International Journal of Research
e-ISSN: 2348-6848 & p-ISSN 2348-795X Vol-5, Special Issue-10
International Conference on Innovations in Information and Communication Technology
Held on 9th and 10th March 2018 organized by The Department of Information Technology,Meenakshi Sundararajan Engineering College Kodambakkam,Chennai, India

Virtual Machine Migration using Particle Swarm Optimization Algorithm in Mobile Cloud Computing

Prem Kumar. M[1] , Vishnu Kumar. M[2] & V. M. Sivagami[3]
[1,2] IV Year B.Tech IT, Sri Venkateswara College of Engineering
[3]Associate Professor, Department of Information Technology, Sri Venkateswara College of Engineering

Abstract
Mobile Cloud Computing (MCC) can play a vital role by offloading Big Data related tasks such as sharing, processing and analysis, from mobile applications to cloud resources, ensuring Quality of Service (QoS) demands of end-users. Such resource migration, is also termed Virtual Machine (VM) migration. The emergence of Mobile Cloud Computing (MCC) facilitates reduction of task-execution time and real-time communication latency.

Introduction

The emergence of mobile cloud computing (MCC) facilitates reduction of task-execution time and real-time communication latency. The [1], proposes how cloud computing is integrated into the mobile environment (MCC). MCC effectively utilizes distributed cloud server resources such as CPU, memory, network, and ports to execute the mobile Smart healthcare applications. Using MCC, mobile devices (MD) can offload these applications to a resourceful cloud server for faster execution. However, the long distance between cloud servers and the MDs may increase the response time for interactive applications, increasing the total execution time.

The [5] proposes that a dynamic server migration and consolidation algorithm is introduced. The [6] proposes identification of various computing paradigms promising to deliver the vision of computing utilities. The [7] proposes a CloneCloud i.e. a system that automatically transforms mobile applications to benefit from the cloud. The [8] proposes a cloud - based solution that is based on the client running on the smartphone and the server side running in the cloud.

User mobility is not the only reason forcing a VM to migrate. Migration can be initiated to minimize the overprovisioned resources and thus improve the overall system objectives. For instance, if a VM is required to be migrated from a cloudlet to any of the candidate cloudlets, the new cloudlet may not have the same type of VM. In that case, a VM with more resource than the current one must be chosen and provisioned in order to migrate the VM and thus minimize task-execution time. However, in this approach, the VM migration is provisioned more resources than the required. Therefore, this over-provisioned resources greatly decreases the system objectives, as it reduces the number of provisioned VMs in the cloudlets

Related Work

The [1], proposes how cloud computing is integrated into the mobile environment (MCC).They showed the performance improvement with respect to battery life, storage, and bandwidth .Also, they have considered , environmental factors like heterogeneity, scalability, and availability issues. They also focussed on security issues related to reliability and privacy.

The Advantage of this paper, is that it improved the performance metrics of cloud. But it doesn't address security of mobile users and data in cloud

The [5] proposes that a dynamic server migration and consolidation algorithm is introduced. The algorithm is shown to provide substantial improvement over static server consolidation in reducing the amount of required capacity and the rate of service level agreement violations.

The advantages proposed in this paper is that it consolidates the server capacity for better resource usage but the limitation is that it is difficult to virtualize large amounts of data servers

The [6] proposes identification of various computing paradigms promising to deliver the vision of computing utilities; defines Cloud computing and provides the architecture for creating market-oriented Clouds by leveraging technologies such as VMs.

The advantages proposed in [6] is better load balancing strategies to improve performance. But some restrictions on available services may be faced, as it depends upon the cloud service provider.

The [7] proposes a CloneCloud i.e. a system that automatically transforms mobile applications to benefit from the cloud. The system is a flexible application partitioner and execution runtime that enables unmodified mobile applications running in an application-level virtual machine to seamlessly off-load part of their execution from mobile devices onto device clones operating in a computational cloud.

The advantage proposed by [7] is Off-loading a part of an application to a CloneCloud, reduces the task

execution time. But more elasticity means more less control in case of public clouds.

The [8] proposes a cloud - based solution that is based on the client running on the smartphone and the server side running in the cloud. Sensors embedded in the smartphone are used for the measurement of the different information about the monitored person. Basic algorithms evaluating current person's health status run on the smartphone. Measured data are sent to the second part of the application running in the cloud for deeper analysis.

The advantage proposed in [8] is cost - effectiveness from the consumer side. But it vastly demands for cloud security.

The [9] states that ant colony optimisation is applied to the problems where a number of alternative *pheromone representations* are available, each of which interacts with this underlying bias in different ways. The [9] explores both the structural aspects of the problem that introduce this underlying bias and the ways two pheromone representations may either lead towards poorer or better solutions over time.

Mobile computing continuously evolves through the sustained effort of many researchers. It seamlessly augments users' cognitive abilities via compute-intensive capabilities such as speech recognition, natural language processing, etc. By thus empowering mobile users, we could transform many areas of human activity. The [10] discusses the technical obstacles to these transformations and proposes a new architecture for overcoming them. In this architecture, a mobile user exploits virtual machine (VM) technology to rapidly instantiate customized service software on a nearby cloudlet and then uses that service over a wireless LAN; the mobile device typically functions as a thin client with respect to the service. A cloudlet is a trusted, resource-rich computer or cluster of computers that's well-connected to the Internet and available for use by nearby mobile devices. Our strategy of leveraging transiently customized proximate infrastructure as a mobile device moves with its user through the physical world is called cloudlet-based, resource-rich, mobile computing.

Existing system

In the existing system using ant colony algorithm its Mobile users may move from one Access Point (AP) to another, increasing their distances between current locations and the cloudlet, where the tasks are provisioned. So, it increases the task-execution time.

Pheromones and Initial Pheromone values are calculated In the ACO algorithm, pheromones represent the desirability of choosing a solution. The pheromones represent the desirability of assigning a VM to a cloudlet. Each ant starts with an initial pheromone value for each VM to cloudlet pair. The initial solution is generated using a greedy First Fit(FF) VM migration approach.

A Heuristic Value is calculated to build a solution each ant uses the local heuristic value to select a cloudlet for a VM. This heuristic value defines the favourability of choosing a cloudlet for a VM to construct the solution. As PRIMIO tries to jointly optimize the objectives of total task-execution time and resource over-provisioning in cloudlets. So, the local heuristic value has to be defined in such a way that the system can optimize the task execution time and resource overprovisioning to reflect the system objectives. Local heuristic is defined as, $\eta u,v = \alpha \times \eta E\ u,v + (1-\alpha)\eta R\ u,v$ (19) where, α is the system parameter defines the relative weight between the objectives of total task execution time and resourceover-provisioning of cloudlets. It can be tuned according to the system environment. ηE i, j and ηR i, j is the objectives of total task execution time and the resource over-provisioning respectively, if VM migrates from cloudlet i to cloudlet j.

Local Pheromone Update: When an ant chooses a VM pair to construct a solution, it immediately updates the local pheromone value in relation to the initial pheromone value. Local pheromones are updated by each ant using the following relation, $\tau u,v(t + 1) = (1-\gamma l)\times \tau u,v(t) + \gamma l \tau 0$ (24) where, γl is the system parameters, denoting the relative importance of current pheromone value at time t, $\tau u,v(t)$. Global Pheromone Update: Global pheromone values are updated for each pair of VM and cloudlet only when all ants constructed their local optimal solutions and updated the global optimal solution. The global pheromone values are updated using the following relation, $\tau u,v(t + 1) = (1-\gamma g)\times \tau u,v(t) + \gamma g \Delta \tau u,v$ (25) where the γg is the global pheromone system parameter, this parameters can be tuned according to the system objectives. $\Delta \tau u,v$ is the global pheromone value for the updated global solution, which is defined as $\Delta \tau u,v = \{\tau u,v$, if $(u,v) \in$ PGS 0, otherwise (26) where, PGS is the global solution set of the selected VM u for migration and the target VM v where the migration will be occurred.

Proposed system

We propose a VM migration technique for a heterogeneous MCC system following the user's mobility pattern. That is, when a user moves from one cloudlet to another cloudlet, the resource or VM must be migrated to the cloudlet that is nearest to the user.

We use particle swarm Optimization (PSO) to identify the optimal target cloudlet. We develop a particle swarm Optimization (PSO) based VM migration model, in which VM are migrated to

International Journal of Research
e-ISSN: 2348-6848 & p-ISSN 2348-795X Vol-5, Special Issue-10
International Conference on Innovations in Information and Communication Technology
Held on 9th and 10th March 2018 organized by The Department of Information Technology,Meenakshi Sundararajan Engineering College Kodambakkam,Chennai, India

candidate cloud servers so as to maximize the total utility of the MCC system.

The Cornfield Vector: Heppner's bird simulations had a feature which introduced a dynamic force into the simulation. His birds flocked around a "roost," a position on the pixel screen that attracted them until they finally landed there. This eliminated the need for a variable like craziness, as the simulation took on a lie of its own. While the idea of a roost was intriguing, it led to another question which seemed even more stimulating. Heppner's birds knew where their roost was, but in real life birds land on any tree or telephone wire that meets their immediate needs. Even more importantly, bird flocks land where there is food. How do they find food? Anyone who has ever put out a bird feeder knows that within hours a great number of birds will likely find it, even though they had no previous knowledge of its location, appearance, etc. It seems possible that something about the flock dynamic enables members of the flock to capitalize on one another's knowledge, as in Wilson's quote above. The second variation of the simulation defined a "comfield vector," a two-dimensional vector of XY coordinates on the pixel plane. Each agent was programmed to evaluate its present position in terms of the equation: Eval= Jw + 4- so that at the (100,100) position the value was zero. Each agent "remembered" the best value and the XY position which had resulted in that value. The value was called pbest[] and the positions pbestx[] and pbestyl] (brackets indicate that these are arrays, with number of elements = number of agents). As each agent moved through the pixel space evaluating positions, its X and Y velocities were adjusted in a simple manner. If it was to the right of its pbestx, then its X velocity (call it vx) was adjusted negatively by a random amount weighted by a parameter of the system: vx[]=vx[] - rand()*p-increment. If it was to the left of pbestx, rand()*p-increment was added to vx[]. Similarly, Y velocities vy[] were adjusted up and down, depending on whether the agent was above or below pbesty. Secondly, each agent "knew" the globally best position that one member of the flock had found, and its value. This was accomplished by simply assigning the array index of the agent with the best value to a variable called gbest, so that pbestx[gbest] was the group's best X position, and pbesty[gbest] its best Y position, and this information was available to all flock members. Again, each member's vx[] and vy[] were adjusted as follows, where g-increment is a system parameter. ifpresentx[l> pbestx[gbest] then vx[] = vx[] - rand() *g-increment if presentx[] < pbestx[gbest] then vx[] = vx[] + rand() *g-increment ifpresenty[] > pbesty[gbestl then vy[] = vy[] - rand() *g-increment if presenty[l< pbesty[gbestl then vy[] = vy[] + rand() *g-increment In the simulation, a circle marked the (100,100) position on the pixel field, and agents were represented as colored points. Thus an observer could watch the flocking agents circle around until they found the simulated cornfield. The results were surprising. With p-increment and g-increment set relatively high, the flock seemed to be sucked violently into the cornfield. In a very few iterations the entire flock, usually 15 to 30 individuals, was seen to be clustered within the tiny circle surrounding the goal. With p-increment and g-increment set low, the flock swirled around the goal, realistically approaching it, swinging out rhythmically with subgroups synchronized, and finally "landing" on the target.

Module description

1) Identify VMs for migration:

As user might move, the distance with respect to cloud differs. This increases latency and cause performance issues. Hence, we need to identify the best VMs that suits the user transaction. We will use Particle Swarm Optimization to identify best VMs.

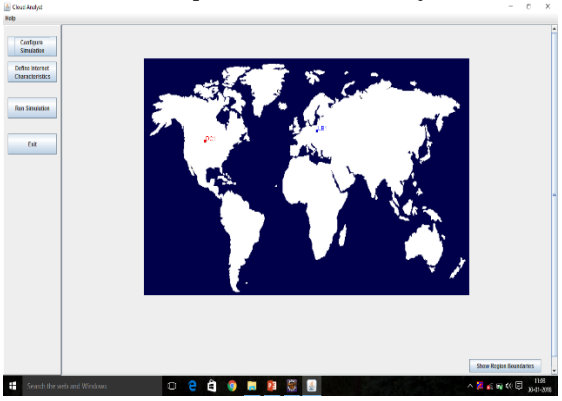

Fig. 1. General Map Image

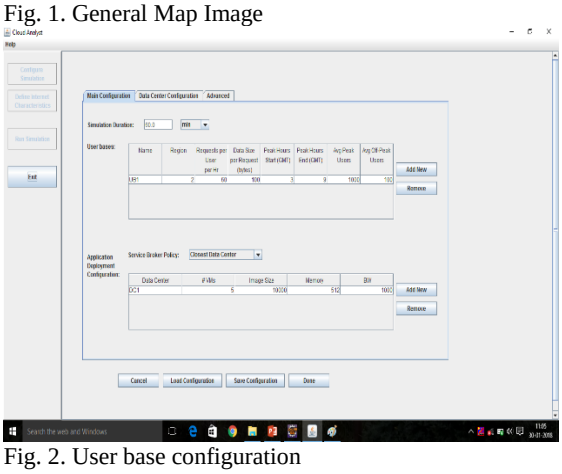

Fig. 2. User base configuration

International Journal of Research
e-ISSN: 2348-6848 & p-ISSN 2348-795X Vol-5, Special Issue-10
International Conference on Innovations in Information and Communication Technology
Held on 9th and 10th March 2018 organized by The Department of Information Technology,Meenakshi Sundararajan Engineering College Kodambakkam,Chennai, India

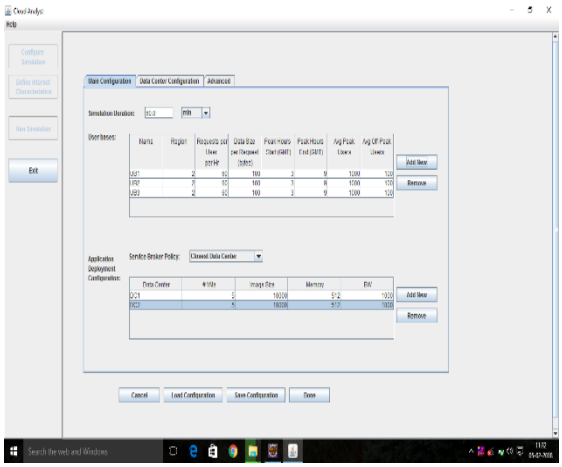

Fig. 3. Data center configuration

2) Calculate fitness value

Fitness value of VM will be computed with PSO algorithm. A fitness value is an objective function that summarizes a single figure of merit to achieve optimal design. The fitness value is used to compute the local best position and the global best position. Here a node, in this case, a virtual machine is a part of the belief space we consider. The virtual machine's state depends upon the fitness value.

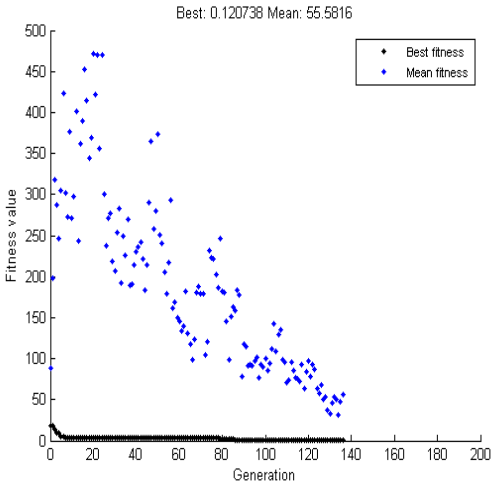

Fig. 4. Fitness graph

3) Cloudlet migration for higher performance

Once the best VMs are identified, we will validate the cloudlet and migrate transaction to the respective cloudlet using Live Virtual Migration technique. This reduces downtime for migrating overloaded VMs.

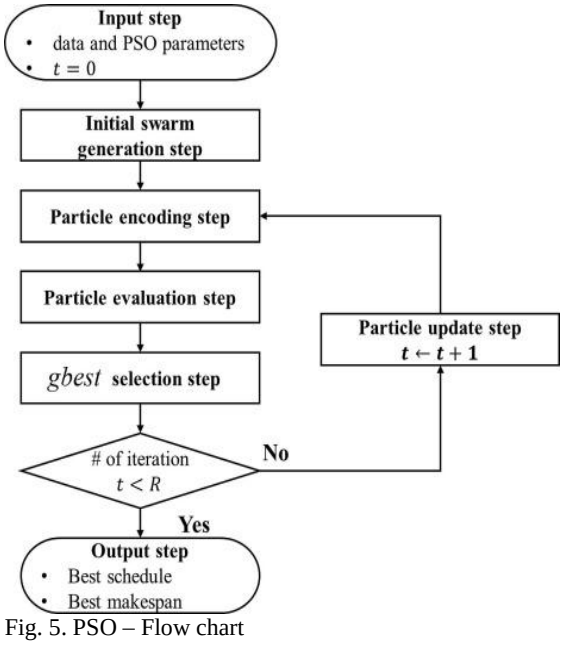

Fig. 5. PSO – Flow chart

4) Performance evaluation

Performance of user transaction will be validated using PSO algorithm. The performances results are generated in the form of graphs. We use the generated graphs to evaluate the performance of the transaction before the Virtual Machine Migration and the performance after the Virtual Machine Migration.

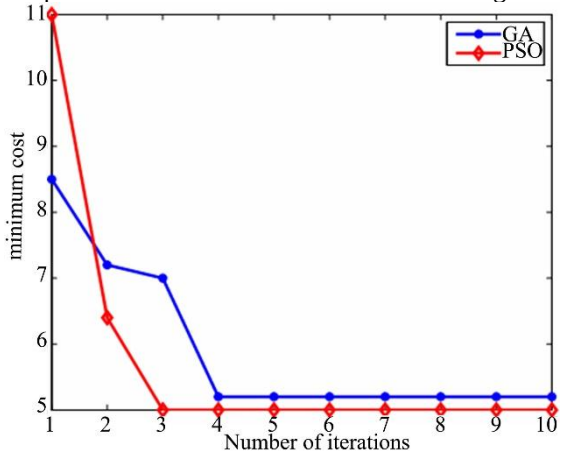

Fig. 6. Greedy algorithm Vs. PSO

VM Migration using Particle Swarm Optimization - Architecture

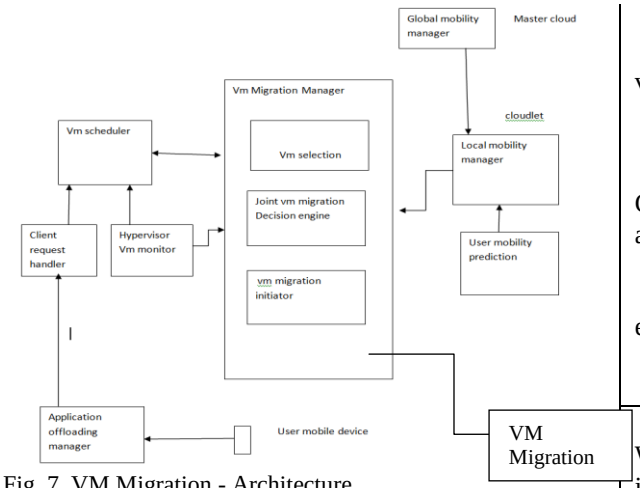

Fig. 7. VM Migration - Architecture

Description of the system architecture

In the proposed architecture, a computation migration system is depicted, which has two important functionalities. First, the computation migration between resource-constrained mobile devices and cloudlets; Second, virtual machine migration between two cloudlets to minimize the task execution time as well as resource over-provisioning. The application offloading manager helps a mobile device to decide which part of the mobile application should be offloaded. The client-request handler in a cloudlet manages incoming task execution requests from mobile devices and dispatches to the virtual machine scheduler. The VM scheduler uses the information from hypervisor to initiate the VM scheduling. Moreover, after a specific time epoch, the VM migration manager interacts with the VM scheduler to select tasks and to migrate so as to reduce the task-execution time. In this work, our main concern is to develop an algorithm for the VM migration manager for selecting a set of tasks and cloudlets, remapping tasks to minimize execution time. In the VM migration manager, there are three main components- VM Selection, Joint VM Migration Engine, and VM Migration Initiator, which contribute to making optimal migration decisions. Moreover, the user mobility manager takes local mobility management information in conjunction with global mobility prediction manager to decide on a set of probable cloudlets for a user. Finally, the VM migration manager initiates migration process for a set of tasks requiring faster execution.

PSO Algorithm

For each particle
Initialize particle;
END

Do
For each particle
Calculate fitness value;
If the fitness value is better than the best fitness value (pBest) in history
set current value as the new pBest;
End

Choose the particle with the best fitness value of all the particles as the gBest;
For each particle
(a) Calculate particle velocity according equation
(b)Update particle position according equation
End

While maximum iterations or minimum error criteria is not attained

Conclusion

We propose a VM migration technique for a heterogeneous MCC system following the user's mobility pattern. That is, when a user moves from one cloudlet to another cloudlet, the resource or VM must be migrated to the cloudlet that is nearest to the user.

We use Particle Swarm Optimization (PSO) to identify the optimal target cloudlet. We develop an Particle Swarm Optimization (PSO) basedVM migration model, in which VM are migrated to candidate cloud servers so as to maximize the total utility of the MCC system.

References

[1] Dinh, Hoang T., et al. "A survey of mobile cloud computing: architecture, applications, and approaches." *Wireless communications and mobile computing* 13.18 (2013): 1587-1611.

[2] J. H. Abawajy and M. M. Hassan, "Federated internet of things and cloud computing pervasive patient health monitoring system," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 48–53, 2017.

[3] E. Ahmed, M. Imran, M. Guizani, A. Rayes, J. Lloret, G. Han, and W. Guibene, "Enabling mobile and wireless technologies for smart cities," *IEEE Communications Magazine*, vol.55, no.1, pp.74–75, 2017.

[4] M. Basiri, A. Z. Azim, and M. Farrokhi, "Smart city solution for sustainable urban development," *European Journal of Sustainable Development*, vol. 6, no. 1, pp. 71–84, 2017

[5] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations," in *Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium* on, May 2007, pp. 119–128.

[6] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality

for delivering it services as computing utilities," in *High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on. IEEE*, 2008, pp. 5–13.
[7] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proceedings of the Sixth Conference on Computer Systems, ser. EuroSys '11*. New York, NY, USA: ACM, 2011, pp. 301–314.
[8] R. Cimler, J. Matyska, and V. Sobeslav, "Cloud based solution for mobile healthcare application," *in Proceedings of the 18th International Database Engineering & Applications Symposium, ser. IDEAS '14*. New York, NY, USA: ACM, 2014, pp. 298–301
[9] J. Montgomery, M. Randall, and T. Hendtlass, "Structural advantages for ant colony optimisation inherent in permutation scheduling problems," in *Proceedings of the 18th International Conference on Innovations in Applied Artificial Intelligence, ser. IEA/AIE'2005*. London, UK, UK: Springer-Verlag, 2005, pp. 218–228.
[10] P. G. J. Leelipushpam and J. Sharmila, "Live vm migration techniques in cloud environment : A survey," in *Information Communication Technologies (ICT), 2013 IEEE Conference* on, April 2013, pp. 408– 413.
[11] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, Oct 2009.
[12] M. Guntsch, M. Middendorf, and H. Schmeck, "An ant colony optimization approach to dynamic tsp," in *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, ser. GECCO'01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.*, 2001, pp. 860–867.
[13] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *INFOCOM, 2012 Proceedings IEEE. IEEE*, 2012, pp. 945–953.
[14] Z. L. Phyo and T. Thein, "Correlation based vms placement resource provision," *International Journal of Computer Science & Information Technology*, vol. 5, no. 1, p. 95, 2013.

Abstract:

The abstract is to be in fully-justified italicized text, at the top of the left-hand column as it is here, below the author information. Use the word "Abstract" in 10 point Times, boldface type at starting to the column, initially capitalized. The abstract is to be in 10- point, single-spaced type, and may be up to 3 in. (7.62 cm) long. .

Keywords

Keywords should be the keywords used in the article or related to the articles, Each keywords should be separated by comma or semi-colon, E.g. International Journal of Research, Edupedia Publications, ISOAR Journals, Book Publisher

1. Introduction

Headings should be numbered and in Times New Roman with 12 in size, bold and left alignment. These guidelines include complete descriptions of the fonts, spacing, and related information for producing your proceedings manuscripts. Please follow them and if you have any questions, direct them to the production editor in charge of your proceedings at the IEEE Computer Society Press: Phone (714) 821-8380 or Fax (714) 761-1784.

2. Formatting Your Paper

Page should be of A4 size with normal margin. All printed material, including text, illustrations, and charts, must be kept within a print area. Do not write or print anything outside the print area. All text must be in a two-column format. Text must be fully justified. A format sheet with the margins and placement guides is available in Word files as <format.doc>. It contains lines and boxes showing the margins and print areas. If you hold it and your printed page up to the light, you can easily check your margins to see if your print area fits within the space allowed.

3. Main Title

The main title (on the first page) should begin 1-3/8 inches (3.49 cm) from the top edge of the page, centered, and in Times 14-point, boldface type. Capitalize the first letter of nouns, pronouns, verbs, adjectives, and adverbs; do not capitalize articles, coordinate conjunctions, or prepositions (unless the title begins with such a word). Leave two blank lines after the title.

4. Author Name(s) and Affiliation(s)

Author names and affiliations are to be centered beneath the title and printed in Times 12-point, non-boldface type. Multiple authors may be shown in a two- or three-column format, with their affiliations below their respective names. Affiliations are centered below each author name, italicized, not bold. Include e-mail addresses if possible. Follow the author information by two blank lines before main text.

5. Second and Following Pages

The second and following pages should begin 1.0 inch (2.54 cm) from the top edge. On all pages, the bottom margin should be 1-1/8 inches (2.86 cm) from the bottom edge of the page for 8.5 x 11-inch paper; for A4 paper, approximately 1-5/8 inches (4.13 cm) from the bottom edge of the page.

6. Type-style and Fonts

Wherever Times is specified, Times Roman, or New Times Roman may be used. If neither is available on your word processor, please use the font closest in appearance to Times that you have access to. Please avoid using bitmapped fonts if possible. True-Type 1 fonts are preferred.

7. Main Text

Type your main text in 10-point Times, single-spaced. Do not use double-spacing. All paragraphs should be indented 1 pica (approximately 1/6- or 0.17-inch or 0.422 cm). Be sure your text is fully justified—that is, flush left and flush right. Please do not place any additional blank lines between paragraphs. You can use courier for source code.

```
For I = 1 to 10 do
  Print “this figure”
End;
```

Figure 1. Example figure.

7.1. Figure and Table Captions

Captions should be 9-point Times New Roman font, boldface. Callouts should be Times New Roman, non-boldface. Initially capitalize only the first word of each figure caption and table title. Figures and tables must be numbered separately. For example: “Figure 1. Example figure.”, “Table 1. Table example.”. Figure captions are to be **below** the figures (see Figure 1). Table titles are to be centered **above** the tables (see Table 1.

Table 1. Table example.

One	Two	Three

8. First-order Headings

For example, “1. Introduction”, should be Times 12- point boldface, initially capitalized, flush left, with one blank line before, and one blank line after. Use a period (“.”) after the heading number, not a colon.

8.1. Second-order Headings

As in this heading, they should be Times 11-point boldface, initially capitalized, flush left, with one blank line before, and one after.

8.1.1. Third-order Headings. Third-order headings, as in this paragraph, are discouraged. However, if you must use them, use 10-point Times, boldface, initially capitalized, flush left, preceded by one blank line, followed by a period and your text on the same line.

9. Acknowledgements

This work was supported in part by a grant from the National Science Foundation.

Contribution of others who might have given suggestions or review comments.

10. References

List and number all bibliographical references in 9- point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [2-4], [2, 5], and [1].

Please make the references serial no. in Roman Numerals as indicated below.

[i] Briand, L. C., Daly, J., and Wüst, J., "A unified framework for coupling measurement in objectoriented systems", *IEEE Transactions on Software Engineering*, 25, 1, January 1999, pp. 91-121.

[ii] Maletic, J. I., Collard, M. L., and Marcus, A., "Source Code Files as Structured Documents", in *Proceedings 10th IEEE International Workshop on Program Comprehension (IWPC'02)*, Paris, France, June 27-29 2002, pp. 289-292.

[iii] Marcus, A., *Semantic Driven Program Analysis*, Kent State University, Kent, OH, USA, Doctoral Thesis, 2003.

[iv] Marcus, A. and Maletic, J. I., "Recovering Documentation-to-Source-Code Traceability Links using Latent Semantic Indexing", in *Proceedings 25th IEEE/ACM International Conference on Software Engineering (ICSE'03)*, Portland, OR, May 3-10 2003, pp. 125-137.

[v] Salton, G., *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*, Addison-Wesley, 1989.