OPTIMIZED AD-SCHEDULING FOR E-ADVERTISING USING CONCURRENT MOBILE AGENTS

Roopa G M[1], Dr. Nirmala C R[2]
Department of Computer Science & Engineering,
Bapuji Institute of Engineering and Technology, Davangere

Abstract

e-Advertising is the major source of income for existing Internet ad-websites where maximum revenue is generated by placing advertisements on their web-pages and huge profit generates from Banner-advertisements.Thus, scheduling of advertisements for space-time sharing is challenging task and to address ad-scheduling with specified geometry/display frequency in planned horizon.Constraints with ad-scheduling is that ad-business will not sell the ad-space unless advertisements have predefined ad-shape, scheduling the advertisements with different ad-shapes,websites having multiple display-slots and scheduling/maintaining/tracking of ads is a difficult task for site-admin/third party service providers. This work addresses the problems, by proposing a hybrid framework with (i)Mobile agent paradigm for maintaining/tracking the display-slots and to provide the optimized ad-scheduler by adopting Jquery dynamic div-section to reframe the ad-size. (ii)Client/Sever to allocate the display-space and updates slot by next set of ad-blocks using Ajax-time-control without page refresh. The work focuses on two problems of ad-scheduling, MINSPACE/MAXSPACE and aims to provide an optimized ad-scheduler.

Index Terms: e-Advertising, ad-Scheduling, Ajax-Timer, Banner-Advertisements, J-query, MAXSPACE, MINSPACE, Mobile Agents.

I. INTRODUCTION

Internet Advertising is dramatically increasing due to the popularity of World Wide Web and the enormous usage of Internet. As advertising on the web increases, many firms (Hotmail, Yahoo etc) have adopted business models wherein their revenue is generated from selling advertisement space, while providing free services to their clients. One critical issue with such firms is to maximize their revenue from advertising [2001]. The factors that affect the revenue generation are the cost of advertising based on the number of hits on the page, the ad-size and the ad-frequency. For a web page with the limited amount of space for scheduling the advertisements, the publisher needs to determine the technique to schedule the advertisements at multiple slots by satisfying the advertiser constraints and to either maximize or minimize the performance measures for the website owners.

In an e-Advertising scenario, majority of the website visitors acts as the potential customers for advertisers and hold their customers by purchasing small sections of ad-space for promoting their advertisements on the ad-publisher sites. Thus, an efficient usage of such ad-space to display various advertisements simultaneously is a critical task and scheduling of such advertisements leads to the display of the huge number of ads within the same slots. According to the Internet Ad-Revenue Report from the Interactive Advertising Bureau (IAB), approximately 23% of the advertisements

viewed were in the banner form. Buchwalter et. al. [4] mentioned that 99% of all websites offer standard banner advertisements underlining the importance of this form for online advertising [2006]. For e-Advertising banner acts as a primary form which is a small graphic-image linked to a target page/ad. Various forms of banners advertisements are proposed with different sizes among which the rectangular banner ads are the most commonly used types. Such ads appear on the side/top/bottom of the given target webpage.

The primary issues involved in making the decisions about the assigning/scheduling of advertisements at multiple ad-slots are:

- Number and size of banner slots
- Size of advertisements:
  1. Predefined sizes
  2. Arbitrary sizes
- The number of advertisements to allocate in each slot
- Continuous display of advertisements (Cyclic)
- Splitting the advertisement space into two or more ad-partitions
- Sharing of advertisement space and time between the advertisers
- No overlapping of advertisements
- Issues related to resource minimization and profit maximization
- Percentage of ad-space utilization

Most of the researchers considered two problems MINSPACE and MAXSPACE to address the problem of ad-scheduling . The set of advertisements is competing for display in a specific block for a given planned horizon which are divided into a number of time-intervals called as display-slots. It is observed that, each banner slots are handled separately and the major issue relates to the mapping of advertisements to ad-space in each slot based on the frequency and size of the advertisements. Here, constant managing/monitoring the various banner slots simultaneously at the ad-publisher site (with many web pages) on the web server by the site administrator is a challenging task because frequent requests from the set of advertisers for allocation of advertisement to ad-space increases the load on the site-server for scheduling the advertisements. Majority of the website owners hire companies for constant managing/monitoring the banner ad-campaigns which requires huge manual administrative tasks and heavy investment.

To overcome the major issues related to ad-scheduling, the framework is extended with a hybrid approach which uses concurrent mobile agents with asynchronous and autonomous features and client/server in constant managing/monitoring the various display-slots. This leads to the efficient tracking of multiple display-slots and reduces the manual administrative tasks. Next, the client / server paradigm at the publisher site triggers J-query's dynamic div-section script for creating a new div-section by re-framing ad-size to fit into the remaining ad-space , this results in efficient utilization of ad-space and overcomes the drawback of assumed fixed display-space . Finally , Ajax-Timer control to refresh the advertisements at multiple display slots at regular intervals without refreshing the page.

II. BACKGROUND THEORY

Numerous research works have been carried out in the field of web ad-scheduling. Yager et. al. [1997] proposed a framework that aimed at a competitive means of ad selection on the target websites. Their study described the methodology to use Intelligent-agents to determine the appropriateness for displaying the given advertisements to the website visitors using the specific information about the potential customers. Fuzzy system models were used to build these intelligent-agents.

Aggarwal et. al. [1998] , Alder et. al. [2002] , Kumar et. al. [2006] considered the major issues related to the optimizing the utilization of ad-space on the website. Aggarwal, et. al.[1998] proposed a framework for optimizing the ad-management on the web-servers. Minimum cost-flow model was employed to optimize the ad-assignment within predefined standard banner sizes on webpages. Adler et. al. [2002] presented a heuristic for SUBSET-LSLF for optimal ad-scheduling on webpages with the objective to minimize the total space required to place all the accepted ads. Their research work

was extended by Dawande et. al. [2003] with realistic model and enhanced algorithms. Their work focused on placing multiple advertisements in multiple banners with varying frequency.

Freund and Naor [2004] formulated ad-placement problem which deals with time/space sharing by the advertisements on the Internet. The work considered rectangular display slots that are efficiently utilized by allowing simultaneous side-by-side display of several small ads . Ad-scheduling by adopting cyclic scheme and displaying various ads at different times. The scheduler proposed may either accept /reject the publishing request. (3+ €) approximation was proposed to deal with general problem and (2 + €) approximation algorithms for two special cases.

Amiri and Menon [2006] formulated ad-scheduling problem with an option for the customers to specify the set of accepted display frequencies. They proposed a solution using Lagrangian Decomposition based method to provide better ad-scheduling in a reasonable amount of time.

Boskamp et. al. [2011] proposed a solution for maximizing the revenue by optimal allocation of multiple ads on the web-banner. The problem defined to be single, two dimensional , orthogonal , knapsack-problem , which was applied for ad-pixel. As the defined problem was NP-hard , they proposed various heuristic-algorithms to generate patterns for allocation which was simulated using MAT-LAB.

Deane and Agarwal [2012] extended the research work of Adler et. al.[2002] , Menon and Amiri [2004] , Dawande et. al. [2003, 2005] , Kumar et. al. [2006]. They suggested variable display-frequencies , which can vary between lower and upper bound values. Their results proved that this change significantly improved the ad-space utilization with regard to the fixed display-frequency model. They also included the concept of online ad-targeting for ad-scheduling Problem.

Albers and Passen [2013] suggested the concept of story-boarding where advertiser's presents an ad-sequence (stories) to check the ad-position on the given target web page. In the story-boarding concept , as the user surfs the web and visits the intended website , every advertiser can control a major ad-position for certain predefined period of time. The set of time-slots are used by the advertiser to showcase a variety of products and builds a linear story-line.

RESEARCH GAPS IDENTIFIED:

The major limitations of the online ad-scheduling identified in the literature is most of the research work does not consider the real-time constraints mentioned below:

- Restrictions on the selection of advertisements on a given webpage, not more than once in a specified time-interval.
- To have a desired impression on a particular advertisement , assign an advertisement to the minimum /maximum number of time slots if it is selected for scheduling over a planned horizon.
- To enhance the cost effectiveness of e-advertising , it is necessary to reach the maximum number of target-customers .
- Ad-space partitioning is required for varying ad-dimensions which leads to partition based-costing to provide huge flexibility.
- To provide Banner-sharing for additional revenue generation of web-service providers .
- To reduce the burden on the hiring companies and the manual administrative task of managing /scheduling the advertisements on various ad-slots defined.
- To deal with the situations related to blocking of advertisements over the target web-page.

III. MATHEMATICAL FORMULATION OF AD-SCHEDULING PROBLEM

Consider a given set of N-ads , A = { a_1 , a_2 , a_3, ……., a_n} which are competing for an ad-space within a given ad-slot in a planned horizon. Each planned horizon is divided into N number of equal time-intervals and within each time-interval the accepted advertisements are scheduled/displayed in the rectangular-space called as slots with size (S *W).

Where, S = Pre-defined slot-height and W =Pre-defined slot-width.

In general, the width of all accepted advertisements is same as the width of the defined slot-width W. Here, every ad ai is set with height Sai and frequency Wai . For a given ad ai its height Sai represents the amount of space occupied in the slot and its frequency Wai specifies the number of slots which includes a copy of that ad. But, in real-time scenario, advertisers are not interested in displaying more than one copy of an advertisement in the same-slot and thus an advertisement is displayed only once in a slot.

Definition: An Advertisement ai is scheduled if and only if Wi copies of ai appear in the allocated slot with only one copy in each slot.

Two models have been proposed to address the problem of scheduling online-advertisements:

- MINSPACE problem:

In the MINSPACE problem, all the selected ads need to be scheduled in N-slots. The objective is to define an ad-scheduler with minimum-height fmin .

The ad-presentation scheme followed by a majority of the existing websites is, the ad-width of all the ads placed in a particular slot is same as that of the slot-width. This determines the fullness of the given slot-j as given equation 1. Where, B_{slotj} defines the set of ads which have their copy in slot-j.

The scheduler-height h , is given by maximum slot-fullness as f_{slotj}= (Max_{slotj} x F_{slotj})

$$Full_{Slotj} = \sum_{ai \in Bj} S_{ai} \qquad (1)$$

Ad-ai	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
S_{ai}	6	5	5	4	4	3	2	1
W_{ai}	3	2	1	1	2	1	1	1

Fig. 1(a): Problem instance for ad-scheduling.

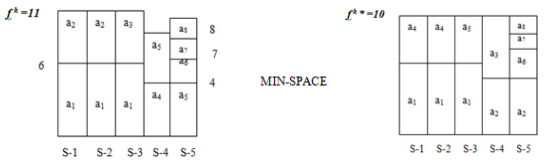

Fig. 1(b): Feasible, Non-optimal scheduling 1(c)Optimal scheduling.

The main objective of the MIN-SPACE problem is to find an ad-scheduler h* with $f^{h*} \leq f^{h}$ for all the feasible ad-schedules h. Here, the problem focuses on scheduling all the selected ads such that the defined slot-height is minimized. Figure 1(c) shows an optimized ad-scheduling with f^{h*} = 10.

- MAXSPACE problem:

In the MAX-SPACE problem, a fixed slot-height S is defined such that each slot has a height of S. Here, the total number of ads-placed (ad-size) cannot exceed the slot-height S. Hence a feasible ad-schedule in such situation must select a sub-set A'⊆ A ads so that the following constraints are satisfied; (i) for each ad $a_i \in A'$, exactly w_{ai} copies need to be placed in the given slots such that only ad- copy must be scheduled/displayed in each slot. (ii) For a given N-slots , the sum of ad-size assigned to slot-j should not exceed S. That is,

$$\sum_{a_i \in B_{slotj}} S_{ai} \leq S \; \forall \; slotj \qquad (2)$$

Where, Bslotj ⊆ A' refers to a set of ads that have a copy in slot-j (sai ≤ S $\forall$ i) . The main objective of MAX_SPACE problem is to define a feasible ad-scheduler for sub-set of ads A' ⊆ A , such that the total slot-weight .

$$\sum_{a_i \in A'} S_{ai} * W_{ai} \qquad (3)$$

For the above problem-instance , Figure 1(d) shows the feasible ad-scheduler with S = 9 with (s_{ai} * w_{ai}) = 42 , where A' includes { a_1, a_2 , a_3 , a_4, a_5, a_6 , a_7 , a_8 } . Figure 1(e) shows an optimal ad-scheduler with (s_{ai} * w_{ai}) =45 where A' includes{a_1, a_2 , a_3 , a_4, a_5, a_6 , a_7 , a_8 }.

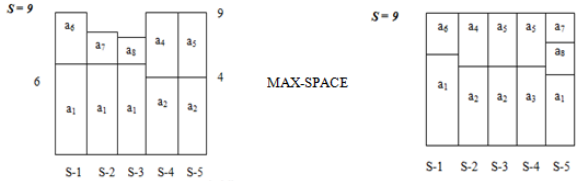

Fig. 1(c): Feasible, Non-optimal scheduling 1(d)Optimal scheduling.

- Solution Procedure:

To address the two major ad-scheduling problems , in this work a hybrid framework is proposed that uses the concurrent mobile agent paradigm and the client/server model to perform the task of scheduling the ads on various ad-slots within a given webpage. In general, online

scheduling of advertisements is usually performed hourly, daily, fortnight, weekly and so on. For example, if the ad-scheduling is updated on an hourly basis, then in total, we have 24 time units. Thus, an advertisement which is displayed on the banner in the current time-interval will be replaced by the next competing ad in the subsequent time-interval.

The pictorial representation of the webpage layout adopted to carry out the task of ad-scheduling is given in figure 2, which includes 3/2 banner ad-slots with pre-defined number of partitions. For the above pre-defined layout , the static agent in the aggregator module creates 3/2 mobile agents for webpage 1 and 2. Here, we have made an assumption that the number partitions assigned to each ad are kept as dynamic over the planned horizon.

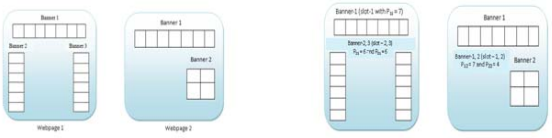

Fig. 2: Pictorial representation of the layout with rectangular banners.

For ad-scheduling, every advertiser needs to provide their preferences about the particular webpage and the banner-slot to schedule/display on the target website. Further, for the given ad-size advertiser needs to specify the minimum (mk) and maximum (Mk) time-units for the Kth advertisement to be displayed on the planning horizon within the predefined rectangular banner slots. Consider, an ad with mk = 5 and Mk = 7 is selected for display within Banner1 of webpage at time unit 3, then the ad can be displayed at-least 5 times and at-most 7 times within the scheduled banner-slot.

To maximize the ad-publisher revenue, the major problem faced by the service-providers is to schedule the ads of varying sizes on various rectangular banner-slots with partitions of the same size of distinct time units. Thus, in order to overcome the problem faced by the service providers and the publisher-admin, we adopt the mobile agent paradigm. The aggregator static agent creates and loads mobile agents with the logic to schedule the advertisements at the ad-publisher sites as per the advertiser preference given and dispatches to the predefined ad-publisher sites. After reaching the specified destination, mobile agent executes the logic at the remote host which in-turn manages/maintains/tracks the specific advertisements and display-slots. This results in faster execution of the advertiser request as there is no dependencies among the mobile agents and mobile agents never miss their deadline in ad-scheduling. It extends to reduce the burden on the publisher-admin / the third party service providers for managing/scheduling the advertisements at various display-slots.

IV. ALGORITHMS DEVISED

- MAX-SPACE Problem: SUBSET- ALSLF (Subset-Agent Based Largest-Size Least-Full) Algorithm

The Mobile Agent based SUBSET-ALSLF algorithm , initially determines the maximum slot-fullness. If the maximum slot-fullness determined is less than / equal to available slot-space for each slot size (S). , then an optimal solution is found., otherwise some of the ads are discarded. Since the MAX-SPACE problem belongs to the class of packing algorithm [13-2001], we propose an agent based SUBSET-ALSLF algorithm to address the MAX-SPACE problem. Mobile agent at each display-slot triggers the procedure to sort the set of ads by ad-size (sai) in the non-decreasing order . It then extends to schedule the ads in the sorted-order such that wai copies of an ad ai is assigned wai maximum full-slots with not more than one-copy of the same ad exists in any slot.

At step-1, ALSLF algorithm initially determines the maximum slot-fullness at every display-slots defined on the webpage . It checks if the maximum slot-fullness is less-than or equal to slot-size S, then stops at step-2. If the maximum slot-fullness is greater-than the slot-size S , then the mobile agent selects the ads with smaller values of (sai * wai) and discards(pushes) these ads to the buffer until the maximum slot-fullness becomes less-than or equal to S. Once this condition is met, mobile agents add some of the discarded (buffered) ads in the next time interval .Such that the maximum slot-fullness remains less-than or equal to slot-size S. Finally the subset of ads A' in each iteration are assigned to the slots.

Step-1 : Create/load/dispatch concurrent mobile agents for ad-scheduling.

Step-2 : Mobile Agent at each display-slot triggers

SUBSET-ALSLF procedure , to determine the maximum fullness f of slot-j.

Step-3 : Check if f ≤ S,(Maximum slot size) terminates , otherwise goto step-4.

Step-4 : Let Bi = (sai * wai) for ad ai , compute Bi for all ads i={1,2,……..n}.

Step-5 : With the value of Bi , sort the ads from smallest-to-largest.

Step-6 : Set the value of K=1, mobile agent removes the first-sorted ad from the scheduled list , and again trigger SUBSET-ALSLF to determine the value of maximum fullness f. Removed ad is stored as Discard(K) (buffer).

Step-7: Check if f ≤ S,(Maximum slot size) goto step-9, otherwise goto step-8.

Step-8 : Increment the value of K= K +1 , mobile agent removes the next-sorted ad from the scheduled list and again trigger SUBSET-ALSLF to determine the value of maximum slot-fullness f. Removed ad is stored as Discard(K). Goto step-7.

Step-9 : Check if K =1 terminate , else add the ad to Discard(K-1) (buffer)in the scheduled list and mobile agent triggers SUBSET-ALSLF to determine the value of maximum slot-fullness f.

Step-10 : Check if f >S, then goto step-11, otherwise goto step-12.

Step-11 : Remove the ad Discard(K–1) from the scheduled list

Step-12: Set the value K = K -1, goto step-9.

- MIN-SPACE Problem: ALSLF / ASSLF (Agent based Largest Size Least full)/ (Agent based Smallestnsize Least Full)Algorithm

The objective of the mobile agent based ALSLF / ASSLF algorithm , is to find a ad-scheduler to assign all the selected ads such that the height of the schedule is minimized , to ensure that for a given ad ai , exactly wi copies are assigned for display-slots and at most one copy of the ad ai is assigned to a slot-j.

Step-1: Create/load/dispatch concurrent mobile agents for ad-scheduling

Step-2 : Sort the list of ads in the non-increasing order of their ad-size $\{s_1 \geq s_2 \geq s_3 \ldots\ldots\ldots \geq s_n\}$.

Step-3: Assign the ads based on the sorted order, where the ad a_i is assigned for w_i least full slots.

At step-1, the given ads-list is sorted in the non-decreasing order based on the ad-size. The mobile agent starts assigning the ads from the sorted-list to the particular display-slot and checks that all the w_i copies of the ads are scheduled in such a way that the height of the schedule is minimized.

V. EXPERIMENTAL SETUP

To conduct the simulations , the setup was made in the CS&E Labs of Research Centre with the computing machines having the similar system specifications of 4GB RAM , 1TB Hard Disk, i3/i5 core processor, connected through 10/100/1000Mbps LAN running on Microsoft Windows-7. ASDK-2.0.2 (Aglet Software Development Kit) is used for creating/loading/dispatching the Mobile Agents to perform the task of ad-scheduler at various slots. Next, we use Jquery UI Widget Factory for re-framing the advertisement size based on the available slot-partition, to allocate the slot space for scheduled advertisements and to update/load the next block of advertisements over the slots after a certain time interval by adopting Ajax Time control without page refresh. MySQL is used for storage purpose.

The algorithms defined for two major problems of Ad-scheduling MINSPACE and MAXSPACE like ALSLF / ASSLF and SUBSET- ALSLF are implemented using JAVA. The stack implementation is used at every slot for storing and loading the selected advertisements at specified time interval set for each slots. For the case study described , five display slots are defined which are implemented with five individual stacks and each stack is maintained/tracked by individual mobile agnet.

VI. CASE STUDY

For a given set of ads A = { a_1 , a_2 , ……a_n } , let k set of ads are competing for display over the planned horizon. Subset S_{xyt} refers to the set of ads that needs to be placed on the xth banner of yth webpage at tth time unit. Thus, for each selected advertisement the aggregator static agent forms the S_{xyt} .For example, if an ad is selected with preference S_{123} ,then it is part of 1st rectangular banner-slot of 2nd webpage at time-unit 3. Let us assume that the advertiser-k with $A_k \in S_{xyt}$ pays C_{xy} cost for per partition-unit/ per time-unit for placing the ad on xth banner-slot on yth webpage.

Let the total number of existing banners for each time-interval is given b $\sum_{y=1}^{n} B_y$ and the number of partitions on each banner is different. The mobile agent based ad-scheduler aims at generating the maximum revenue by efficiently placing the advertisements in the planned horizon for the given target web page or website. Generating the revenue from Advertising can be defined as the function which measures the website traffic to attract the customers and the procedures used by the mobile agents to present the advertisements for the website visitors. In existing websites , the ad-revenue mainly depends partition cost paid by the advertiser for displaying the advertisements on the xth banner of yth webpage at tth time-unit. Finally, the cost associated with a particular banner depends on the displaying advertisements and the ad-click rates.

Table 1: Semantic-Meaning of variables used.

Notation	Semantic-Meaning
N	Number of web-pages defined for a target website
B_y	Number of banner-slots on yth webpage
R_{xy}	Number of partitions in xth banner-slot of yth webpage
A	Total number of advertisements
MA	Total number of concurrent mobile agents created
T	Number of time-units for the planned horizon
C_{xy}	Cost for per partition-unit/ per time-unit for placing the ad on x^{th} banner-slot on yth webpage.
Ssub	Subset of selected ads from A
Sxyt	Set of ads competing for the xth banner-slot of yth webpage at t^{th} time unit
A ak	The current kth advertisement
Mak	Maximum time-units for kth ad to be displayed
a ak	Number of partitions needed for kth ad
Pxylkt	1 if kth ad is placed on lth partition of xth banner-slot on yth webpage at tth time-unit. 0 otherwise
Qxykt	1 if kth ad is placed on xth banner-slot on yth webpage at tth time-unit. 0 otherwise
Z ak	1 if kth ad is selected for display Otherwise.

The assumption made by the mobile agent to maximize the ad-publisher revenue can be computed as:

$$Maximize\ Ad-Revenue\ (R) = \sum_{y=1}^{n}\sum_{x=1}^{B_y}\sum_{l=1}^{R_{xy}}\sum_{A\in P_{xyt}}\sum_{t=1}^{T} C_{xy}\, P_{xylkt} \qquad (4)$$

Constraint applied by the mobile agents to ensure that the ath advertisement is scheduled at-least W_k time-slots if it is selected for scheduling over the given planned-horizon is given by:

$$\sum_{y=1}^{n}\sum_{x=1}^{B_y}\sum_{l=1}^{R_{xy}}\sum_{t=1}^{T} P_{xylkt} \geq M_{ak} * a_{ak} * z_{ak} \qquad \forall k \in S_{xyt} \qquad (5)$$

Constraint applied by the mobile agents to ensure that ath advertisement is assigned at a most M_k time-slots if it is selected to be scheduled over the given planned horizon is given-by:

$$\sum_{y=1}^{n}\sum_{x=1}^{B_y}\sum_{l=1}^{R_{xy}}\sum_{t=1}^{T} P_{xylkt} \leq M_{ak} * a_{ak} * z_{ak} \qquad \forall k \in S_{xyt} \qquad (6)$$

Constraint applied by the mobile agents to ensure that the number of partitions required by the a^{th} advertisement selected for display on the predefined banners of yth web page at t^{th} time-unit is equal number of partitions defined for the Bth banner of y^{th} web page at t^{th} time-unit is given by:

$$\sum_{t=1}^{R_{xy}} P_{xylat} = a_{ak} * P_{xykt} \qquad \forall x, y, t, k \in S_{xyt} \qquad (7)$$

Constraint applied to ensure that the total number of banner partitions required by all

selected advertisements for display on the x^{th} rectangular-banner of the y^{th} webpage at tth time-unit does not exceed the predefined number of partitions of x^{th} banner is given by:

$$\sum_{l=1}^{B_y} \sum_{k=S_{xyt}} P_{xylkt} \leq R_{xy} \qquad \forall x, y, t \qquad (8)$$

Constraint applied to ensure that , if an advertisement is selected for display on the webpage for any given time period, then that particular advertisement cannot appear on that webpage for the same time-period is given by:

$$\sum_{l=1}^{B_y} \sum_{l=1}^{R_{xy}} P_{xylkt} = a_{ak} \qquad \forall y, t, k \in S_{xyt} \qquad (9)$$

Constraint applied to ensure that for a particular time-unit and for each banner partition on a given webpage only one advertisement can be placed which is given by:

$$\sum_{k \in S_{xyt}} P_{xylkt} \leq 1 \qquad \forall x, y, l, t \qquad (10)$$

Constriants applied to ensure that the number of times a particular advertisement k (selected for display) appears over the predefined planned horizon can be one/more than one and is given by:

$$\sum_{k \in S_{xyt}} Z_{ak} \geq 1 \qquad (11)$$

Combining the constraints from equations (1) - (11) a mobile agent based automatic ad-scheduling model is formulated which solves the scheduling problem in the existing scenario where decision-makers have more than one task to perform and results in feasible/infeasible solutions. A mobile agent at each banner ensures the efficient utilization of display space by scheduling the advertisements by filling completely for all time-slots.

Let us consider a set of 100 ads S = { a_1 , a_2 , a_3 ,……., a_{100}} which are competing to be placed on the webpage 1 and 2 with various display-slots .From Figure 2 , in total we have 5 banner slots and thus the aggregator static agent creates five concurrent mobile agents for tracking and scheduling the advertisements at predefined banner slots simultaneously. In the real-time scenario displaying the ads competing for ith banner on j^{th} webpage at all the time units is not feasible, thus we present the data pertaining to the randomly selected time units [1]. The details of the ad-frequency and number of ad-partitions requirement is given in Table2.

The number of rectangular banners considered for a given webpage is $B_1 = 2$ and $B_2 = 2$ with paritions made within each banners is $P_{11} = 7$, $P_{21} = 6$, $P_{31} = 6$, $P_{12} = 7$ and $P_{22} = 4$. Here , the planned horizon is assumed to be set for 1 day. As 1-day holds 24 hrs and advertisements are updated for every 2 hrs.Then, in total we have 12 time-units (1 * (24/2)) to schedule the advertisements.

Table 2: Details of the ad-frequency and number of required ad-partitions.

Ads	M_{ak}	a_{ak}	slot	Ads(A_{ak})	M_{ak}	a_{ak}	slot	Ads	M_{ak}	a_{ak}	slot	Ads	M_{ak}	a_{ak}	slot
a_1	30	3	1	a_{26}	25	6	3	a_{51}	30	3	1	a_{76}	25	6	3
a_2	35	4	2	a_{27}	20	5	1	a_{52}	35	4	2	a_{77}	20	1	1
a_3	18	3	1	a_{28}	22	1	2	a_{53}	18	3	1	a_{78}	22	4	2
a_4	10	2	2	a_{29}	15	1	3	a_{54}	10	2	2	a_{79}	15	2	3
a_5	30	4	3	a_{30}	32	3	1	a_{55}	30	4	3	a_{80}	32	1	1
a_6	22	5	2	a_{31}	10	2	3	a_{56}	22	2	2	a_{81}	10	2	3
a_7	18	7	1	a_{32}	14	5	3	a_{57}	18	7	1	a_{82}	14	5	3
a_8	16	6	2	a_{33}	28	2	2	a_{58}	16	4	2	a_{83}	28	4	2
a_9	10	5	1	a_{34}	17	3	1	a_{59}	10	5	1	a_{84}	17	2	1
a_{10}	28	2	3	a_{35}	14	3	2	a_{60}	28	4	3	a_{85}	14	3	2
a_{11}	30	1	1	a_{36}	17	5	3	a_{61}	30	2	1	a_{86}	17	5	3
a_{12}	15	2	3	a_{37}	45	6	1	a_{62}	15	4	3	a_{87}	45	6	1
a_{13}	34	1	2	a_{38}	32	6	2	a_{63}	34	3	2	a_{88}	32	6	2
a_{14}	20	6	3	a_{39}	40	3	2	a_{64}	20	6	3	a_{89}	40	3	2
a_{15}	16	5	2	a_{40}	26	3	1	a_{65}	16	2	2	a_{90}	26	4	1
a_{16}	35	2	1	a_{41}	44	4	3	a_{66}	35	4	1	a_{91}	44	4	3
a_{17}	15	3	1	a_{42}	29	3	2	a_{67}	15	3	1	a_{92}	29	2	2
a_{18}	53	3	2	a_{43}	18	4	3	a_{68}	53	3	2	a_{93}	18	4	3
a_{19}	45	3	2	a_{44}	27	1	1	a_{69}	45	3	2	a_{94}	27	2	1
a_{20}	52	4	3	a_{45}	33	3	2	a_{70}	52	2	3	a_{95}	33	3	2
a_{21}	25	4	1	a_{46}	14	2	3	a_{71}	25	4	1	a_{96}	14	3	3
a_{22}	42	2	1	a_{47}	15	4	1	a_{72}	42	3	1	a_{97}	15	4	1
a_{23}	54	1	2	a_{48}	20	5	2	a_{73}	54	2	2	a_{98}	20	1	2
a_{24}	10	4	1	a_{49}	34	3	3	a_{74}	10	4	1	a_{99}	34	5	3
a_{25}	40	5	2	a_{50}	22	2	2	a_{75}	40	1	2	a_{100}	22	2	2

Table 3: Advertisements competing for various display slots on webpage 1 and 2.

Advertisements competing for slot-1(webpage 1) Managed by Mobile Agent-1

Ads	M_{ak}	a_{ak}	slot
a_1	30	3	1
a_3	18	3	1
a_7	18	7	1
a_8	10	2	1
a_{11}	30	1	1
a_{16}	35	2	1
a_{17}	15	3	1
a_{21}	25	4	1
a_{22}	42	2	1
a_{23}	10	4	1
a_{27}	20	5	1
a_{28}	32	3	1
a_{31}	17	5	1
a_{37}	45	6	1
a_{38}	26	3	1
a_{44}	27	1	1

Advertisements competing for slot-1 (webpage 2) Managed by Mobile Agent-4

a_{37}	15	4	1
a_{51}	30	3	1
a_{53}	18	3	1
a_{57}	18	7	1
a_{58}	10	2	1
a_{61}	30	2	1
a_{66}	35	4	1
a_{67}	15	3	1
a_{71}	25	4	1
a_{72}	42	3	1
a_{74}	10	4	1
a_{77}	20	1	1
a_{80}	32	2	1
a_{81}	17	1	1
a_{87}	45	6	1
a_{88}	26	4	1
a_{91}	27	1	1
a_{97}	15	4	1

Advertisements competing for slot-2 (webpage 1) Managed by Mobile Agent-2

Ads	M_{ak}	a_{ak}	Slot
a_2	35	4	2
a_4	10	2	2
a_6	22	5	2
a_9	16	6	2
a_{13}	34	1	2
a_{15}	16	5	2
a_{18}	53	3	2
a_{19}	45	3	2
a_{23}	54	1	2
a_{25}	40	5	2
a_{26}	22	1	2
a_{29}	28	2	2
a_{33}	14	3	2
a_{35}	32	6	2
a_{39}	40	3	2
a_{42}	29	3	2
a_{43}	33	3	2
a_{45}	20	5	2

Advertisements competing for slot-2 (webpage 2) Managed by Mobile Agent-5

a_{56}	22	2	2
a_{62}	35	4	2
a_{63}	10	2	2
a_{66}	22	2	2
a_{68}	16	4	2
a_{69}	34	3	2
a_{69}	16	2	2
a_{68}	53	3	2
a_{69}	45	1	2
a_{73}	54	2	2
a_{73}	40	1	2
a_{73}	22	4	2
a_{83}	28	4	2
a_{83}	14	1	2
a_{88}	32	1	2
a_{89}	40	3	2
a_{92}	29	2	2
a_{93}	33	3	2
a_{93}	20	1	2
a_{100}	22	2	2

Advertisements competing for slot-3 (webpage 1) Managed by Mobile Agent-3

Ads	M_{ak}	a_{ak}	slot
a_5	30	4	3
a_{10}	28	2	3
a_{12}	15	2	3
a_{14}	20	6	3
a_{20}	52	4	3
a_{24}	25	6	3
a_{29}	15	1	3
a_{31}	10	2	3
a_{32}	14	5	3
a_{36}	17	2	3
a_{41}	44	4	3
a_{43}	18	4	3
a_{46}	14	2	3
a_{48}	34	3	3
a_{50}	30	4	3
a_{60}	28	4	3
a_{62}	15	4	3
a_{64}	20	5	3
a_{70}	52	2	3
a_{76}	25	6	3
a_{79}	15	2	3
a_{81}	10	2	3
a_{82}	14	5	3
a_{86}	17	2	3
a_{91}	44	4	3
a_{93}	18	3	3
a_{96}	14	3	3
a_{99}	34	1	3

Table 4: Ad-Scheduling at ith Banner of jth webpage in time unit from (1- 12)

Time slots (t)	Banners				
	Slot-11	Slot-21	Slot-31	Slot-12	Slot-22
1.	a_1, a_3, a_{11}	a_2, a_1	a_5, a_{10}	a_{47}, a_{53}	a_{50}, a_{54}
2.	a_7	a_8	a_{12}, a_{20}	a_{57}	a_{52}
3.	a_9, a_{16}	a_6, a_{13}	a_{14}	a_{59}, a_{61}	a_{58}
4.	a_{17}, a_{21}	a_{18}, a_{19}	a_{26}	a_{53}, a_{66}	a_{56}, a_{65}
5.	a_{27}, a_{22}	a_{15}, a_{23}	a_{31}, a_{41}	a_{67}, a_{71}	a_{68}, a_{75}
6.	a_4, a_{30}	a_{25}, a_{28}	a_{36}, a_{43}	a_{72}, a_{74}	a_{63}, a_{69}
7.	a_{44}, a_{31}	a_{35}, a_{39}	a_{29}, a_{32}	a_{80}, a_{84}, a_{90}	a_{78}
8.	a_{34}, a_{40}	a_{38}	a_{64}	a_{77}, a_{87}	a_{83}
9.		a_{42}, a_{45}	a_{76}	a_{94}, a_{97}	a_{88}, a_{92}
10.		a_{33}, a_{48}	a_{55}, a_{79}		a_{85}, a_{89}
11.			a_{60}, a_{81}		a_{88}, a_{92}
12.			a_{62}, a_{70}		a_{95}, a_{98}, a_{100}
1.			a_{46}, a_{91}		
2.			a_{49}, a_{96}		
3.			a_{82}, a_{99}		
4.			a_{86}, a_{93}		

The set of stages involved in scheduling the advertisements at various display slots is:

Stage 1: Optimized ad-scheduler plan

The aggregated static agent selects the accepted set of advertisements from the aggregated advertisements-list, which are competing for display at various slots on the target webpage. Then, it creates concurrent mobile agents and extends to load these agents with the specified advertisements to allocate the space on the particular slot based on the advertiser preference. The task of ad-scheduler is carried out by individual mobile agents at every predefined ad-slots, which uses the resizable procedure of j-query to reframe the advertisement-size when the load event takes place. Here, each slot is identified as #container with predefined width and height (Leaderboard(728X90), skyscrapers(120X600)and square pops(250X250)) and is associated with a unique id. For each container we trigger the procedure #resizable and use the property padding with the value of ± 0.5/1.0 inches based on the partition size to handle the Max/Min width and height.
For the above case study, the scheduling decision plan made by the mobile agents for the 5 slot on web pages 1 and 2 is given in Table 4 for time units from (1 -12). The dispatched mobile agents load this decision plan at the various display slots.

Stage 2: Executing/Displaying the advertisements according to the scdeduler.

Concurrent Mobile Agents maintain/tracks the set of advertisements at each slot on the target webpage. Once the advertisements are loaded, Ad-Rotator Control is triggered which uses the AppendTo( ) method of J-Query to allocate the slot space for the selected advertisements for the advertiser preference. In this case study, we have implemented five individual Ad-Rotators for 5 slots. The aggregated advertisements-list is divided into blocks which assign a key word to differentiate with the other blocks that belong to other slots. This keyword is used to bind the advertisements to the Ad-Rotator with the same keyword. After binding the advertisements to the Ad-Rotator , the advertisements are allocated the space and are displayed on the slots for the defined number of frequencies.
A code snippet (asp.net) which creates an Ad-Rotaor:

```
<asp: Ad-Rotator
Id=”Ad-Rotator-slot-1”
Advertisement-Files=”- / Ad-Data/slot-1”
Keyword-Filter=”slot-1”
runAt=”server”/>
```

Stage 3: Asynchronous advertisement refreshes for scheduled time interval.

To rotate the advertisements on every predefined slot without page refresh, an Ajax timer code at every slot is added. To carryout this task, the Ad-Rotator is associated with the UpdatePanel( ) procedure and then use the timer-control to refresh the advertisements at regular intervals. The resultant code with the Ad-Rotator and the Ajax-Time control is given below,

```
<form id= “slot-1” runat= “server”>
<div>
<asp:Script-Manager
ID=“Script-Manager-1”runat= “server” />
<asp: Timer ID= “Timer-1” Interval = “1000”
runat= “server” />
  <asp:Update-Panel ID= “UPD-1” runat=
“server” />
  <Triggers>
    <asp:AsyncpostBackTrigger
Control-ID=“Timer-1”
                EventName= “Tick” />
  </Triggers>
<ContentTemplate>
  <asp: Ad-Rotator
        Id=”Ad-Rotator-slot-1”
```

```
        Advertisement-Files="-                    /
Ad-Data/slot-1"
        Keyword-Filter="slot-1"
        RunAt="server"/>
</ContentTemplate>
</asp:UpdatePanel>
</div>
</form>
```

The code uses triggers inside the updatepanel, which enables to refresh the advertisements over a specific time interval. The event captured is a "Tick" event of the Ajax Time control. Thus, every-time the timer-control raises a Tick-Event, an asynchronous Postback occurs and the contents are loaded to the UpdatePanel with different Ad-Blocks which are refreshed at regular intervals on every slot.

VII. CONCLUSION

In the realtime scenario, as the number of advertisers competing for the allocation of display-slot increases it is observed that there requires a continuous mapping between the advertisements and the ad-slot. This in-turn becomes a difficult task for the site-administrator or the third party service provider for maintaining/tracking the advertisements based on the advertiser preference given. Thus in this research work, a hybrid framework is formulated to address the two major problems of ad-scheduling which are MAXSPACE and MINSPACE.

The work mainly focuses on the most commonly used three different banner types like Leaderboard, Skyscraper and square pops.The framework presents a concurrent mobile agent optimized ad-scheduler for efficient maintaining/tracking of advertisements at various display-slots simultaneously. It incorporates the concept of slot-sharing, reframing of ad-size for efficient slot-space utilization , restricts on the selection of an advertisement on the same slot not more than once at any given time-unit in such a way that maximum revenue is generated.

A case study is presented , to show the efficient scheduling of advertisements by concurrent mobile agents. The major key contribution of this work is to incorporate Mobile Agents with the concept of Artificial Intelligence to automatically and asynochronously perform the task on behalf of the advertiser and the site-administrator which reduces the manual work of posting and maintaining/tracking the advertisements.

The major limitations of this work is that we have not taken into consideration about the advertising bugets, the total cost of an advertiser ad-campaign. The future work can be extended to show the comparative analysis of the various algorithms adopted for ad-scheduling. Also, ad-scheduling problem can be further explored by considering other scheduling types like priority based scheduling.

REFERENCES

[1] Nitish Korula , Vahab Mirrokni , Hamid Nazerzadeh "Optimizing Display Advertising Markets, Challenges and Directions" IEEE Internet Computing January 2016.

[2] Prerna et. al. "A Goal Programming model for selectopn and scheduling of advertisements on online News Media." Asia-Pacific journal of Operational Research Vol. 33, No. 2 (2016) 1650012 (41 pages) © Worls Scientific Publishing Co. and operational Research Society of Singapore.

[3] Hongyu Cao and Yue Gao "Research and Implementation of Using Ajax Technology to improve the user experience" 2011 International Conference on Mechatronic Science, Electrical Engineering and computer, August 19-22, 2011 Jilin, China.

[4] Victor Boskamp, Alex Knoops, Flavius Frasincar, Adriana Gabor "Maximizing revenue with allocation of multiple advertisements on a web banner" Econometric Institute ,Erasmus University Rotterdam, Netherlands.Preprinted submitted to Computer & Operations Research, Dec 8 , 2010.

[5] Aif Kimms and Michael Muller Bungart "Revenue Management for broadcasting commercials: the channel's problem of selecting and scheduling the advertisements to be aired. International Journal Revenue Management Vol. 1, No. 1, 2007. Inderscience Enterprises Ltd.

[6] Ali Amiri and Syam Menon " Scheduling Web Banner Advertisements with Multiple display Frequencies" IEEE Transactions on Systems,Man and Cybernetics – Part A:

System and Humans Vol 36, No. 2, March 2006.

[7] Milind Dawande, Subodha Kumar and Chelliah Sriskandarajah " Scheduling web advertisements : A Note on the MINSPACE Problem, Journal of Scheduling 8: 79-106, 2005 Springer Science and Business Media.

[8] Subodha Kumar, Varghese S Jacob, Chelliah Sriskandarajah "Scheduling Advertisements on Web page to maximize revenue" European Journal of Operational Research 173 (2006) 1067-1089.

[9] Ari Freund and Joseph (Seffi) Naor "Approximating the Advertisements Placement Problem" Journal of Schedulingn7: 365-374, 2004, Kluwer Academic Publishers.

[10] Milind Dawande, Subodha Kumar and Chelliah Sriskandarajah "Performance Bounds of Algorithms for Scheduling Advertisements on Web page" Journal of Scheduling 6: 373 – 393, 2003 , Kluwer Academic Publishers.

[11] Ali Amiri and Syam Menon "Efficient Scheduling of Internet Banner Advertisements" ACM Transactions on Internet Technology, Vol 3, No. 4. Nov 2003, Page 334 – 346.

[12] Milind Dawande, Subodha Kumar and Chelliah Sriskandarajah " Hybrid Genetic Algorithms for Scheduling Advertisements on a web page" 2001 – Twenty Second International Conference on Information Systems.

[13] Micah Adler, Philip B Gibbons and Yossi Matias " Sxheduling Space-Sharing of Internet Advertising" , March 17, 1998.

[14] Ronald Yager " Intelligent Agents for World Wide Web Advertising Decisions" International Journal of Intelligent System, Vol. 12, 379-370, John Wiley and Sons, Inc., 1997.

[15] Charu C , Arrarwal , Joel L and Philip S , Yu. " A Framework for the optimizing of WWW Advertising."