International Journal of Pure and Applied Mathematics
Volume 118 No. 24 2018
ISSN: 1314-3395 (on-line version)
url: http://www.acadpubl.eu/hub/
Special Issue

http://www.acadpubl.eu/hub/

QUASI OPPORTUNISTIC SUPERCOMPUTING USING POPULATION CODING OF NEURONS IN INTEGRATION WITH CO-SIMULATED NETWORKING

Ankush Rai[1], R. Jagadeesh Kannan[2]
School of Computing Science
Engineering,
VIT,
Chennai, India.
ankushressci@gmail.com
Jagadeeshkannan.r@vit.ac.in

May 23, 2018

Abstract

Research advances in supercomputing has lately been slowed down owing to multiple reasons like attentions given to design features and to modification from little to medium sized system that results in trivial enhancement in performance. There exists a huge demand of requesting applications to avail opportunistic computing for considering the right selection of prerogative supercomputers. Learning based communication models of opportunistic computing classes conventionally require hundreds or a large number of training samples. This study presents a new co0simulation based framework that gives the enhanced performance in contrast with traditional parallel computing frameworks. The objective of this research is to build a co-simulation

framework for on the fly computational commencing on socially convolved terrestrial grids to facilitate the sharing of computing workload over wireless networks as virtual supercomputers of unprecedented power. This will assist computing in heterogeneously interconnected mobile assets, Internet of Things, in crowdedscenarios for variable ranges like: stadiums, shopping centers, and so forth.

***Key Words*:**Quasi opportunistic supercomputing, cosimulation, modelling.

1 THE PROBLEM

We have implemented an opportunistic computing framework through socially connected device rather than establishing connections based on proximity of neighboring devices. The framework reduces the relevancy key for searching a network close to source. It is implementable for wider and larger distance away from the source. This imparts significant increase in the packet life over virtual grids. Here, message passing will be based on Evolving Boolean Networks which allow connections to self-regulate with the scarcity of mobile computing assets - for quasi-supercomputing applications. Thus, the ripples of message passing will facilitate the mapping of computational workload offered geographically by the source and that of local computing activities in a device at the same time; such that it avail real time to and fro computing without a permanent storage space. It will also be possible to share the computing workload of local devices over such virtual terrestrial grids - in case of crowd disaster control, management and sensor data mapping - make queering of socially relevant data (to fight crimes, terrorism, report mishaps, mapping of epidemic disease propagation). Thus, it ultimately increases the trust between asset owners (as assets facilitator) and data scientist (as broadcaster of computational workload).

2 QoS FRAMEWORK

Opportunistic computing systems have the potential to break up the standard of customary systems network and incorporate cor-

respondence with the dependency on human conduct (crowd behavior). In fact, few studies of opportunistic computing have been made to exploit social network features for driving the protocols as per the social conventions configuration. This seems to be a promising idea, as contacts between hubs are in a far-reaching way tied with human activities and consequently with interpersonal organization structures. Modeling over social availability is an exploration region that has pulled in expanding consideration (see fig 1). Modeling of versatility with the integration of our computationally evolving Boolean networks enables to accommodate the social availability of computing assets. We intended to build a Quasi-opportunistic computational framework to be utilized predominantly by applications made out of different specialists [6]. This framework is developed over that from [6] and offers a dynamic topology with diverse levels of social correspondence. The framework will be capable to model the environment context situations and it autonomously derives the inference that if the co-allocation of computing asset is proficient and viable, then its co-assignment must consider the various leveled structure of assets demand. We assure the importance by continuously evolving Boolean networks in which vertices share its computational components and edges share correspondence joints. Thus, it enables the establishment of an effective coordination between the asset solicitations and assets demand for addressing coordinating issue. When the focus is on distinguished assets, it ought to save the energy and made it accessible to run the parallel undertakings by issuing workload broadcast for computational application. We then will be able to execute a co-distribution component that uses propelled reservation of assets. This obliges that the co-designation frameworks of nearby bunches backing propelled asset reservation highlights. The issue of co-allotment of a huge number of assets in topographically disseminated regulatory spaces has been concentrated on broadly. We trust that for vast scale, the presented idea over opportunistic network situations, where no certifications of asset accessibility are given to the asset owner for all intents and purposes will ease the difficulty to take care of the co-allotment issue. The first step is to define the synchronous machine model of populated neuron based distributed systems in antiport setting with its weighted sensed data X_{ij} and control 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 control 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 symport rule writing through distributed reinforcement learning. Now, for the first step let us suppose that we have a total of Excitatory neurons (E_N) and Inhibitory neurons (I_N) where they are in the ratio of $I_N = 0.2E_N$. 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 a 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 0; thereby the binary vectors is given by $(E_N)\epsilon\{0,1\}$ and $y(I_N)\epsilon\{0,1\}$ for the excitatory and inhibitory neural activity at time step t, respectively. Hence, the antiport sequencization of P system 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$X_i j \& y_k i$, the synaptic weights is given as:

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

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

3 SETUP AND NOVELITY

The system is based upon a basic peripheral interface based upon a Programmable Interface controller. Here, the sensors act as peripherals which collect information and the energy harvesting units collect and store energy in the battery. The novelty here is that energy is harvested through 3 independent mechanisms and one of these is the magneto levitation unit which is relatively new and unused. The circuit is powered by the energy harvested via the 3 methods making it self-sustaining. The system design is given in Figure 1. θ is the Heaviside step function; T^E and T^I are the threshold values for excitatory and inhibitory neurons, where it is initially drawn from the uniform distribution within the interval [0, T^E_{max}] and [0, T^1_{max}]. $\xi E_i(t)$ and $\xi I_i(t)$ are white Gaussian noise processes with $\mu_\xi \epsilon$ [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 excitatory neurons in response for a sequence of activated neurons is made pass through the previously generated targeted sequence code blocks of firing neuron states $S_{ij}^{codeblock}$; which is determined by the adaptation rate η_{AD} 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 excitatory neurons which stabilize the amount of excitatory and inhibitory to drive sensory information through the excitatory control neurons. Therefore, the evolutionary dynamics of the neuronal membrane potential that mediates the excitatory 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 upon the current state and not onto its history. Thus, a deterministic strategy P assigns each state a unique action. While taking after a strategy we perform at time t action a_t at state s_t and observe a reward r_t (distributed according to $R_{MPD}(s,a)$). and the next state s_{t+1} (dispersed according to $P^{MDP}_{s_t,s_t+1}(a_t)$). 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 $\gamma\epsilon(0,1)$, and the discounted return of policy P is:

$$V^{P}_{MDP} = \sum_{t=0}^{\infty} \gamma^{t} r_t.$$

,

Where r_t is the reward observed at time t. Since all the rewards are bounded by R$_m$*axthediscountedreturnislimitedby* :

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

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.

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

Where, $I^i_{xy}(t)$ is the consequent frame (frame is a sequence of stimuli or also known as P system data sets) with the location in the form of (x, y) for the frame at time t,$I^i_{xy}(t)$ averaged over information of all $I^i_{xy}(t)$ value for time t [14, 15]. Hence, STD for the frame comprising of local decision space (δ^0_1)) and its multiscale decision space (δ^2_1) can be mapped as the quantitative information about its trajectory in continuous frame sequence can be derived from:

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

Thus, the dynamics of the equation for a computational job using is computed as shown below: At time t, standard deviation is re-

quested by every robotic devices. 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 given that $||Q_T - Q*||\varepsilon$, given that

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

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

The above bound is somewhat complicated. To simplify, assume that 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) be an indicator to the event that the solution is in state i during the p_th 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 sequence of information to excite the necessary regions and asses the information in an associative form. This enables several machines all at once to not only learn but 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 sep-

arated 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 machines 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, I_N are the universal set of level for the control actions, P_i P_2 are the adjoint sequence pairs with the levels L_W L_U respectively,S_{P_1,P_2} 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 sequence density layout of differing state at the control instance of the frame U. Also, t_1 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 [11-13]. Thus, we calculate the Tree of Action based on continuous feedback loop:

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

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: $C_{out} = [Q_2^{(1)} Q_2^{(2)} .... Q_2^{(N)}]$ where $Q_2^{(1)}, Q_2^{(2)}, ..., Q_2^{(N)}$ are the network outputs of each agent using reinforcement learning. The individual network outputs can be computed as:

$$Q_i^{(1)} = \sum_{r=1}^{N_2} w_{2r1} Q_1(r)$$

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

where W_{2r1} is the weight of the connection from the $2r_{th}$ input element to the 1_{th} 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 $W = W + \Delta W$, where ΔW is the change in weight estimated as: $\Delta w = y.Y_2.BP_{err}$, where γ is the learning rate. Generally the value of learning rate is between 0.2 to 0.5.

4 RESULTS

All the components of the presented Quasi-opportunistic computing system have been actively developed at VIT. While the framework is still immature and is unequipped for performing powerful calculations over 500 GFLOPS, we here show the execution of the resource based job allocation for subsystems, as it is plainly a prevailing factor in the possibility of the quasi-opportunistic processing concept.We have outlined a simulation environment to test our computing model. This environment permits us to depict the model, which incorporates numerous authoritative areas attempting to submit jobs to the framework. Resource allocation by the incorporated element is additionally simulated, and the Service Level Agreements are made by predefined social function. We characterize our framework's usage record or Utilization Index as the rate of submitted demands of resources in which the client really issued the demand for assets in the following computing job assignment round. The results demonstrate that there is dependability in number of connections between the jobs that the regulatory space gets and its utilization index. In this manner, each managerial space is persuaded to expand its workload. Likewise, we have found that managerial areas that share their assets are engaged much of the time and liberally collect higher workload over the long haul. Notwithstanding, an authoritative space that has numerous successive clients tends to have a lower utilization index. This outcome

adjusts to the automated Boolean search space for resource allocation; since it adjusts with the regulatory network as a steady number of registering workload to share among countless resources which tend to have fewer jobs per client. In Fig 2, we test the impact of the sharing recurrence parameter or frequency parameter on the utilization index. Sharing frequency is characterized as the likelihood that the managerial area will share its assets in every round. Our trials are completed on a settled number of regulatory spaces (N=10), each of which begins with the same number of assets and the same measure of computational workload. Every line in the diagram depicts consistent workload sharing frequency with an alternate bidding frequency by the computing assets. The bidding frequency is the likelihood of the authoritative area to offer in every round of designation of processing jobs.

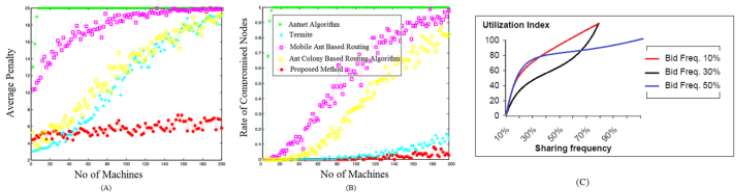

Figure 2: (A) Performance analysis of distributed learning where average penalty is taken as metric to determine the encounter of failure, (B) Scalability Analysis for the presented system. (c) Plot between utilization index and sharing frequency of the subsystems for the automated sharing of computational workload.

These outcomes are surely positive, as they demonstrate the legitimacy of our framework: sharing assets is an ideal procedure for each regulatory area. In this way, we can expect that the reasonable conduct of each authoritative area will fit in with the framework planners' aims. Testing the impact of expressed held costs on the normal long haul jobs. Here, we found that any given space could amplify its computational processing by discovering saved values of the ideal conditions during the processing of workload. These saved values can be expressed in form of cost for achievement of job allocation by every single socially convolved networked area; all available throughout the framework's history.

This is achieved with the help of Dynamic Social Networking Algorithm (CO-SIMULATION). We likewise built up an estimated calculation for the count of an optimal held cost. The goal of CO-SIMULATION is to learn a dimensionality reduction of their inputs after having obtained a reduced dimensional vector representation of saved values. This step has only been used in a setting where the tree structure was given a-priori. The settings before continuing with POPULATION CODING model can learn the tree structure. Fig. 3 shows an instance of a CO-SIMULATION applied to a given tree. Assume that we have been given a list of word vectors x = (x1, . . . ,xm). Also we have given a binary tree structure for this input in the form of branching triplets of parents with children: $(p \rightarrow c1c2)$. Each child can be either an input word vector x_i or a non terminal node in the tree. Thus we have the following triplets: $((y1 \rightarrow x3x4), (y2 \rightarrow x2y1), (y1 \rightarrow x1y2))$. In order to apply the same neural network to each pair of children, the hidden representations yi have to have the same dimensionality as the xi s. Given this tree structure, it can now compute the parent representations. The first parent vector y1 is computed from the children (c1, c2) = (x3, x4): p = f(W_e[c1; c2] + b_e),where we have multiplied a matrix of parameters $W_e \epsilon$ R n2n by the concatenation of the two children. After adding a bias term; the element-wise the tanh (activation) function is applied to the resulting vector. One way of assessing its suitability for this n-dimensional vector which represents its children is to try the reconstruction of the children in a reconstruction layer. During training, the goal is to minimize the reconstruction error of this input pair. For each pair sets, it shall compute the Euclidean distance between the original input and its reconstruction from CO-SIMULATION. Again, after computing the intermediate parent vector y2, we can assess how well this vector captures the content of the children by computing the reconstruction error. The process meant to repeats itself until the full tree is constructed to have a reconstruction error allocated at each nonterminal node. This model requires a fixed tree structure which is fulfilled by the CO-SIMULATION. So far the project goals that we have achieved are: (i) 300 400 GFLOPS at the crowd/device ratio of 600:450 for the conventional crowd control simulations with multiple reproducibility of the results at uncontrolled environment. (ii) Map the type and nature of computing workload with terres-

trial specific computing at local devices. (iii) Able to autonomously determine the best suited socially convolved terrestrial places for establishment of virtual networks to facilitate quasi opportunistic supercomputing. (iv) Provideaubiquitous computing infrastructure to support IoT. Another imperative conclusion suggests that the introductory asset dissemination has no impact in the long run. Despite the fact that the space based introductory workload influences the allotment in the initial few adjusts, its future usage of the framework depend just on natural conduct of crowd (static or dynamic). This conclusion is in light of the suspicion that all the calculation for astute processing has profitable assets, i.e., there are areas willing to give assets of any of the regulatory or nonmanagerial subsystems of the network lattice. This suspicion is authentic in the connection of semi pioneering networks. The impact of such a model on the nature of automation in asset distribution guarantees to ascend as a condition of state of art in the ongoing research.

References

[1] T. C. Hsia and Z. Y. Guo. Joint trajectory generation for redundant robots. International Conference on Robotics and Automation, Scottsdale, 1989.

[2] C. Gill, R. Cytron and D. Schmidt. Middleware scheduling optimization techniques for distributed real-time and embedded systems. Seventh IEEE International Workshop on bject-Oriented Real-Time Dependable Systems, San Diego, 2002.

[3] H. A. Duran-Limon and G. S. Blair. Reconfiguration of resources in middleware. Seventh IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, San Diego, 2002.

[4] S. S. Yau and Xiaoyong Zhou. Schedulability in model-based software development for distributed real-time systems. Seventh IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, San Diego,2002.

[5] S. Krishnamurthy, W. H. Sanders and M. Cukier. Performance evaluation of a probabilistic replica selection algorithm. Seventh IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, San Diego, 2002.

[6] Ankush Rai, Sakkaravarthi Ramanathan, R. Jagadeesh Kannan. Quasi Opportunistic Supercomputing for Geospatial Socially Networked Mobile Devices. IEEE 25th International Conference on Enabling Technologies, 2016.