasticity rule the targeted neurons spontaneously spike shortly after the training neurons. It is observed that the training neurons spike synchronously and make convergent connections to the same sequential set of neurons and strengthens these connections. In this case, the duality in the middle of realities and objectives stretches out to the confirmation framework, which treats forward and in reverse thinking on an equivalent balance. Vital properties of our intelligent structure, for example, robustness, fulfillment, and end conditions have been set up under exceptionally broad conditions (Fig. 1).

A key component of the presented system is its dynamic and intelligent nature, implying that certainties speak to perceptions, and objectives can prompt changes in the environment that will show themselves as new actualities streaming into the framework. Once strengthened connections are developed the prominent sets of neuron spikes is readily evoked in these targets on every run of the stimulation. For the targets to overcome membrane noises, it is important that the synapses are cooperated through the convergent synapses. The next step follows for the closed loop of synfire chain is to propagate the firing chin to other neurons to recruit the new group in association with the previously recruited neurons. This iterative process yields stable topologies of synfire chains which are actively efficient in producing long stereotypical sequences of spikes for mediating training sets to other neurons; such that this chains consists of introductory sequence generated by training neurons in the first step and feeds this loop of strong synaptic connectivity to other pools of neurons, as network size is increased. Thereby, forming an interconnected network. Whether a unique neighborhood objective can be fathomed is frequently optional, in light of the fact that the consolidated impact of an arrangement of nearby objectives on the digital physical framework and its nondeterministic progress can prompt arrangements of larger amount objectives even without requiring arrangements of every lower level.

CONCLUSSION

In this study, the presented scalable technique for online distributed learning to facilitate coordinated decision making in scalable smart cities. The presented approach proved its efficacy in coordinated decision-making for a distributed microservice in geographic information system-based health management system. The experimental results showed that the method can be effectively implemented for a minimum of 473 services. The technique can be extended to more complex domains of distributed sensing and medical IoT.

REFERENCES

1. National Academy of Science and Engineering. Recommendations for implementing the strategic initiative INDUSTRIE 4.0. In: Final report of the Industrie 4.0 Working Group, April; 2013. p. 261-71.
2. Perera C, Liu CH, Jayawardena S, Chen M. A survey on internet of things from industrial market perspective. IEEE Access 2014;2:1660-79.
3. Svensson B, Danielsson F. A multi-agent based microservice approach for flexible and robust manufacturing. Robot Comput Integr Manuf 2015;36:109-18.
4. Cheng SJ, Raja A, Lesser V. Multiagent meta-level microservice for radar coordination. Int J Web Intell Agent Syst 2013;11(1):81-105.
5. Abbasi-Yadkori Y, Bartlett P, Malek A. Linear programming for large-scale Markov decision problems. In: Proceedings of the 31st International Conference on Machine Learning; 2014. p. 124-32.
6. Ammar HB, Eaton E, Ruvolo P, Taylor ME. Online multi-task learning for policy gradient methods. In: Proceedings of the 31st International Conference on Machine Learning; 2014. p. 124-32.
7. Bratukhin A, Sauter T. Functional analysis of manufacturing execution system distribution. IEEE Trans Industr Inform 2013;7(4):740-9.
8. Wilson A, Fern A, Ray S, Tadepalli P. Multi-task reinforcement learning: A hierarchical Bayesian approach. In: Proceedings of the 24th International Conference on Machine Learning (ICML); 2007. p. 1015-22.
9. Li H, Liao X, Carin L. Multi-task reinforcement learning in partially observable stochastic environments. J Mach Learn Res 2009;10:1131-86.
10. Fernandez F, Veloso M. Learning domain structure through probabilistic policy reuse in reinforcement learning. Prog Artif Intell 2013;2(1):13-27.
11. Rai, Ankush, Sakkaravarthi Ramanathan, and R. Jagadeesh Kannan. "Quasi Opportunistic Supercomputing for Geospatial Socially Networked Mobile Devices." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016.
12. Rai, Ankush. "Attribute Based Level Adaptive Thresholding Algorithm (ABLATA) for Image Compression and Transmission." Journal of mathematics and computer science, 12 (2014), 211-218.
13. Rai A. Shell implementation of neural net over the UNIX environment for file management: A step towards automated operating system. J Oper Syst Dev Trends 2014;1(2):10-4.
14. Rai A. Automation of community from cloud computing. J Adv Shell Program 2014;1(1):21-3.