ISSN: 1231-4005
e-ISSN: 23540133
DOI: 10.5604/01.3001.0010.3056

MATHEMATICAL SYSTEM MODEL FOR ACOUSTICS BASED TELEMATIC MICRO SERVICES IN IOT FOR TRANSPORTATION SETTING

Jagadeesh Kannan R., Ankush Rai

VIT University
School of Computing Science & Engineering, India
e-mail: jagadeeshkannan.r@vit.ac.in; ankushressci@gmail.com

Janusz Szpytko, Yashesh C. Pandya

AGH University of Science and Technology, Poland
Adama Mickiewicza Av. 30, 30-059 Krakow
e-mail: szpytko@agh.edu.pl, ypandya@student.agh.edu.pl

Abstract

Despite the wide adoption of Internet of things (IoT) with several webs standards and cloud technologies, building of city wide IoT based smart city platform for solving transportation problem remains a daunting task. Owing to the dynamic nature of IoT and components of transportation systems, smart city architecture would require development of a scalable, distributed and evolving architecture on the web. With the advancement in autonomous transportation system there is a need for in adaptive telematic system for communicating with other vehicles, sensor nodes etc. As transport, services have special requirements of which are related to the size and type of information to be exchanged between vehicles (vehicle-to-vehicle communication) and the control centre. . By the time the data makes its way to the cloud for analysis, the opportunity to act on it might be gone. Thus handling such huge streams of data on the fly is a daunting task. In the study we present an interoperable swarm, logic based mobile terminals running multimedia micro services based telematic system.

Keywords: *Telematic micro services, Acoustics based Internet of Things, Telematic, Smart City*

1. Introduction

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

a. Distributed sensing from the environment [3],
b. Performing local computation of the sensed data,
c. 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 integration of manufacturing system on a global scale with several advances to achieve simultaneous communication, computation and telematic micro service in manufacturing domain [1]. From a system level point of view DMP is crucial to achieve a scalable holonic system; wherein small sub-systems 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 telematic micro service should avail overall coordination between industrial services. This requires enhancing machine level decision making process, adaptive sharing of resources, developing matrix of specificity of actions as per the varying context [4, 6]. 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 telematic micro service actions [9, 10]. In addition, the rising trend of on-demand dispatch of telematic micro service is a predicament owing to its complexity in modelling. A good solution is to integrate the problem of modelling scalable system and its distributed telematic micro service features within the same framework.

2. Methodology

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-modelling approach will enable high learning rate for high dimensional on demand telematic micro service. The system is tested and validated in virtual manufacturing setting.

The first step is to define the synchronous machine model of distributed systems with its weighted sensed data x_{ij} and telematic micro service 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 telematic micro service 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.

2.1. Learning model for acoustic middleware telematic service

We can model the correlation between different corresponding users (CoUs) and collaborative users (CUs) in a wireless communication environment as the receiving baseband signal $y_n(t)$ as the nth CU during the spectrum-sensing interval denoted by t_{ss} can be composed as:

$$y_n(t) = \begin{cases} x_i(t), h_{nC} \\ G_i(t)C(t) + x_i(t)\ h_C \end{cases} t \in [0, t_{ss}].$$

Where, $x_i(t)$ represents the additive white Gaussian noise, $G_i(t)$ is the channel gain which models the multipath fading channel, $C(t)$ represents the CoUs signal. Also, h_{nC} & h_C are the hypothesis of Non-Corresponding User signal and the transmitted signal form CoU. For sensing the communication spectrum, a correlation coefficient is required to model the two sensing signals as a ratio of its covariance in the time intervalt_{ss} and product of its standard deviation. However, this signals are subject to multipath fading between two cross sensing channels$G_a(t)$ & $G_b(t)$. Thus, the cross correlation coefficient is given as:

$$\delta_{ab}(t) \approx \frac{cov\,(G_a(t), G_b(t))}{\sigma_{y_a(t)} \cdot \sigma_{y_b(t)}}.$$

This will enable us to derive the channel gain at each CUs end from the deterministic contribution of scatter signal loss due to diffusion $G_{i,DIFF}(t)$ and channel gain Line of Sight path (LOS) as:

$$G_i(t)^* = G_{i,DIFF}(t) + \delta_{ab}(t)\,.\,G_{i,LOS}(t).$$

Thus, the correlation coefficient between CoUs and CUs can be expressed as:

$$c_{a,b}(t) = \begin{cases} 0\ h_{nC} \\ \frac{G_i(t)^* + \delta_{ab} y_n(t)}{\sigma_{y_a(t)} \cdot \sigma_{y_b(t)}} \end{cases} h_C.$$

It will be easier to use a machine learning algorithm in situation like this to determine the connectivity strength and based on that accommodate a dynamic wireless network topology; such that the mission commands shall be dispatched to the CUs if the CoU can't directly issue the mission command to it owing to the problems of poor connection strength, multipath fading of signals etc. Therefore, we are using the reinforcement learning technique to reconfigure the action state pairs of the ontological decision-making in the proposed middleware service.

Reinforcement Learning (RL) is a region of the machine learning which is concerned with the association of operators with its environment. At each correspondence the operators recognizes the current states of nature, and picks an action to execute. This action causes changes in environment, in its turn, sends a scalar fortification sign as punishment or reward; demonstrating the adequacy of its actions. Thusly, "The RL issue is proposed to be a direct resultant of joint effort of computational agents to achieve a goal". The RL issue can be settled by component programming and the perfect course of action made sense of whether the probability of reward, state and actions are known. Regardless, this is not frequently the circumstance, and quantifiable testing schedules were delivered. One such approach is Q learning. In Q-learning software agent, make sense of acceptable behaviour in an ideal world in a Markovian territory by experiencing the results of their actions [16, 17]. Operators can utilize Q learning out how to secure a perfect technique using delayed reward. Hence, the computational agent can figure the perfect methodology despite when there is no earlier data of the effects of its action on the environment. Q learning utilizes the rewards and the best estimation of the present state to upgrade the evaluation of the past state-action pair. Presently, we portray a Markov Decision process (MDP) as takes after:

Definition 1: A Markov Decision Process MDP is a 4-tuple (S, A,T_p*, R), where S is signifies set of the states, A characterize set of actions; where, A(i) is the set of actions available at state of transition.* $T_{p(i,j)}^{MDP}(a)$*which is the probability of transition from a particular state i to state j during performing action a* $\in$*U(i) in state i, and* $R_{MDP}(s,a)$ *is the reward received when performing action a in set of state marked as s.*

We take $R_{MDP}(s,a)$ as non-negative and confined toR_{Max}, i.e.,$\forall s,a$: 0$\leq R_{MDP}(s,a) \leq R_{Max}$. For the simplicity we adopt the idea that the reward $R_{MDP}(s,a)$ is considered to be deterministic, granting the fact that all of our results relate when $R_{MDP}(s,a)$ is stochastic. We assign a policy for an MDP at each time *t*, for each states as a probability for performing action *a* $\in U(s)$, as per the given history of action state pairs during the ontological communication of our middleware service as:

$$H_{t-1} = \{s_1, a_1, r_1, \dots, s_{t-1}, a_{t-1}, r_{t-1}\}.$$

This includes the states and actions with its concerned rewards observed until time $t-1$. A policy P principally depends only upon the current state and not onto its history. Thus, a deterministic approach P assigns for each state a unique action. While taking after a strategy *P* we execute 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 $P_{S_t,S_{t+1}}^{MDP}(a_t)$).

Hence, now we can establish the sequences of rewards to a single value as the return, and the goal is to maximize it. Hence, the computational process is to focus on this discounted return, which has a parameter $\gamma \in (0,1)$, 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 byR_{Max}.

For a grouping of sets for state and activity, let the covering time, meant by C', be a furthest point of confinement on the quantity of state-activity sets starting from any pair, until all state-activity shows up in the consecutive course of action. The Q-learning algorithm gauges the state-action value function (for discounted return) as takes after:

$$Q_{C+1}(s,a) = Q_C(s,a) + \alpha_t(s,a)\left(R_{MDP}(s,a) + \gamma \max_{b \in U(s')} Q_C(s',b)\right).$$

Where is the state reached from states when performing action *a* at time *t*. Since, Q-learning is a non-concurrent process as it updates a single entry every step. Note that the covering time can be a component of both the MDP and the consecutive plan or just of the grouping itself. At first we acknowledge that from any start of a state, inside of C' steps all state-activity pair grouping show up in the plan. From that point, we rest the presumption and acknowledge that with likelihood in any event, from any begin state in C' steps all state-activity shows up in the gathering. This hidden approach creates the succession of state activity sets for or middleware administration. This algorithmic procedure can be summed in as takes after:

Algorithm: Q-learning based Ontological Middleware Service Algorithm

Step 1: for N←N'; Evaluate (N' is the number of CoUs & N is the number of CUs):

$$\delta_{ab}(t) \approx \frac{cov\,(G_a(t),G_b(t))}{\sigma_{y_a(t)}.\sigma_{y_b(t)}}.$$

After each C cycles ← Update the state and actions using:

$$\delta_{ab}(t) \approx \frac{cov\,(G_a(t),G_b(t))}{\sigma_{y_a(t)}.\sigma_{y_b(t)}}.$$

Step 2: Evaluate correlation coefficient between each pairs of CoUs and CUs:

$$c_{a,b}(t) = \begin{cases} 0 & h_{nC} \\ \frac{G_i(t)^* + \delta_{ab} y_n(t)}{\sigma_{y_a(t)}.\sigma_{y_b(t)}} & h_C \end{cases}.$$

Step 3: Repeat steps 1-2 until all state sets are mapped.

Step 4: Compute Q learning based state-action value function:

$$Q_{C+1}(s,a) = Q_C(s,a) + \alpha_t(s,a)\left(R_{MDP}(s,a) + \gamma \max_{b \in U(s')} Q_C(s',b)\right).$$

Return V_{MDP}^{P}.

Step 5: Check:

$$\textbf{if } V_{MDP}^{P} < V_{Max}.$$

Based on weighed ordering of $Q_{t+1}(s,a)$:

$$H_{t+1} = \{s_1, a_1, r_1, \dots, s_{t+1}, a_{t+1}, r_{t+1}\}.$$

Else

return:

$$H_t = \{s_1, a_1, r_1, \dots, s_{t-1}, a_{t-1}, r_{t-1}\}.$$

Step 6: Repeat steps 4, 5 & 6 Until C←max(C).

Step 7: End Process.

Following the above step, the generated data need be forwarded semantically to 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 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. 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 services 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 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 telematic micro service 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 super positioned pair density layout of differing state at the service's instance of the frame U. Also, 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 delayst_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: $C_{out} = [Q_2^{(1)}Q_2^{(2)} \ldots . Q_2^{(N)}]$ 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_{2r1} 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 $w = w + \Delta w$, where Δw is the change in weight estimated as:$\Delta w = \gamma . Y_2 . BP_{err}$, where γ is the learning rate. Generally, the value of learning rate is between 0.2 and 0.5.

2.2. Modelling M2M communication of acoustic devices

The e is the number of M2M devices *e* is divided into three classes: device initial readiness be i_e, operating M2M devices denoted as u_e and loitered number of devices be v_e. Similarly, the average packet arrival density per device S is divided into two classes: initial i_s and operational denoted as u_s. By considering the criss-cross interaction availability of devices and packet arrival density, the equations that describe the spread of the signals can be written as:

$$\frac{di_e}{dt} = \mu_e + A - k_{+1} i_e - c(S) i_e u_s + \delta v_e, \frac{du_e}{dt} = c(S) i_e u_s - \gamma u_e - k_{+1} u_e,$$

$$\frac{dv_e}{dt} = k_{-1}(u_e) - k_{+1} v_e - \delta v_e, \frac{de}{dt} = \mu_e + A - k_{+1} e \ ,$$

$$\frac{di_s}{dt} = \mu_s - k_{-1} i_s - \beta_2 i_s u_e - \beta_3 i_s v_e, \frac{du_s}{dt} = -k_{-1} u_s + \beta_2 i_s u_e + \beta_3 i_s v_e \ \&,$$

$$\frac{dS}{dt} = \mu_s - k_{-1} e \ .$$

where $e = i_e + u_e$ & $S = i_s + u_s$.

In the system, μ_e is priority index, A is the maximum delay threshold and k_{+1} is forward reaction rate constant. c is the total concentration of the enzyme-substrate complex. γ is the recovery rate and δ is the parameter denotes the flow rate such that the v_e will join the u_e class, μ_s is probability that the preceding delay threshold is violated and k_{-1} is its reverse flow rate of signals. β_2 and β_3 are the interaction rates of operational number of devices with the initial and recovered classes of the M2M devices respectively ($\beta 2 > \beta 3$). The model gives following two cases to be analysed:

(a) The waiting time for a queued packet c of the operating M2M devices with the infective M2M devices is a constant, and

(b) It depends upon the initial values of operational M2M units. For positive constants $a0$ and $a1$, thus c takes the form $c = a_0 + a_1 S$.

Case (b) is impractical at high numerical values such as ours. Therefore, we shall exempt rest of the calculation for this case.

Case a. When $c = c_0;\ c_0$ is a constant

Since $i_e + u_e + v_e = e$ & $i_s + u_s = S$, the system (1.1) can be reduced to the form:

$$\frac{du_e}{dt} = c_0(e - u_e - v_e)u_s - (\gamma + k_{+1})u_e, \frac{dv_e}{dt} = \gamma(u_e) - (k_{+1} + \delta)v_e, \frac{de}{dt} = \mu_e + A - k_{+1} e, \frac{du_s}{dt} =$$
$$-k_{-1} u_s + \beta_2 (S - u_s) u_e + \beta_3 (S - u_s) v_e, \frac{dS}{dt} = \mu_s - k_{-1} e \ .$$

The region of attraction of the above system is

$$T_1 = \{(u_e, v_e, e, u_s, S) : 0 \le u_e + v_e \le N_1 \le \bar{e}, 0 \le u_s \le S \le \bar{S}\}.$$

To give proper versatility at the most minimal expenses, the stage has been architected to keep running in an open/private acoustic data processor environment or server farms of M2M environment. The framework tends to offer undertaking administrations crosswise over open, private and in addition crossover cloud situations. This shall help meet necessities of information security, administrative prerequisites with reference to information stockpiling, offering undertakings control over their stockpiling and additionally network through secure passages or confined acoustic ranges. The administrations are provisioned and oversaw utilizing format driven setting, permitting administrations to be taken off rapidly and effectively. This model offers

a genuinely shared methodology and it permits a 'pay-as-you-develop' business model for customers in a transportation services. It also answers to be made from pilot to industrialized worldwide take off on an anticipated and controllable expense model, utilizing a typical acoustic-based communication environment. In particular, the stage offers uniform base that is accessible all around, along these lines empowering device free access to the adaptive telematic services.

3. Results & 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 stimulations 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 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, fulfilment, and end conditions, have been set up under exceptionally broad conditions (Fig. 1).

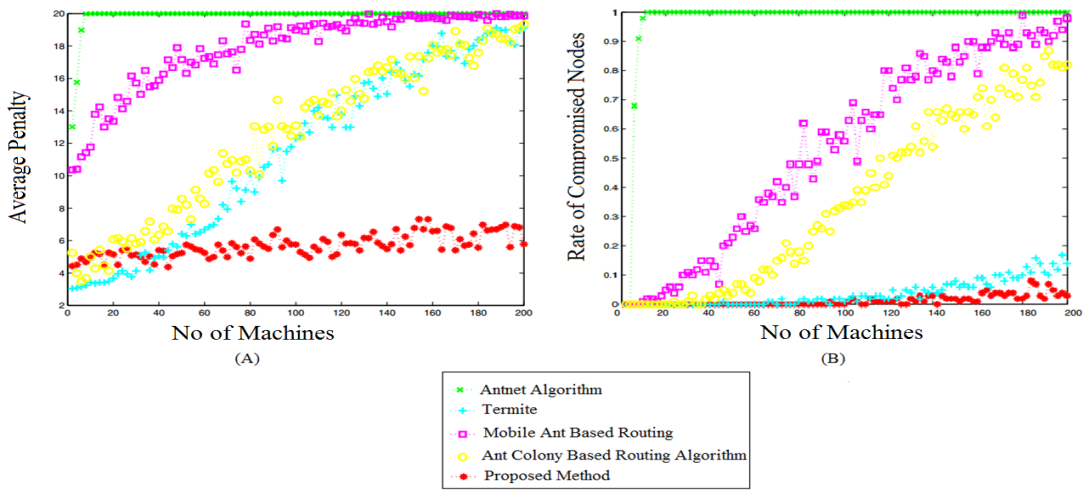

Fig. 1. Performance results and comparison. (A) Performance analysis of distributed learning where average penalty is taken as metric to determine the encounter of failure, (B) Scalability Analysis for the presented system.

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 history decisions sets is to propagate the firing chin to other neurons in order 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 neighbourhood 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.

4. Conclusion

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 telematic and its micro service system. The experimental results showed that the method could be effectively implemented for a minimum of 473 concurrent 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., Final report of the Industrie 4.0 Working Group, pp. 261-271, 2013.
[2] Perera, C., Liu, C. H., Jayawardena, S., Chen, M., *A survey on internet of things from industrial market perspective*, IEEE Access, Vol. 2, pp. 1660-1679, 2014.
[3] Svensson, B., Danielsson, F., *A multi-agent based telematic micro service approach for flexible and robust manufacturing*, Robotics and Computer-Integrated Manufacturing, Vol. 36, pp. 109-118, 2015.
[4] Cheng, S. J., Raja, A., Lesser, V., *Multiagent meta-level telematic micro service for radar coordination,* An International Journal of Web Intelligence and Agent Systems, Vol. 11, No. 1, pp. 81-105, 2013.
[5] Abbasi-Yadkori, Y., Bartlett, P., Malek, A., *Linear programming for large-scale Markov decision problems*, Proceedings of the 31st International Conference on Machine Learning, pp. 124-132, 2014.
[6] Ammar, H. B., Eaton, E., Ruvolo, P., Taylor, M. E., *Online multi-task learning for policy gradient methods*, Proceedings of the 31 st International Conference on Machine Learning, pp. 124-132, 2014.
[7] Bratukhin, A., Sauter, T., *Functional analysis of manufacturing execution system distribution*, IEEE Trans. Ind. Informat., Vol. 7, No. 4, pp. 740-749, 2013.
[8] Wilson, A., Fern, A., Ray, S., Tadepalli, P., *Multi-task reinforcement learning: a hierarchical Bayesian approach*, Proceedings of the 24th International Conference on Machine Learning (ICML), pp. 1015-1022, 2007.
[9] Li, H., Liao, X., Carin, L., *Multi-task reinforcement learning in partially observable stochastic environments*, Journal of Machine Learning Research, Vol. 10, pp. 1131-1186, 2009.
[10] Fernandez, F., Veloso, M., *Learning domain structure through probabilistic policy reuse in reinforcement learning*, Progress in AI, Vol. 2, No. 1, pp. 13-27, 2013.