Volunteer Hive Computing and Virtualization in Botswana's Edification techniques

Suresh Shanmugasundaram &
Divyapreya Chidambaram
Botho University
Botho Education Park
PO. Box 501564, Gaborone
Botswana
suresh.shanmugasundaram@bothouniversity.ac.bw
,

ABSTRACT

Finest Trail Setup Technique in Hive computing & virtualization Ambience[1] proposes a fuzzy logic based SVM approach to secure a collision free path avoiding multiple dynamic obstacles. The navigator consists of an FSVM -Based Collision Avoidance. The decisions are taken at each step for the providers & consumers to negotiate without collision. Fuzzy-SVM rule bases are built, which requires simple evaluation data based on the active participants in the Hive computing & virtualization at a given point of time. The effectiveness of the proposed method is verified by a series of simulations and implemented with a BOINC Manager. The time optimization algorithm that plans and responds a user request on auction-based resource allocation systems can be used for further improvement of Hive computing & virtualization broker services. Issues pertaining to scalability have to be addressed in this scenario. This dispute in the Hive computing & virtualization System can be addressed with Incremental FSVM with Kernel which will give better result in group nodes/clusters. Further to augment the technique, hive computing & virtualization service orchestration is applied. This enhances the approach by creating an application-aligned infrastructure that can be deployed by defining the policies. This paper displays a middleware Hive computing & virtualization Process Orchestration (CPO) for directing and observing computer-generated mechanisms into a Hive computing & virtualization computing environment.

KEYWORDS

Negotiations in Hive computing & virtualization, Optimal path planning, Hive computing & virtualization Service orchestration

1 INTRODUCTION

A robot is autonomous when it is able to move purposefully with no intervention from a human user in un-engineered real-world environment. The development of techniques for autonomous navigation in a real-world environment constitutes one of the major trends in the current research on robotics. By incorporating multi-agent optimal path planning [2], [3], [4] and Support Vector Machine learning, special efficiency for robotic navigation can be achieved. One of the important aspects that are still deemed important to consider in mobile robots is obstacle avoidance. A pertinent problem in autonomous navigation is the need to cope with a large amount of uncertainty that is inherent of natural environments, and to respond reactively to unforeseen events as soon as they are perceived. Fuzzy logic with Support Vector Machine is an adequate method to obtain certain and finite data in an optimal manner. However, this approach proves that it is enough for areas of highest danger coefficients to cause the robot to change direction, which consequently reduces the number of fuzzy rules that control the robot motion [3], [4]. An SVM learns the decision surface from two distinct classes of the input points [5], [6]. In many applications, each input point may not be fully assigned to one of these two classes. In this paper a fuzzy membership to each input point is applied and the Support Vector Machines are reformulated so that different input points can make different contributions to the learning of decision surface.

2. CPO ARCHITECTURE

Fig. 1 shows the Hive computing & virtualization Process Architecture with the features of the negotiation which is supported by the componenets like the CPO Run, CPO Maestro Factory Service and workflow files. To negotiate, the provider and consumer nodes must establish connection, which is achieved by following the optimal planning path. With CPO Orchestration, the issue of scalability is addressed.

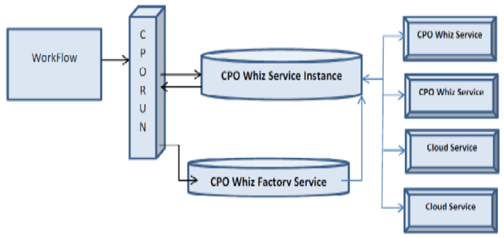


Fig 1. CPO Architecture

3. FUZZY LOGIC CONTROLLER

3.1. Decision making system

For decision-making, either type 1 fuzzy or type 2 fuzzy can be used as per the requirement [7], [8]. The
four principal components of type 1 fuzzy decision-making systems are: a) *The Interface*: This determines the input and output variables and maps them into linguistic variables that are to be displayed for the Hive computing & virtualization participants; b) *The Knowledge Base*: This is a part of expert systems that contains the domain knowledge. Membership functions, Group Key agreement and control rules are decided by the experts at this point, based on their knowledge of the system; c) *The decision-making logic*: This treats a fuzzy set as a fuzzy proposition. One fuzzy proposition can imply another, and two or more fuzzy propositions can be associated by a Boolean connectivity relation to infer a final fuzzy proposition; d) The defuzzification interface converts the fuzzy output into a crisp value. A type 1 fuzzy set has a grade of membership that is crisp, where as a type 2 fuzzy set has grade of membership that are fuzzy, so
it is called 'fuzzy-fuzzy set'. As the type 1 gives only sub optimal solution, type 2 fuzzy is good in
dealing with uncertainty, which is usual in Hive computing & virtualization ambience.

4. HIVE COMPUTING & VIRTUALIZATION ORCHESTRATION & SUPPORT VECTOR MACHINES

SVM is a Machine Learning Technique developed on statistical learning theory. For machine learning tasks involving pattern classification, multi sensors information fusion, non-linear system control etc, SVMs have become an increasingly popular tool. To calculate the margin, two *parallel* hyper planes are constructed, one on each side of the separating hyper plane, which are "pushed up against" the two data sets. Intuitively, a good separation is achieved by the hyper plane that has the largest distance to the neighboring data points of both classes, since in general the larger the margin the better the *generalization error* of the classifier [6], [9]. This could be implemented to identify the nodes in Hive computing & virtualization through optimal path. A Hive computing & virtualization workflow is classified into two portions: descriptions and progression. In the description part, variables are acknowledged, services that will be performed are recognized, and the cluster of directives which will be used many epochs in the progression is generated. The progression portion outlines the order in which the services are appealed, how to switch their arrival programs, what to do when error has ensued, how to complete the occurrences, and displays the exit program that need be resumed to the CPO Run in the conclusion of its accomplishment.

4.1 Multi class support vector machine

For the conventional SVM an n class problem is converted into n two-class problem and for the ith two class problem, class i is separated from remaining class with decision function that classifies class i and remaining classes be

$$D_i(\mathbf{x}) = w_i^t \mathbf{x} + b_i ,$$

SVM, if for the input vector **x** Di (**x**) > 0 is classified for i, **x** is classified in to class i. Let the decision function for class i against class, with the maximum margin [5], [6] be

$$D_{ij}(\mathbf{x}) = w_{ij}^t \mathbf{x} + b_{ij} , \qquad ($$

where $D_{ij}(\mathbf{x}) = D_{ji}(\mathbf{x})$ for the input vector **x** can be calculated

$$D_i(\mathbf{x}) = \sum_{i=1}^{n} sign\left(D_{ij}(\mathbf{x})\right) \qquad (3$$

and classify **x** into the class

$$\arg\max_{i=1}^{n} D_i(\mathbf{x}) . \qquad (4$$

4.2 Fuzzy support vector machine (FSVM)

An SVM learns the decision surface from two distinct classes of the input points. In many applications, each input point may not be fully assigned to one of these two classes. In this paper, a fuzzy membership is applied to each node in the path and the SVMs are reformulated so that different nodes in the Hive computing & virtualization can make different contribution to the learning of decision surface. Class i is defined as a one dimensional membership functions mii(**x**) on the direction orthogonal to the optimal separating hyper planes Dj(**x**) = 0 as given below

$$1.\ \text{For } i = j \quad m_{ii}(\mathbf{x}) = 1 \quad \begin{matrix} \text{for } D_i(\mathbf{x}) > 1, \\ D_i(\mathbf{x}) \text{ otherwise,} \end{matrix} \qquad (5)$$

$$2.\ \text{For } i \neq j \quad m_{ij}(\mathbf{x}) = 1 \quad \begin{matrix} \text{for } D_j(\mathbf{x}) < -1, \\ -D_i(\mathbf{x}) \text{ otherwise.} \end{matrix} \qquad (6)$$

The class i membership function of **x** is defined using the minimum operator for

$$m_i(\mathbf{x}) = \min_{j=1}^{n} m_{ij}(\mathbf{x}) . \qquad (7)$$

Now the datum **x** is classified in to the class $\arg\max_{i=1}^{n} m_i(\mathbf{x})$ if x satisfied

$$D_k(\mathbf{x}) \begin{cases} > 0, \text{ for } k = i, \\ \leq 0, \text{ for } k \neq i,\ k = 1, \cdots, n \end{cases} \qquad (8)$$

and $m_k(\mathbf{x})$ is given by

$$1.\ k \in i_1, \quad m_k(x) = \min_{\substack{j=i_1 \\ j \neq k}}^{i_1} \left(-D_j(x)\right), \qquad (9)$$

$$2.\ k \neq j, \left(j = i_1, \cdots, i_1\right), \quad m_k(x) = \min_{j=i_1}^{i_1} \left(-D_j(x)\right), \qquad (10)$$

Thus the maximum degree of membership is achieved among $m_k(\mathbf{x})$, $k = i_1, \cdots, i_1$ (*Fig. 2*) [4], [5].

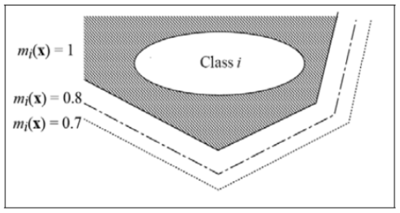


Fig. 2. Contour lines of the class i membership function

5. IMPLEMENTATION DETAILS

Architecture: The Hive computing & virtualization architecture consists of three layers. The Resource Layer is a collection of nodes communicating with the Hive computing & virtualization's hardware resources so as CPU Cycles and memory. One module can be shared by two or more Brokers, which reduces redundancy in coding. The development tools used is Java language.

5.1 Multi-agent system & Hive computing & virtualization Orchestration

Fig. *3* depicts the simplified diagram representing the Abstract-Owner system using an AO Broker/Agent.

The agent basically interacts with the other components of the Hive computing & virtualization system by manipulating information on the agent. The information on the agent may represent facts, assumptions, and deductions made by the system during the course of

solving the problem by the consumers. An agent is a partial problem solver, which may employ a different negotiation strategy and contribute to the interaction of consumers and providers by viewing the information on the AO agent. The CPO manages workflow applications, tasks, and mainly Hive computing & virtualization services. Its workflows are termed and they can be produced straight by the users. To exemplify the use of the CPO the scenario shown in Figure 3 is considered. In this scenario of three laboratories, A, B and C user controls and organizes the orchestration mechanisms in order to create an experimentation measured by hive computing & virtualization services available in computers that are connected to. The trials in A and B can be performed concurrently, but C depends on the A and B outcomes to accomplish its calculation.

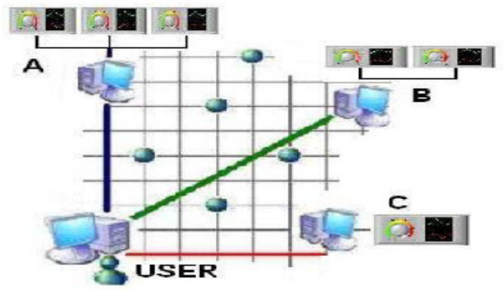


Fig. 3. Application Scenario, Labs A, B & C

5.2 Intelligent AO Agents/Brokers Fuzzy SVM based Collision Avoidance

The agent called the Fuzzy Collision Detector [10], [2] is a fuzzy SVM –based Collision Avoidance Controller, which is used in sensing Robotic movements. In this paper, a fuzzy membership is applied to each node in the Hive computing & virtualization-computing ambience and the SVMs are reformulated so that different input points can make different contributions to the learning of decision surface and hence pave the way for efficient Hive computing & virtualization interaction. The fuzzy-based agent is fed with CPU Cycles or memory as an input, acquired from a set of Hive computing & virtualization nodes. The values that are stored in the AO Broker in terms of CPU cycles and Memory availability (near, medium, and far) are considered for the optimal Hive computing & virtualization interactions thus avoiding the collision. The CPU cycles of the processors is shown in Fig. *4*. For the operation of a computational Hive computing & virtualization, the broker discovers properties of resources that the user can access through the Hive computing & virtualization information server(s), negotiates with (Hive computing & virtualization-enabled) resources or their agents using middleware services, maps tasks to Computational Resources (scheduling), stages the application and data for processing (deployment) and finally gathers results [11]. It is also responsible for monitoring application execution progress along with managing changes in the Hive computing & virtualization infrastructure and resource failures. *Fig. 5* shows the high-level view of Hive computing & virtualization where the computational resources use both brokers and information services.

Computational resource navigation

A method for uniquely detecting the resource availability on a heavy traffic high performance network; separately sensing packets scattered from the congested segment region indicative of a traffic delay and producing signals representative thereof; normalizing the signal from node with respect to the consumer to indicate availability [8].

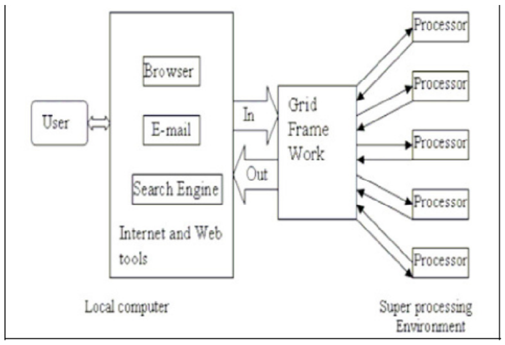


Fig. 4. Task split-up for grid interaction

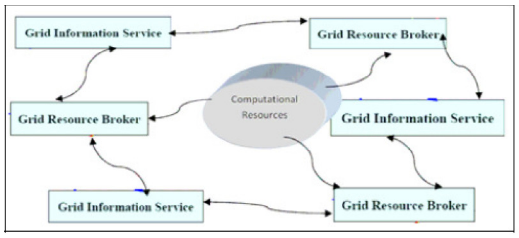


Fig. 5. High-level view of grid

Network monitor
The Network Monitoring Component monitors the remote clients for their availability. Availability of these clients is decided based on the amount of their CPU cycles [9] and their Memory capacity being used for any on-going executions. CPU Usage is the data that represents the amount of CPU cycles being spent at the remote client for local processes. This data is to be taken into account because busy processes should not be overloaded with more tasks, which may lead to system failure or hang up. It is calculated based on the Operating System versions. On Windows NT, CPU usage counter is '% Total processor time' whose index is 240 under 'System' object whose index is 2. And, in Win2K/XP, Microsoft moved that counter to '% processor time' whose index is 6 under 'Total' instance of 'Processor' object whose index is 238. The index value is specified as a parameter to the method that finds the CPU Usage information using performance counters. The above proposed algorithm can be implemented using Java and the performance is evaluated for this envisioned system. PERFHIVE COMPUTING & VIRTUALIZATION tool is used to analyze the CPU and memory usage of four nodes at the normal state. This is compared against values obtained for the CPU and memory usage by implementing the optimized IC Scheduling algorithm.

5.3 Multi-agent optimal path planning

Problems of multi-agent Hive computing & virtualization systems have got significance [7]. Each AO agent Hive computing & virtualization system has some transport subsystem, which consists of several nodes. The method based on graph optimization algorithms [12] has been applied to control the dynamic group of nodes. Novelty of the developed multi-agent path planning algorithm [13], [14] is as follows:

- Hive computing & virtualization participants are considered as dynamic obstacles;
- Graph representation of common distribution environment mode is used for path planning;
- The quickest path is planned (time optimization);
- Expert rules for speed and path correction are synthesized to provide Hive computing & virtualizationlock avoidance.

These algorithms provide global optimality. Optimal Path Planning Approach to Hive computing & virtualization Environment also provides Hive computing & virtualizationlock avoidance. *Ant colony optimization* also can be used to identify a group of nodes in Hive computing & virtualization.

6. EXPERIMENTATION AND RESULTS.

The algorithm schedules the eligible tasks so as to maximize the number of eligible tasks at each step of computation, thus making maximum utilization of the available resources in the Hive computing &

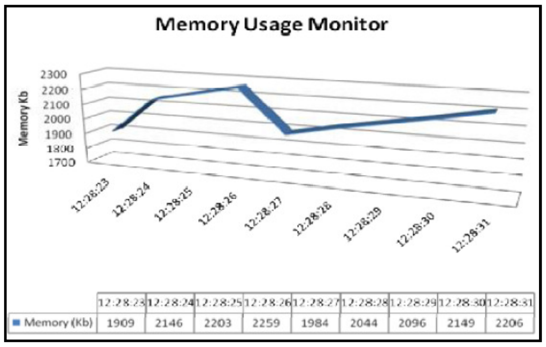
virtualization Framework. The example job, which is used for evaluating the performance and thereby comparing the results is shown below;
Example Job: (((a+b)*(c+d))-((e+f)*(g+e))).
The above job was split into tasks using Job Splitter and the tasks are allocated to the four nodes based on their availability. This allocation was done using the Optimized IC Scheduling Algorithm. The CPU usage and Memory availability before and after implementing the algorithm was observed using PerfHive computing & virtualization Tool and the Network Monitor designed using Java. The Processor utilization in terms of percentage is tabulated in *Table I* as follows:

Table I

Network monitor

Resource Usage		PerfGrid	Network Monitor
CPU Usage %	Client 1	1.52	1.4
	Client 2	0.82	27.12
	Client 3	0.09	12.33
	Client 4	0.87	34.52

Fig. 6 represents the line graph, which depicts the percentage of CPU usage. For PerfHive computing & virtualization [12] the CPU usage is equal to zero for all the four clients, which means the CPU is not used as there are no processes allocated to these clients. However, there will be a significant value for memory availability because of the RAM usage even in the idle state of the machine. The time optimization algorithm [15] that plans and responds a user request on auction-based resource allocation systems can be used for further improvement of Hive computing & virtualization broker services. Hive computing & virtualization System can also be enhanced with Incremental FSVM [9] with Kernel will give better result in group nodes/clusters. Since the Hive computing & virtualization ambience requires an optimized approach of scheduling resources, the proposed technique can further improved by a more robust hybrid algorithm [16], which actually provides a way for scheduling projects/tasks in resource constrained environment like the Hive computing & virtualization. Hence the combination of the approaches [15], [16] is believed to have better scheduling solutions which could be made as future enhancements.



	12:28:23	12:28:24	12:28:25	12:28:26	12:28:27	12:28:28	12:28:29	12:28:30	12:28:31
■ Memory (Kb)	1909	2146	2203	2259	1984	2044	2096	2149	2206

Fig 6. Memory Monitor - Orchestration

7. CONCLUSIONS

The multi-agent and multi-agent optimal path planning approach provides an extra level intelligence. This paper deals with the real-time navigation made by the Hive computing & virtualization nodes in a totally uncertain environment called Hive computing & virtualization Ambience. Type 2 fuzzy gives a better approach to solve uncertainty. Based on fuzzy logic and Support Vector Machine, a collision free technique has been proposed that partitions a task and hence fed to the processors. This proposal has been validated in different unknown environments cluttered with static and dynamic obstacles and has proven to give the node, a means of to be an active Hive computing & virtualization participant. These techniques provide better scheduling solutions and hence make resource utilization more efficient. With the popularity of *hive computing & virtualization computing* it is now necessary to apply Service Orchestration in the context of this paradigm to address the scalability issue. Workflows and processes are used in different domains. An orchestrator is the entity that manages complex cross domain (system, enterprise, firewall) processes and handle exceptions which can be observed by the memory usage monitor [17]. Since an orchestrator is valuable in the fulfillment, assurance as well as billing processes, in their most advanced service aware incarnations should be capable of adjustments based on feedback from monitoring tools. The main difference between a workflow automation and orchestration is that work flows are processed and completed as processes within a single domain

8. REFERENCES

[1] Suresh S.,Divyapreya C., 'Finest Trial Setup Technique in Hive computing & virtualization Ambience', Proceedings of the International Conference on Informatics & Applications, 3-5 June 2012, University Sultan Zainal Abidin, Kula Terangganu, Malaysia pp 449-455.

[2] Merchant-Cruz E. A., Morris A. S. Fuzzy-GA-based trajectory planner for robot manipulators sharing a common workspace, *IEEE Tr. Robotics*, Vol. 22, No. 4, 2006, pp. 613–624.

[3] Poornaselvan K. J., Gireesh Kmar T., Vijayan V. P. Agent based ground flight control using type-2 fuzzy logic and hybrid ant colony optimization to a dynamic environment, in *Proceeding of First International Conference on Emerging Trends in Engineering and Technology*, 16-18 July 2008, Nagpur,Maharashtra, India, pp. 38–42.

[4] Kumar T.G., Vijayan V. P. A multi-agent optimal path planning approach to robotics environment, in *Proceeding of International Conference on Computational Intelligence and Multimedia Applications*, 13-15 December 2007, Sivakasi, Tamil Nadu, India, pp. 400–404.

[5] Abe S., Inoue T. Fuzzy support vector machines for pattern classification, in *Proceeding Of International Joint Conference on Neural Networks* (IJCNN'01'), 15-19 July 2001,Washington DC, USA, Vol. 2, pp. 1449–1454.

[6] Abe S., Inoue T. Fuzzy support vector machines for multiclass problems, in *Proceedings on European Symposium on Artificial Neural Networks*, 24-26 April 2002, Bruges, Belgium, pp. 113–118.

[7] Huq R., Mann G. K. I., Gosine R. G. Behavior-modulation technique in mobile robotics using fuzzy discrete event system, *IEEE Tr. Robotics*, Vol. 22, No. 5, 2006, pp. 903–916.

[8] Levine S. P., Bell D. A., Jaros L. A. Simpson R. C. Koren Y., Borenstein J. The NavChair assistive wheelchair navigation system, *IEEE Tr. Rehabilitation Engineering*, Vol. 7, No. 4, 1999 pp. 443–451.

[9] Evgeniou T., Pontil M., Poggio T. Regularization networks and support vector machines, in *Advances in Large Margin Classifiers*, Cambridge, MA, MIT Press, 2000, pp. 171–204.

[10] Fayad C., Webb P. Optimized fuzzy logic based on algorithm for a mobile robot collision avoidance in an unknown environment, *7th European Congress on Intelligent Techniques and Soft Computing*, Aachen, Germany, 13-16 September 1999, pp. 26–33.

[11] Foster A., Kesselman C. (Ed.) The grid: blueprint for a future computing infrastructure, Morgan Kaufmann Publishers, USA, 1999.

[12] Abbas A. *Grid computed to technology and applying: A practical guide to technology & applications*, Firewall Media, New Delhi, India, 2009.

[13] Assistive Technology & Artificial Intelligence, Lecture Notes in Computer Science, KISS Institute for Practical Robotcs, Vol. 1458, 1998, Springer-Verlag, Berlin, 1998, pp. 126–136.

[14] Xiaolei Yin Y.,Go, Y., Bowling A. Navigability of multi-legged robots, *IEEE/ASME Tr.Mechatronics*, Vol. 11, No. 1, 2006, pp 1-8.

[15] Sulistio A., Buyya R., A time optimization algorithm for scheduling bag-of-task applications in auction-based proportional share systems, in *Proceedings of the 17th International Symposium on Computer Architecture on High Performance Computing* (SBAC-PAD '05), Rio de Janeiro, Brazil, 24-27 October 2005, pp. 235–242.

[16] Szendr_i E. A robust hybrid method for the multimode resource-constrained project scheduling problem, *Pollack Periodica*, Vol. 5, No. 3, 2010, pp. 175–184.

[17] Vignola, S., Zappatore, S., "The LABNET-Server Architecture For Remote Management of Distributed Laboratories: Currrent Status and Perspectives", in Springer-Verlag, Distributed Cooperative Laboratories, Issues in Networking, Instrumentation and Measurement, Davoli, F., Palazzo, S., Zappatore, S., Eds., USA, 2006, pp. 435-450