Journal of Theoretical and Applied Information Technology
15th December 2011. Vol. 34 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MULTICORE PARALLEL LOOSELY COUPLED FRAMEWORK FOR PROCESSING MOBILE OBJECTS

JAWDAT ALSHAER[1], IBRAHIM AL-OQILY[2], SULIEMAN BANI-AHMAD[1], ADNAN ALRABEE[1]

[1] Dept. of Computer Information Systems, Faculty of Information Technology. Al-Balqa Applied University, Salt, Jordan

[2] Faculty of Prince Al Hussein Bin Abdullah II for Information Technology, Hashemite University, Jordan.

E-mail: jawdat_alshaer@hotmail.com, izaloqily@hu.edu.jo, sulieman@bau.edu.jo, adnan_alrabea@yahoo.com

ABSTRACT

Large numbers of mobile objects and continuous queries about them characterize mobile object applications. Efficient and parallel evaluation of queries about mobile objects that continuously move is important for achieving acceptable response times. In such applications, the traditional approaches suffer from the need for parallel updates processing and real time querying and visualization. This results in poor performance in such critical systems. The emerging multicore and manycore microprocessing technologies have the potential to offer scalable performance improvement. How to explore the multicore resources to speed up mobile objects applications is thus a natural question but also a huge challenge for Moving Objects Applications (MOA). In this paper, we propose and evaluate a methodology to explore parallelism via multi-threading, transactional loosely coupled methodology design, and to implement it on multicore processors for Moving Objects Applications. We apply the proposed methodology of the loosely coupled strategy, which has been identified as the key strategy in many design optimizations for different data dependency types, to time-constrained location applications. A parallel framework and its implementation on multicore processors for the proposed strategy have been developed based on multi-threaded concepts. Experiments of using loosely coupled multithreaded parallelization to produce mobile objects parallel query processing and visualization showed clear improvement in terms of speed and efficiency.

Keywords: *Mobile Objects, Parallel Programming, Spatial Temporal, Loosely Coupled Parallel Processing, Multicore Applications.*

1. INTRODUCTION

Research in mobile objects application (MOA) technology has become increasingly important and popular due to recent advances in the technologies of mobile computing, wireless communications, sensor networks, and location services. Many spatial temporal access methods have been proposed. These methods include R-Tree [1], R+-Tree [2], STR-Tree [3] and TPR-Tree [4]. These methods ignore constraints on mobile objects by the underlying transportation networks, speed of objects and the huge processing requirements. In fact, the new trends in multicore programming and the multithreaded programming can be used to efficiently manage network constrained mobile objects. The strategy used in our previous work in [5] to develop TNR+-Tree which manages mobile objects on transportation networks, in regard to their past and current movements and provide support for querying this information reduces number of updates from mobile objects to the database server. Although, it is important to be able to answer queries regarding past and current movements, it is also important to be able to answer queries regarding future movements of mobile objects and visualize these movements; the previous work in this paper is enhanced by merging the proposed algorithms with the multicore technology to efficiently visualize queries about the movement of objects using multi threaded framework to parallel processing parts of the network that concern queries about current state of traffic and predicted future ones. This work is new enhancement to real-time applications concerned with tracing mobile objects as in [24] and in [25]. Taking advantage of the new trends of multicore computing helps in solving the delays of transmission and processing the continuous movement of mobile objects which leads to the ability of monitoring huge amounts of mobile

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

objects. This task was not feasible in unicore processors in the past. This research promises reimplementing the traditional single threaded systems with multi-threaded ones to take advantage of the multicore trends.

2. RELATED WORK.

The applications of locating mobile objects attracts a lot of researchers. Wolfson et al. in [6, 7] firstly proposed a Moving Objects Spatio-Temporal (MOST) model which is capable of tracking the current and near future position of moving objects. Chon et al. in [8] proposed a Space-Time Grid Storage model for moving objects. However, none of these works have considered the constraints on moving objects by the underlying transportation networks and the overhead of the concurrent events in handling locations of moving objects(MO). Vazirgiannis and Wolfson [9] first introduced a model for moving objects on road networks, which connects the moving object's trajectory model with the road network representation. In [10], the authors presented a computational data model for network constrained moving objects in which the road network has two representations namely a two-dimensional representation and a graph representation to obtain both expressiveness and efficient support for queries. In this model, the moving objects treated as query points are represented by graph points located on edges or edges. Ding et al. [11] proposed a MOD model, based on dynamic transportation networks. They model transportation networks as dynamic graphs and moving objects as moving graph points. In addition, Papadias et al. in [12] presented a framework to support spatial network databases. However, these models capture movement information of objects only by their speed and assume the linear movement using serial implementation, which limit applicability in a majority of real applications. Prediction methods for future trajectories of moving objects play an important role in indexing and querying current and anticipated future positions. Most existing prediction methods, used in the indexing and querying, assume linear movement and serial implementation, which cannot reflect the real movement. Aggarwal et al [13] introduced a non-linear model that uses quadratic predictive function. Tao et al [14] proposed a prediction method based on recursive motion functions for objects with unknown motion patterns. In [15], Tao et al developed Venn sampling (VS), a novel estimation method optimized for a set of pivot queries that reflect the distribution of actual ones. These prediction methods improve the precision in predicting the location of each object, but they ignore the constrained movements and the effect of the movement of an object on the others, and thus may not reflect the realistic traffic scenario. Et al in [16] introduced a cellular automaton model describing traffic flow vehicles on transportation network. In Location aware applications for mobile objects , the traditional approaches suffer from the need for the parallel frequent updates processing and real time querying and visualization thereby results in poor performance.. This challenge can potentially be mitigated by emerging multicore and manycore systems.

Since 2004, multicore microprocessor has become the main engine of mainstream servers and personal computers [17], [8]. Nowadays, it is rare to see uni-core processors even in laptop computers, and servers often come with eight cores on one or two CPUs. Therefore, it is natural to hope that manycores available in modern computers may be effectively utilized to speed up MO programs. However, numerous unsuccessful past attempts have shown that, programming model has great influence on usable parallelism; without exploring concurrency in application design, it is impossible to achieve reasonable speedup in multicore or manycore systems. Recently, multicore parallel applications has drawn significant attention in the design automation field [19], [20]. Various existing techniques to explore program concurrency have been borrowed for parallel programming. Automated parallelization is a compilation approach that extracts parallelism from a sequential program. It has been extensively investigated for many years, but has shown limited success [21]. The general consensus in the community is that a program's automatically exploitable concurrency is generally fixed by the programming or the programmer's way of thinking. Conventional sequential programming heavily limits a program's usable concurrency. Message passing approaches explicitly implement a computation by multiple processes that work in separate memory spaces and synchronize via passing messages. It is easy to understand. However, the programming model is at a low abstraction level, closer to the physical platform. Similar to assembly language programming, it requires the programmer to think in physical details and the (even more difficult) concurrent execution of processes. Furthermore, such a program needs to be redesigned for different generations of many-core processors.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Threading (or multithreading) implements a computation using multiple threads that share a common memory space and can be executed concurrently. Thread synchronizations are most commonly achieved by locking. However, coarse-grain locking does not perform well, while fine-grain locking is error-prone. Common problems in fine-grain locking include deadlock and the inability to compose program fragments that are correct in isolation [9]. In addition, it is not known how a programmer can come up with a multithreaded program with correctness guarantee.

3. FRAMEWORK FOR MOBILE OBJECT APPLICATIONS (MOA)

Our modeling of mobile objects on dynamic transportation networks is composed of two steps: first step is the modeling of the Database that stores and indexes transportation network edges and historical and present locations of objects mobile on these edges. For this purpose we used TNR+-Tree as in [5]. Every time an object finishes moving on an edge and starts moving on another, it sends an update of location to the database .The speed on these edges is known and stored. The traffic condition is stored too. In case of change of condition, the edge information should be updated.
The second step is to simulate movements of vehicles on the edges of the network (only edges that crossing query window) considering all factors and infrastructures that affect these movements. For this purpose we used Cellular Automation as in [16].The structure of the proposed architecture for MOA is shown in Fig. 1

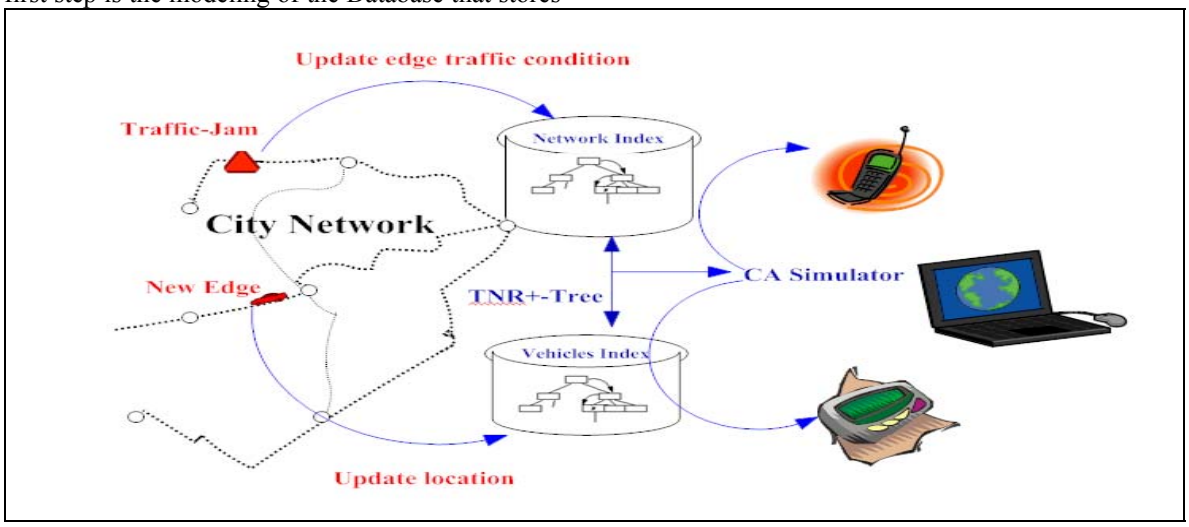

FIGURE 1: MOA ARCHITECTURE

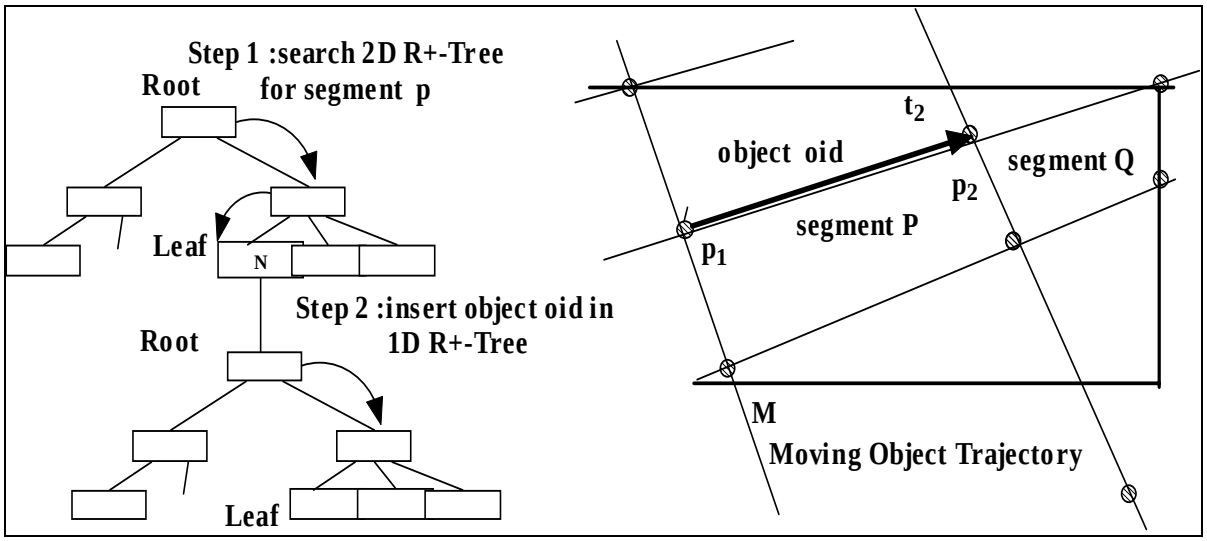

FIGURE 2: TNR+-TREE (INSERTING EDGES AND OBJECTS)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The TNR+-Tree is used in our proposed model to serve as essential database for the network. Stores all necessary information about the network and vehicles moved in edges of this network in different timestamps. The state of the network is stored and efficiently indexed using TNR+-Trees to provide states of edges of the network that is required by queries to be processed by the CA. The general idea for TNR+-Tree is that for a road network consisting of n parts, the TNR+-Tree can be thought of as forest of several 1D TNR+-Trees on bottom of a single 2D TNR+-Tree. The 2D TNR+-Tree is used to index the spatial data of the network (e.g. roads consisting of line edges), while each one of the 1D TNR+-Trees corresponds to a leaf node of the 2D TNR+-Tree and is used to index the time intervals that any mobile vehicle was mobile on a given link of the network. Therefore, the 2D TNR+-Tree remains static during the lifetime of the TNR+-Tree as long as there are no changes in the network. In the TNR+-Tree structure two different kinds of insertion are allowed (see Figure 2):

- Network insertion is performed to construct the transportation Network. The algorithm for network insertion is very simple: just insert the network edges in the top spatial 2D R^+-Tree. The network insertion algorithm takes as arguments the network edge identification, the next network edges identifications, length of the edge, the maximum allowed speed and the minimum allowed speed.
- Movement insertion algorithm takes as arguments the mobile vehicle identification , the finished edge identification , the new edge identification, the last movement coordinates p as in (1) :

$$p=(p_1, p_2). \quad (1)$$

Where p_1 is the start point and p_2 is the end point and the corresponding movement time interval t as in (2):

$$t=(t_1,t_2). \quad (2)$$

To Search a given spatial-temporal query window (x_1, x_2, y_1, y_2, t_1, t_2) for query of the form: "find all objects that have crossed the area r (x_1, x_2, y_1, y_2), during the time interval t = (t_1, t_2)".

The search algorithm receives a spatial-temporal query window (w) and proceeds in three steps:

1. Searching the 2D TNR+-Tree in order to find the line edges of the network contained in the spatial query window. Locate the corresponding 2D TNR+-Tree leaves. Store these line edges in main memory.
2. Searching the 1D TNR+-Trees that correspond to the leaf nodes of Step 1. Retrieve all entries in leaf nodes.
3. Output entries of Step 2, that have line edge identifications equals to edge line identifications in step 1.

3.1. Simulating the State of Transportation Network Edges using CA

In order to process queries about the status of vehicles moving on transportation network edges, We modeled the movements of vehicles on these edges using the Cellular Automation model (CA) which was introduced in this context in [16] .According to this model the edge of network is subdivided into cells, which can be either empty or occupied by one vehicle. The vehicle location is the number of cells behind it to the start node (start point of transportation edge that the vehicle mobile on). Every vehicle has a non-negative integer velocity that represented as (time, location). Fig. 2 shows an example of CA for 3 edges of the network. Each node represents intersection between road edges .The road edge is a CA that connected many cells.

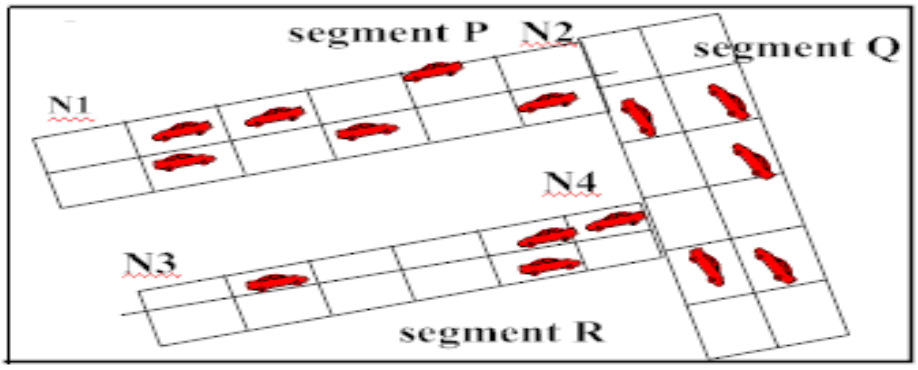

FIGURE 3: REPRESENTING NETWORK EDGES AS CA

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Let i be vehicle moving on an edge. Let v(i) be its velocity, x(i) its position, gap(i) the number of empty cells ahead (forward gap), and Pd(i) a randomized slowdown rate which specifies the probability it slows down. v_{max} is the maximum velocity on the edge. For each update of the road the following four steps are performed simultaneously for all vehicles:

1. Acceleration. The vehicle accelerates until reaching the maximum speed as in (3):
   $v(i)=min\ (v(i)+1, v_{max})$. (3)
2. Avoiding crashes. The vehicle behind other vehicle slow down as in (4) :
   $v(i)=min\ (v(i),gap(i))$.
   (4)
3. Random deceleration(rand). When v(i) needs adjustment, driver may actually overreact and reduce the speed a little more than requirement. v(i) will be reduced to v(i)-1 with a random deceleration probability Pd. As in (5):

If rand < Pd(i) then

$v(i)=max\ (v(i)-1,0)$. (5)

4. Update. The new position of vehicle: Each vehicle is advanced v(i) cells.

3.2. Querying the MOA Structure

Query processing in MOA structure is straightforward. For a client to query the MOA structure, in his handheld device, he clicks the transportation network edge that he wants to query its status, then the top level 2D TNR+ -Tree in the MOA is searched to get the edges of the network that related to the query and then the bottom level 1D TNR+ -Tree to retrieve all vehicles moved on these edges at current time. This information is supplied to CA simulator. User can use the time control in simulator to view past , present or future status of the selected edges, besides the infrastructure objects in these edges.

4. IMPLEMENTING THE MOA USING MULTIPLE LOOSELY COUPLED TASKS

MOA as a *Multiple Loosely Coupled Tasks* is a slight variation on the theme of multiple independent tasks in parallel processing ,where the tasks are different, but they work together to form a single application. Some applications do need to have multiple independent tasks running simultaneously, with each task generally independent and often different from the other running tasks. However, the reason this is an application rather than just a collection of tasks is that there is some element of communication within the system. In MOA the communication runs from central single task to the other task controllers, or the tasks might report some status back to a status monitor. In MOA, the tasks themselves are largely independent. They may occasionally communicate, but that communication is likely to be asynchronous or perhaps limited to exceptional situations. Fig. 4 shows the proposed MOA system running 5 application tasks. Task T is a reporter to the other four loosely coupled tasks, and tasks (T1,..,T4) are a controller tasks to task T.

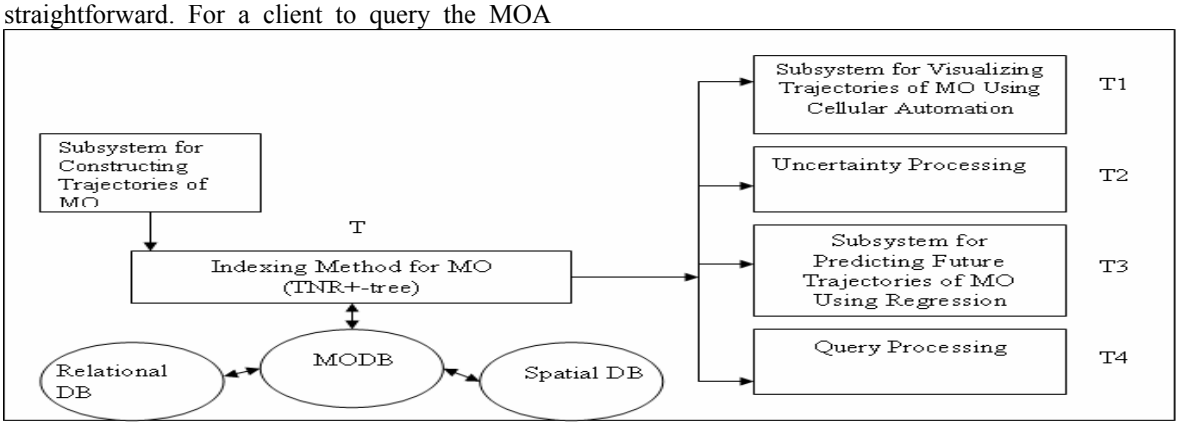

FIGURE 4: PARALLEL REPRESENTATION FOR MOVING OBJECTS APPLICATION

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5. PERFORMANCE OF THE MULTIPLE LOOSELY COUPLED MOA

The performance of the application depends on the activity of these individual tasks. As the CPU-consuming parts of the "application" has been split off into a separate tasks,
then the rest of the components become more responsive the single-threaded application ,T is responsible for database storing and querying MO and the other tasks are responsible for the enhanced querying of MO including visualization and predicting future locations.
The performance gain arises in this case because we have shared the work between Multiple threads. The data storing task only has to process MO and does not get delayed by the other activities. The location processing tasks does not get stalled reading or writing data. If we assume that it takes 1ms to read and forward the location and another 1ms to store to DB, then with the original code, we can process a new MO every 2ms (this represents a rate of 5,000 MO per second).

6. EXPERIMENT RESULTS

We have implemented the multicore MOA application in C++ programming language with Intel Threading Building Blocks [22]. All the experiments are carried out on a Linux server with two dualcore 3.0GHz CPUs and 2GB RAM, which supports up to 4-core parallelism. The multicore program is compiled once and runs with a user-specified number of cores. First, we measured the update cost and query performance for the multicore structure against the single cored. Datasets were generated by the Network-based Generator in [23]. The page size was set to 1024 bytes .The node capacity 100. For the transportation network we produced trajectory datasets of 500, 750, 1000, 1500, 2000.And 10 K, …, 100 K moving objects(vehicles), where each object's position was sampled 400 times. We demonstrate the effectiveness of performance improvement techniques described in Section 5. Because of no determinacy in runtime, the program is run for 30 times on every test case and the results are reported in Table I. The run time is computed against the single core program.

TABLE 1: THE EXECUTION TIME FOR PROCESSING DIFFERENT MO NUMBERS.

Number of MO	Single Thread running time	Threads running time	Speed up
500	1.39	0.22	6.32
750	3.22	0.62	5.19
1000	8.32	1.45	5.74
1250	10.21	1.88	5.43
1500	11.41	2.12	5.38

The results of the experiments showed that running the application of storing the MO then applying some information processing on these MOs as a multithreaded tasks on multicore is in average 5 times faster than running all application in a single thread. This is because the operating system sechdules each thread to be executed on a different core. However, a single-thread application runs on one core.

7. CONCLUSION AND FUTURE WORK

It is desperately needed for computationally intensive MO applications to speed up according to the reality with the increasing number of cores in each generation of microprocessors, since their operating frequencies are largely flattened. We proposed in this paper to use Loosely Coupled approach to explore parallelism in the application design for tracking large number of continuously moving objects; this approach can be successfully used in grouping different types and sometimes unrelated queries(tasks) about mobile objects in one application that efficiently use the multicore technologies. The proposed approach effectively used the multithreaded paradigm applying loosely coupled parallelism to successfully speed up the mobile object processing. This gives the real life system the ability of tracking and monitoring relatively large number of MOs.

REFERENCES

[1]. Guttman A. "R Trees A dynamic index structure for spatial searching". In Proc. 13th Association for Computing Machinery SIGMOD Conference on Management of Data, , 1984; 13:47-57.
[2]. Sellis T, Roussopoulos N, and Faloutsos C. "The R+ Tree: A Dynamic Index for Multi-Dimensional Objects". In Proc. 13th International Conference on Very Large Data Bases, 1987; 13:507-518.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[3]. Pfoser D, Jensen C S, and Theodoridis Y. "Novel Approaches to the Indexing of Moving Object Trajectories". In Proc. 26th International Conference on Very Large Databases, 2000; 26:395-406.

[4]. Saltenis S, Jensen C S, Leutenegger S T, and Lopez M A. "Indexing the Positions of Continuously Moving Objects". In Proc. 2000 ACM SIGMOD International Conference on Management of Data, 2000; 331-342.

[5]. Alshaer J, Gubarev V ."Indexing Moving Objects on Transportation Network". In the 10th International Workshop on Computer Science and Information Technologies (IFOST 2008), NSTU, Novosibirsk,Russia,2008.

[6]. Sistla P, Wolfson O, Chamberlain S, Dao S. "Modelingand Querying Moving Objects". In Proc. 13th Int. Conf. on DataEngineering, 1997; 13:422-432.

[7]. Wolfson O, Xu B, Chamberlain S, Jiang L. "Moving Object Databases: Issues and Solutions". In Proc. of the Tenth International Conference on Science and Statistical Database Management (SSDBM'98), 1998; 10: 111-122.

[8]. Chon H D , Agrawal D,. Abbadi A E. "Using Space Time Grid for Efficient Management of Moving Objects". In Proceedings of the 2nd ACM International Workshop on Data Engineering for Wireless and Mobile Access, 2001; 2:59-65

[9]. Vazirgiannis M, Wolfson O. "A Spatiotemporal Model and Language for Moving Objects on Road Networks". In Proc. of the 13th annual ACM international workshop on Geographic information systems , 2001; 13:20-35.

[10]. Ding Z, G.uting R H. "Managing Moving Objects on Dynamic Transportation Networks". In proceedings of the International Conference on Scientific and Statistical Database Management, 2004; 287-296.

[11]. Chen J, X. Meng X, Y. Gut Y, S. Grumbach S, H. Sun H. "Modeling and Predicting Future Trajectories of Moving Objects in a Constrained Network". In Proceedings of the 7th International Conference on Mobile Data Management (MDM 2006), Nara, Japan. IEEE Computer Society 2006: 156.

[12]. Papadias D, Zhang J, Mamoulis N, Tao Y. "Query Processing in Spatial Network Database". In Proc. of the 29th Conf. on Very Large Databases, 2003; 29: 790-801.

[13]. Aggarwal C, Agrawal D. " Nearest Neighbor Indexing of Nonlinear Trajectories". In PODS, 2003; 252-259.

[14]. Tao Y, Faloutsos C, Papadias D, Liu B. " Prediction and Indexing of Moving Objects with Unknown Motion Patterns". In Proc. of ACM Conference on Management of Data , 2004; 611-622.

[15]. Tao Y, Papadias D, Zhai J, Venn Q L. "Sampling: A Novel Prediction Technique for Moving Objects". In Proc. of the 21st IEEE International Conference on Data, In ICDE, 2005; 21:680-691.

[16]. Nagel K, Schreckenberg M. " A cellular automaton model for freeway traffic". In Journal Physique 1992; I 2. 2221-2229.

[17]. J. F. et al. Design of the Power6TMmicroprocessor. In ISSCC, 2007.

[18]. U. G. et al. An 8-core 64-thread 64b power-efficient SPARC SoC. In ISSCC, 2007.

[19]. B. Catanzaro, K. Keutzer, and B. Y. Su. Parallelizing CAD: A timely research agenda for EDA. In DAC, 2008.

[20]. W. Dong, P. Li, and X. Ye. Wavepipe: Parallel transient simulation of analog and digital circuits on multi-core shared-memory machines. In DAC, 2008.

[21]. J. P. Shen and M. H. Lipasti. Modern Processor Design: Fundamentals of Superscalar Processors. McGraw-Hill Professional, 2005.

[22]. Intel. Threading building blocks. http://www.threadingbuildingblocks.org/.

[23]. Brinkhoff T. "Generating Network-Based Moving Objects". In Proc. 12th Int'l Conference on Scientific and Statistical Database Management, 2000; 12:253-255.

[24]. Anind Dey, Jeffrey Hightower, Eyal de Lara, Nigel Davies (2010): Location-Based Services. *Pervasive Computing* 1/2010, 11-12

[25]. D. Quercia, I. Leontiadis, L. McNamara, C. Mascolo, and J. Crowcroft. SpotME If You Can: Randomized Responses for Location Obfuscation on Mobile Phones. In Technical Report of The University of Cambridge, 2010.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AUTHOR PROFILES:

Jawdat Jamil Alshaer received the BSc degree from the Department of Computer Science, Mu'ta University, Jordan, in 1993, MSc degree from the Department of Computer Science, Wichita State University, USA, in 2003, and PH.D. degree from the Department of Computer Science, Novosibirsk State Technical University, Russian federation. He is currently working as a full time He is presently a full-time lecturer at Al-Balqa Applied University. Alshaer research interests cover topics from Spatial temporal Databases to parallel programming.

Sulieman Bani-Ahmad has received his B.Sc. degree in Electrical Engineering / Computer Engineering from the department of Electrical Engineering Jordan University of Science and technology in 1999. He received an MS in Computer Science from the school of Information Technology at Al-Albayt University in Jordan, in 2001. He received his Ph.D. degree in Computing and Information Systems from the department of Electrical Engineering and Computer Science at Case Western Reserve University, Cleveland - Ohio, USA, in 2008. He is presently an assistant professor at Al-Balqa Applied University. Bani-Ahmad's research interests cover topics from Web-computing and high performance computing.

Ibrahim Al-Oqily received the BSc degree from the Department of Computer Science, Mu'ta University, Jordan, in 1993, MSc degree from the Department of Computer Science, Jordan University, Jordan, in 2003, and PH.D. degree from the Department of Computer Science, the School of Information Technology and Engineering, University of Ottawa, Canada. He is currently working as a vice dean at the IT faculty in the Hashemite university. He also served as a chair for the software engineering department, computer science department, and computer information science department at the IT faculty in the Hashemite University. His current research interests include Overlay Networks Management, Autonomic Management, and policy-based network management. He is a member of the IEEE since 2006.

Adnan Ibarahim Alrabea received the PH.D.Eng. Degree in 2004 from the Electronic and Communication Department, Faculty of Engineering, Donetsk University, Ukraine. He is a visiting Assistant Professor and Assistant dean of Prince Abdullah Bin Ghazi Faculty of Science and Information technology at Al-Balqa Applied University, Alsalt, Jordan. His research interests cover: analyzing the various types of analytic and discrete event simulation techniques, performance evaluation of communication networks, application of intelligent techniques in managing computer communication network, and performing comparative studies between various policies and strategies of routing, congestion control, sub netting of computer communication networks.