ULTR-CTR: Fast Page Grouping using URL Truncation for Real-time Click Through Rate Estimation

Hui Liu[◇], Xingquan Zhu[◇], Kristopher Kalish[†], and Jeremy Kayne[†]
[◇]Dept. of Computer & Electrical Engineering and Computer Science, Florida Atlantic University
Boca Raton, FL 33431, USA
[†]Bidtellect Inc., Delray Beach, FL 33483, USA
{liuh2015, xzhu3}@fau.edu; {kris, jeremy}@bidtellect.com

***Abstract*—Click Trough Rate (CTR) estimation is a crucial measure (or procedure) in online digital advertising (Ad). It defines the probability of a displayed Ad being clicked by viewers, and can serve as a performance metric to validate the effectiveness of Ad campaigns with respect to pages, sites, or media types etc. Due to real-time response nature of the online digital advertising eco-systems, it is vital to accurately estimate the CTR in real-time. In this paper, we propose a URL truncation based fast page grouping for real-time CTR estimation (ULTR-CTR). Our hypothesis is that web pages under the same URL folder have similar page style and semantic content, and will share similar CTR values. While grouping web pages based on the page content is computationally expensive and hardly scalable to real-time applications, we use simple URL truncation to estimate CTR values of different site-folder combinations. Our empirical study and A/B test carried out on a commercial bidding engine confirm that ULTR-CTR based bidding achieves 2.0% performance gain in CTR estimation, and 1.4% lift in Gross Profit (GP) gain.**

I. Introduction

Real-time bidding (RTB) is an eco-system which allows different parties to buy and sell on-line advertisement (Ad) inventory on a real-time basis [1], [2]. Whenever a user visits a publisher's page which contains one or multiple banners for advertisement (Ad) display, or visits a search engine to search for information [3], it creases an Ad inventory, allowing advertisers to display their Ad on the banner showing on users' end devices. To facilitate real-time information exchange, an Ad exchange aggregates Ad inventories and notifies potential Ad buyers, advertisers, via a bid request. Advertisers will utilize a Demanding Site Platform (DSP) to calculate the value of the Ad inventory, and place bid through a bid response, if necessary. For each Ad banner, the advertiser offering the highest bid will win the bid [4] and can immediately display their Ad on the page displayed to the user.

Each time an Ad is displayed to viewers, it is counted as one *impression*, and if the user clicks on the Ad displayed on the page, it is counted as a *click*. Fig. 1 illustrates a simplified view of the real-time bidding system. The whole process usually finishes in milliseconds so online users do not experience unpleasant delay. One significant advantage of RTB is that advertisers can buy Ad impressions that are most relevant to their Ads, with very little costs. Such a system provides immediate feedback about the performance of Ad campaign, with significantly reduced costs and wastes, compared to traditional media advertising.

In online digital advertising, three types of revenue models are commonly used: impression based (CPM), click-based (CPC), and action based (CPA) [2]. In this paper, we mainly focus on the CPC based campaigns, where advertisers' revenue is based on the number of clicks generated from users multiplying the price of each click.

Because the revenue of CPC campaigns is determined by the user clicks, advertisers will develop bidding strategies to maximize the revenue. For Ad campaigns which aim to acquire user's clicks and direct them to specific landing pages, the value of an Ad inventory is $CPC \times CTR$, so the value of Ad inventory is determined by Click Through Rate (CTR), which defines the average possibility of an Ad impression producing a user click. Because the best bidding strategy in second price auction (prevail in RTB) is to bid the true value, CTR estimation is essential in bid price determination. Furthermore, RTB system needs Ad inventory purchase to be finished in millisecond, which requires the CTR estimation to be carried out frequently or in real-time [5].

In order to estimate the CTR, existing approaches can be roughly separated into two groups (1) *generative modeling*, and (2) *predictive modeling*. In generative modeling, historical data are used to build a generative models, whose parameters are used to derive the CTR value of a new impression. Common generative models include CTR hierarchy trees or hierarchical Bayesian framework [5]. Predictive modeling, on the other hand, treats user clicks as binary events, and uses supervised learning to train a classifier to predict the likelihood of an impression being clicked by users [6], [7], [8], [9], [10].

Many predictive CTR modeling methods exist which mainly emphasize on predicting the posterior probability of an impression being clicked by a user. For example, search advertising often extracts key features such as landing page URL, key words, Ad title, Ad text, etc. and uses logistic regression model to predict whether a search advertisement will be clicked [11], [12]. Predictive model, such as decision trees, can be built based on features that are strictly related to the quality of Ads (number of words used in the title of an Ad, number of segments and length of the landing page URL, individual words and terms used in the Ad title

and the Ad body, etc.). Because decision rules are easy to interpret, they can be used to improve the quality of Ad [7]. In addition, choosing good features to represent Ad impressions and modeling Ad clicks with Multiple Criteria Linear Programming Regression (MCLPR) prediction model, has demonstrated better accuracy than Support Vector Regression (SVR) and Logistic Regression (LR) [8].

While predictive based CTR modeling has been commonly studied, they have three major disadvantage preventing them from being commonly deployed in generic commercial bidding engines: (1) the prediction of the predictive modeling has rather poor transparency to understand the CTR and different Ad factors; (2) training predictive models are computationally expensive, and difficult for real-time usages; and (3) predictive modeling cannot be easily adapted to real-world biding system which has billions of transitions on a daily basis, and CTR estimation needs to quickly adapt to any changes in the bid data streams.

In comparison, generative modeling based CTR estimation uses tree hierarchy or base models [5], which have transparent interpretation for users to understand how different Ad factors are related to the estimated CTR values. The best of all, updating and query CTR values from such models are super fast, because we can directly apply new data to update parameters of existing trees for quick CTR estimation and change adaption.

An example of a CTR tree hierarchy commonly used in the industry is shown on the left panel of Fig. 2. For a new impression, this CTR tree will first try to estimate CTR at the most specific tier, and move to less specific tier if it doesn't have enough data (more details are explained in the following sections).

Indeed, while the definition of the CTR value is simple, the number of clicks divided by the number of impressions ($Clicks/Impressions$), building a well structured generative models with multiple-tier structure arranged from specific to general for flexible, accurate, and fast CTR estimation is difficult. This is mainly because that CTR values are impacted by thousands of factors such as the content of the web pages, the reputation of the sites, and publishers etc.

In this paper, we propose a refined hierarchical multiple-tier structure, ULTR-CTR, for flexible, rapid, and accurate CTR estimation. The inherent advantage of ULTR-CTR, compared to existing generative and predictive based CTR estimation approaches is threefold:

- **Flexible and Transparent:** ULTR-CTR provides flexible CTR estimation at different granularity levels. If there are sufficient data for a statistically reliable estimation, it will generate CTR estimation at the finest page-creative-placement-device level, otherwise, it will traverse the CTR tree upward until a statistically reliable estimation can be achieved. The estimated CTR values are full transparent, because the underlying CTR tree is completely transparent.
- **Real-Time CTR Estimation:** The CTR estimation for ULTR-CTR is to traverse the CTR tree for estimation. The worst runtime complexity for each estimation is bounded by the height of the CTR tree, which is $\mathcal{O}(h)$ where h is the height of the CTR tree. This complexity is asymptotically equivalent to $\mathcal{O}(1)$. Our experiments on commercial bidding engine confirms that ULTR-CTR can support sub-millisecond level CTR estimation.
- **Accurate CTR Estimation:** Experiments, carried out on a commercial bidding engine, confirm that ULTR-CTR achieves 2.0% performance gain in CTR estimation, and 1.4% lift in Gross Profit (GP) gain, compared to existing algorithms.

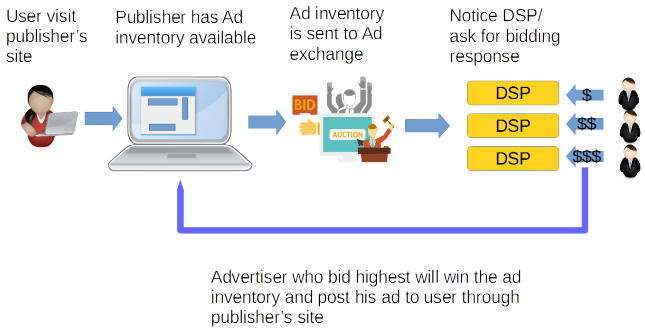

Fig. 1: A conceptual view of the real-time bidding system. From left to right, a user/audience visits a publisher's webpage which has Ad banners. The publisher sends audience information to an AdExchage. The AdExchange sends Ad inventory, as a bid request, to advertisers which use Demanding Side Platform (DSP) to manage their bids. The advertises place bids as bid response. The bidder with the highest bidding price wins the bid, and receives winning notice from the AdExchage. The winning bidder sends Ad scripts to the publisher, so their Ad is displayed on the audience's device.

II. Multiple-tier Hierarchy Structure for Fast CTR Estimation

In order to support real-time CTR estimation, we employ a multiple-tie hierarchy structure, which is essentially a tree structure commonly used in the Ad industry. In the following, we first briefly describe the CTR tree construction, followed by the proposed ULTR-CTR, which carries out fast page grouping using URL truncation for CTR estimation

A. CTR Tree Hierarchy Construction

Formally, assume x defines an object of interest, such as a page, a placement, a site *etc.*, Click Through Rate (CTR) of x denotes the probability of observing a click event from x, assume an Ad is displayed on x. Therefore, a naive CTR estimation can be obtained by using historical data statistics as follows:

$$CTR(x) = \frac{\#\ of\ Clicks\ from\ x}{\#\ of\ Impressions\ from\ x} \quad (1)$$

Despite of its simplicity, the above formula usually gives very accurate estimation when the value of the denominator is sufficiently large (*e.g.* more than 5,000). However, the

limitation of Eq. (1) is that the estimated CTR value is statistically unreliable if there are insufficient number of observed impressions. For example, if we observed one click event out of 5 impressions, its CTR value will be $1/5 = 0.2$ which is abnormally high. The main challenge of the CTR estimation, in addition to the real-time estimation requirement, is twofold:

- **Sparsity of Impressions:** Most sites or pages do not have sufficient number of impressions (*e.g.* more than 5,000 impressions) for obtaining statistically reliable estimation.
- **Revenue Loss Due to Missing CTR Estimation:** Although we can always wait for a sufficient number of impressions to be obtained before obtaining CTR estimation using Eq. (1), this will inevitably incur revenue loss because CTR estimation is the base of determining the bidding price. A delayed CTR estimation will result in inaccurate CTR value and subsequent revenue loss.

For example, in our experiments, we use sampled impressions from the industry partner's 7 consecutive days, and use impressions for the first 5 consecutive days for estimation and the last two consecutive day for validation. There are 244,887,000 impression records in total. Impression records with identical feature values are treated as a unique impression. There are totally 39,540,000 unique impressions occurred in first 5 days. Therefore, on average, there are less than 6 records per unique impression. This number is much lower than industry commonly employed threshold (5,000). In addition, even if using a relaxed threshold (1,000), there are only 10,511 unique impressions with more than 1,000 records in the first 5 days. In other words, only 0.025 percent of unique impressions in the first 5 days would have valid CTR value through Naive estimation. If we use last two days impressions to verify, 20.3% of Ad inventories in the last two days (14,594,000 of total 71,767,000) can be estimated. Therefore, we cannot estimate CTR using naive estimation for vast majority (around 80%) of Ad inventories because of lack of history data.

In order to tackle the above challenges, we propose to build a CTR tree hierarchy, by using a number of selected features closely related to CTR values. In the following, we select eight most relevant features,

$$PageURL, CreativeID, CampaignID, PlacementID, DeviceID,$$

$$Site, Publisher, ParentPublisher$$

. Among the eight features, a $Creative$ belongs to a $Campaign$. A $Page$ belongs to a $Site$, A $Site$ belongs to a $Publisher$, and a $Publisher$ belongs to a $ParentPublisher$. As a result, we can use all eight features, including their combination, to build a CTR tree as shown in the left panel of Figure 2.

The hierarchical tree structure in the left panel of Figure 2 has multiple tiers, and each tier corresponds to a granularity level. The first tier, also referred to as the lowest tier, represents the most specific features combination which includes the four independent features, including $PageURL, CreativeID, PlacementID, DeviceTypeID$.

For each impression, we will first try to find whether the impression can match to any record represented at the lowest tier: $\{PageURL, CreativeID, PlacementID, DeviceTypeID\}$. If the impression matches to a record at this tier, we will increase the number of impressions of the record by one, otherwise, we will drop one feature and move to the next tier:

$$CreativeID, PlacementID, DeviceTypeID$$

. The intuition is that if an impression cannot match a record at the lowest tier, we can remove a feature and relax the matching criteria in order to match a record. Essentially, removing features is equivalent to loosening the criterion of different historical data records matching to an unique impression.

In Figure 3, we demonstrate the conceptual review of removing feature to aggregate impressions for reliable CTR estimation.

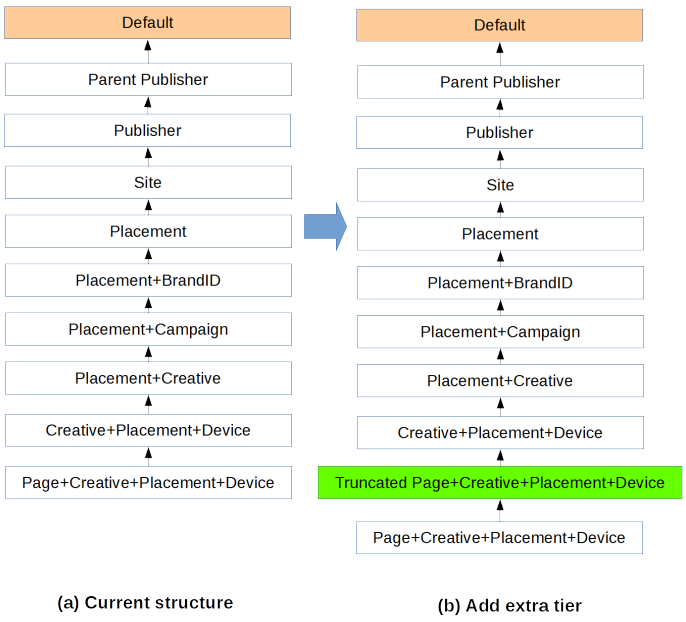

Fig. 2: An example of CTR tree hierarchy (left panel) *vs.* the proposed ULTR-CTR tree hierarchy (right panel). A URL-truncation layer is added to allow similar pages to be quickly grouped based on folder information for CTR estimation.

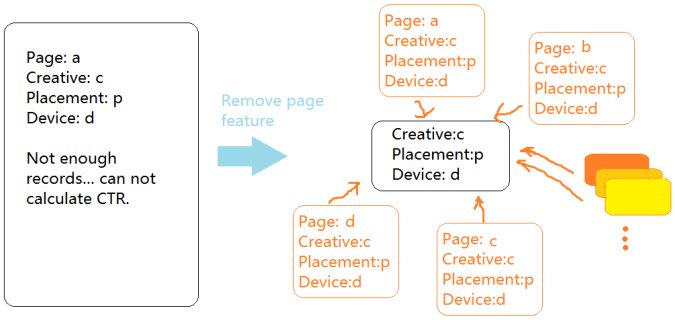

Fig. 3: A conceptual view of page removal in order to obtain reliable CTR estimation. The page on the left panel has page-creative-placement-device information, but insufficient records to calculate CTR value. The panel on the right drops page information to aggregate creative-placement-device information, resulting in reliable CTR estimation.

By following the tree structure shown on the left panel of Figure 2, We can keep removing or replacing features in next tier and make the feature combination more general. This method is a compromise that sacrifices the specificity (*i.e.* accuracy) in order to enable the CTR naive estimation with insufficient number of impressions.

During a CTR estimation process, the CTR estimation engine will always try to find records from the most specific tier, *i.e.* the lowest tier, which is supposed to provide most accurate CTR estimation.

III. ULTR-CTR: URL Truncation for CTR Estimation

The multiple-tier CTR tree, in the previous subsection, solves the problem caused by the sparsity of impressions, at the cost of decrements of CTR estimation accuracy. Whenever, $PageURL$ is removed from the first tier and moving to next tier, the CTR estimation accuracy will drop, this is mainly because that page URL is a very important feature for CTR estimation. Ad banners placed on various pages may have different click through rates, depending on the content of the pages, the style of the page, the publisher of the page (whether pages belong to famous, authoritative and well-designed site), *etc.* Removing page URL will likely incur significant information loss and deteriorate CTR estimation accuracy. Alternatively, if we can keep the page URL information and also overcome the sparsity of impressions, it will help achieve more accurate CTR estimation.

In order to solve the sparsity of impressions and keep the page information, we can merge pages with similar content information into groups, and include such group information in the CTR hierarchy. Intuitively, this problem can be solved using two approaches: (1) using content information of the pages to group pages as clusters, or (2) using page categorization information provide by OpenRTB to group pages with the same categorization as clusters. Unfortunately, our research found that neither of the above two solutions work in commercial bidding systems.

More specifically, OpenRTB [13], the industry standard API specification, requires that each bid request to include specific category information about pages in the json file. The OpenRTB specification defines 26 main page categories, *i.e.* news, sports, arts, economy, *etc.*, and various number of sub categories. While some Ad requests do contain such category information, we found categories extracted from Ad requests are practically uninformative, mainly because that (1) 10% of bid requests contain page category information, whereas majority bid request use same page category across all pages of the site; and (2) the page category information is inaccurate, and include nearly all categories, because publishers intend to use a broader range of keywords to target as many Ads as possible. On the other hand, one can also cluster pages by using their content [14] or user's behaviours [15]. Intensive study about search engine focus on this topic. But their aim is to find similar pages base on user's search words. We don't know whether these clustering techniques is suitable for our case. For example, two pages with similar focus or interest may have completely different layouts and appearance, CTR of ad banners in such sites may vary a lot. Moreover, clustering pages needs intensive calculation and data storage which is impractical for real-time CTR estimation.

A. URL Truncation for Fast Page Grouping

Motivated by the above observations, we propose a new way to group pages by truncating page URL. Page URLs are naturally organized in hierarchies, different subcategories in the same main category are usually placed into one folder. For example:

http://www.foxbusiness.com/markets/2017/05/05/asian-development-lender-takes-stock-as-us-policy-shifts.html vs. http://www.foxbusiness.com/markets/2017/05/05/state-farm-to-close-tulsa-operations-center-in-2019.html

The above two URLs are under the same folder "markets", and according to the new Web 2.0 specification, they are supposed to have the same category and similar content. In Figure 4, we visually demonstrate the two pages under the same folder, which confirm that pages under the same folder usually have similar page layouts and close appearance. As a result, we expect that such similarity in both content and appearance will deliver a similar CTR value.

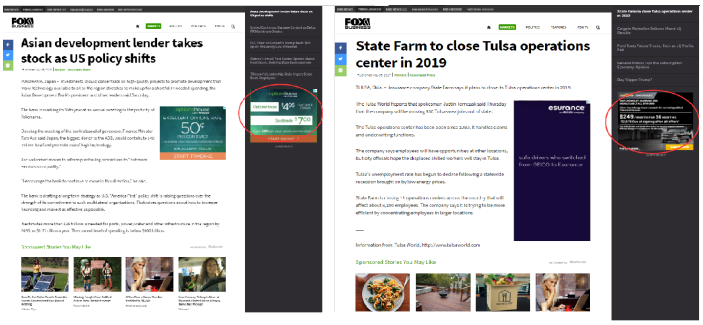

Fig. 4: Example of pages under the same folder—markets. The page on the left panel and the page on the right panel are from the same website, and are under the same folder – markets. Both pages have very similar page style and nearly identical banner locations, which are red-circled in each page. The content of the pages are different but are closely related to one theme – markets.

Based on the above study, we can use page URL truncation as a new type of page clustering approach to quickly merge similar pages into groups. A page truncation is to remove detailed information and only keep domain and folder information in each URL. Comparing to the complete removal of the page URL, URL truncation can keep part of page information for accurate CTR estimation. In addition, URL truncation can be solved with $\mathcal{O}(1)$ cost, so it can be implemented in real-time, whereas other approaches, such as page content clustering will at least requires $\mathcal{O}n$ complexity which cannot be implemented in reality for commercial usages.

The right panel in Figure 2 demonstrates the proposed URL truncation based CTR tree structure, where a new tier $\{$*Truncated page, Creative, Placment, Device*$\}$ is added to avoid complete page information removal for accurate CTR estimation.

For example, the URL shown in the previous example, can be truncated as follows:

Original URL: *http://www.foxbusiness.com/markets/2017/05/05/asian-development-lender-takes-stock-as-us-policy-shifts.html*

Site Only Truncation (Site0F): *http://www.foxbusiness.com/*

Site + 1 Folder Truncation (Site1F): *http://www.foxbusiness.com/markets/*

Site + 2 Folder Truncation (Site2F): *http://www.foxbusiness.com/markets/2017/*

Site + 3 Folder Truncation (Site3F): *http://www.foxbusiness.com/markets/2017/05/*

If we remove all folder information in the URL, only the site information will remain. On the other hand, as page truncation including more folders, it will become more specific, but it will be less effective to tackle the impression sparsity challenge. In the next subsection, we will use real-world experiments to validate that using " Site + 1 folder" truncation (Site1F) will result in best performance for CTR estimation.

B. Site1F URL Truncation

In order to determine the optimal number of folders to be used in the URL truncation, we carry out experiments on randomly sampled impressions collected from 14 consecutive days to compare different URL truncation approaches (*i.e.*, using site0F, site1F, site2F, *etc.*, respectively).

In order to check which URL truncation approaches, site0F, site1F, or site2F can result in most accurate CTR estimation, we build five trees as shown in Table I. More specifically, "Baseline" denotes the existing tree currently employed by the industry partner which has the same structure as shown on the left panel of Figure 2. The lowest tier of the "Baseline" tree include Page (P), Creative (C), Placement (T), and Device (D) information. If an impression cannot match to a record at this tier, it will drop page information and move to Tier 3, which only consider creative, placement, and device for CTR Estimation. "Site0F", "Site1F", and "Site2F" denote the proposed page truncation based CTR trees, as shown on the right panel of Figure 2, by using site only, site + 1 folder, and site + 2 folders, respectively. The last tree, "DropPage", does not consider any page information, and its CTR estimation only considers creative, placement, and device. We employ "DropPage" tree in our experiments, because the industry partner has confirmed that "Baseline" performs better than "DropPage" in terms of Key Performance Indicator (KPI), including CTR estimation and Gross Profit (GP) gain. Therefore, if any of the proposed trees, Site0F, Site1F, or Site2F, does not show better performance than "DropPage", it means that these trees cannot outperform "Baseline".

To test the performance of each tree on the impressions collected from 14 consecutive days, we employ a sliding window based loop test as shown in Figure 5. More specifically, we use impressions from 7 consecutive days to build CTR trees, including "Baseline", "Site0F", "Site1F", "Site2F", and "DropPage", and validate each tree's performance on impressions from the next immediate day. The sliding windows move one day in each loop, and will produce 7 loop test results for each tree.

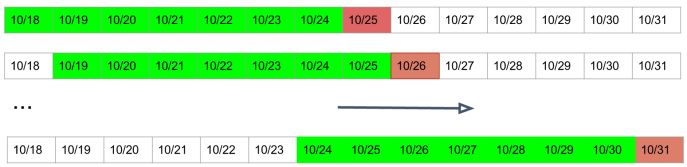

Fig. 5: Sliding window test to compare the performance of different CTR trees. Each rectangle box denotes impressions collected from one day (the value inside the box denotes the index of data chunk). Impressions in green-colored boxes are used to build CTR trees, and impressions in red-colored boxes are used to test tree performance. In other words, we use impressions from 7 consecutive days to build CTR trees, and validate the performance on the next immediate day. The sliding windows move one day in each loop.

To compare performance of each tree, we use Mean square error (MSE) in Eq. (2) to evaluate CTR estimation, where $\hat{Y}$ and Y are estimated CTR value from each tree, and true CTR value, respectively.

TABLE I: Loop test setting for determining the optimal number of folders to be used in the URL truncation. Each column denotes one specific CTR tree. Tier 1 is the most specific layer which includes Page (P), Creative (C), Placement (T), and Device (D). Tier 2 uses page truncation to group pages, and tier 3 completely drops page information. "Baseline" denotes current CTR tree used by the industry partner as shown on the left panel of Figure 2. "Site0F", "Site1F", and "Site2F" denote the proposed page truncation based CTR trees, as shown on the right panel of Figure 2, by using site only, site + 1 folder, and site + 2 folders, respectively. "DropPage" denotes a CTR tree completely dropping page information and only consider creative, placement, and device.

	Baseline	Site2F	Site1F	Site0F	DropPage
Tier1	P+C+T+D	P+C+T+D	P+C+T+D	P+C+T+D	N/A
Tier2	N/A	Site2F+C+T+D	Site1F+C+T+D	Site0F+C+T+D	N/A
Tier3	C+T+D	C+T+D	C+T+D	C+T+D	C+T+D

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{2}$$

In summary, the sliding window based loop test procedures are summarized as follows:

- Collect impressions from seven consecutive days to build a CTR tree (either "Baseline", "Site0F", "Site1F", "Site2F", or "DropPage").
- For each impression in the next immediate day, use the constructed tree to estimate CTR of the impression, which is the CTR estimation, $\hat{Y}$, as defined in Eq. (2).
- Calculate the true CTR of the impression—count the numbers of this impression records and the clicks produced by this impression in all 14 days, using $clicks/impressions$. We use expected CTR values (from 14 days) as the true CTR value, this is mainly because that the true CTR value of the impression is unknown (because we can only observe a portion of impressions), so we can only use sample means to replace the true mean.
- Calculate Mean Squared Errors (MSE) of the tree using Eq. (2), with respect to all impressions in the next immediate day, and repeat the above process for each tree.
- Move the sliding window one day and repeat the above process, until the sliding windows walk through all 14 days.

Table II reports the MSE of all trees with respect to each loop of the test. Each column of Table II denotes the performance of one tree, and each row denotes one loop test of the sliding window. For each loop test, the result of the best performed tree is bold-faced, whereas the result of the worst tree is italic-faced. Overall, the test confirms that CTR trees dropping page information, "DropPage", have the worst performance. This is consistent with the results observed by the industry partner, whose team confirmed that CTR tree completely ignoring page information will lead to inaccurate

TABLE II: Mean squared error (MSE) comparisons of different trees using sliding window loop test results of MSE. Each row denotes a loop test of the sliding window (*i.e.* using impressions from 7 consecutive days to build CTR tree and validate on impressions from the next immediate day). Each column denotes MSE of the corresponding tree. Site1F obtains the best CTR estimation in 5 out of 7 tests, which means that truncating page URL as $site+1folder$ is most likely to result in the best CTR estimation.

	Baseline	Site2F	Site1F	Site0F	DropPage
Loop1	2.219E-6	2.213E-6	2.254E-6	**2.209E-6**	*2.529E-6*
Loop2	**2.994E-6**	2.994E-6	3.017E-6	2.996E-6	*3.128E-6*
Loop3	4.242E-6	4.289E-6	**4.126E-6**	4.240E-6	*4.764E-6*
Loop4	5.242E-6	5.240E-6	**5.110E-6**	5.237E-6	*5.525E-6*
Loop5	2.869E-6	2.848E-6	**2.675E-6**	2.866E-6	*5.041E-6*
Loop6	3.559E-6	3.530E-6	**3,162E-6**	3.552E-6	*4.230E-6*
Loop7	2.905E-6	2.900E-6	**2.768E-6**	2.905E-6	*4.402E-6*

CTR estimation. Therefore, we can safely trust that MSE is a good measure to evaluate the CTR tree performance.

In addition, the results in Table II confirm that page truncation using site and one folder information (Site1F) has better performance (winning five out of seven loop tests). While the value of improvement in Table II is rather marginal, we should note that it's very difficult to achieve substantial performance gain, compared to the existing commercial products which have been continuously optimized for years. On the other hand, even a 1% performance gain will result in thousands of dollar revenue, considered the transaction volumes and the market size.

In summary, our test confirms that using Site1F page truncation results best performance for CTR estimation. In the following, we will report experiments validated on the commercial systems to demonstrate improvement of the proposed design in terms of improved CTR estimation and gross profit (GP) gain.

IV. Experiments and Online A/B Test Results

In the previous section, we have shown that the new hierarchical structure, including the truncated page tier, results in better CTR estimation. However, the above tests are carried out offline by using data collected from historical data. As commercial bidding engines always operate in real-time, we will carry out online A/B test to validate whether the proposed tree can indeed help achieve better performance, in terms of different key performance indicators.

The KPI is a metric to evaluate the performance of realizing company's key objectives. Pursuing better KPIs is the marketing objective of majority, if not all, companies. One typical KPI is Gross Profit(*a.k.a* Gross margin) as defined in Eq. (3), which is the difference between revenue and costs.

$$GrossProfit = Revenue - Costs\ of\ purchasing\ impressions \quad (3)$$

A. Experimental Settings

In practice, it is impractical to estimate CTRs using all historical data since the time spent executing CTR estimation will be overwhelming with the increasing historical data. Therefore, we use sliding windows to obtain the latest historical data and furthermore in case that there are insufficient historic data, we dynamically set several windows. The upper boundary of dynamic window should not be too high or it will diminish the trait of "real-time".

Because the whole bidding process has to be finished in milliseconds, CTR estimation should be obtained instantly. First, we have to calculate the CTR for all unique impressions and cache them in advance so that the CTR can be "looked up" immediately when a new Ad request arrives. We need a data aggregator and a data cache. The impression records are saved by entry, each entry records all the information about this impression:

CreativeID , PlacementID, DeviceID, PageURL, Impressions Number(1) , Clicks (Either 1 or 0), etc.

Date aggregator is used to count the total impression number and click number for a unique impression. Figure 6 shows how it works, each SQL execution gives CTR estimation for all unique impressions, one group by clauses determine a set of features which represents a certain tier.

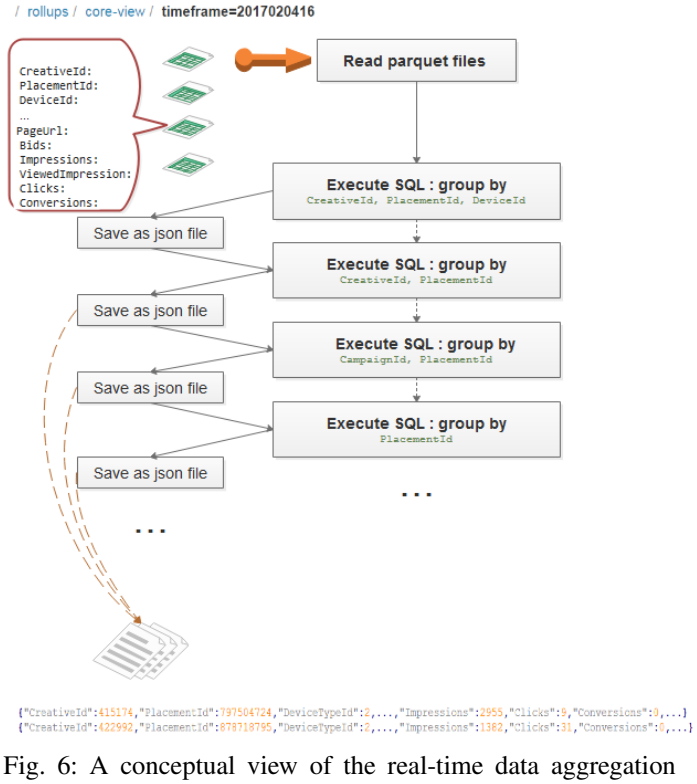

Fig. 6: A conceptual view of the real-time data aggregation for CTR estimation

B. A/B Test Settings

We deployed our CTR estimation method on the Bidtellect DSP and performed online A/B testing for 4 weeks. As for dynamic sliding windows, we chose three intervals, *i.e.* 24 hours, 7 days, and 2 weeks. If insufficient number of impressions can be found in the "24 hours" window, "7 days" window will be used, and so on.

During A/B test, Ad requests traffic are randomly split into two halves, where one half of traffic uses industry partner's existing structure to estimate CTR and the other half uses our method (ULTR-CTR). In regard to the traffic splitting method, we split all Ad inventories because one Ad request may contain multiple placements, each of which is an Ad inventory. We used a random value to select CTR estimation methods. Every time a new Ad request arrives, we generated random values ranging from 0 to 1 for its placements. The traffic flows to the current CTR estimation approach if the value of this placement is under 0.5, otherwise our CTR estimation approach. Once a CTR estimation approach was chosen, we would mark this impression (Ad inventory) with different CTR estimation approach ID for tracking. After one weeks observation, we got two statistics about KPI using different CTR estimation approach respectively. Figure 7 lists the major steps of A/B test.

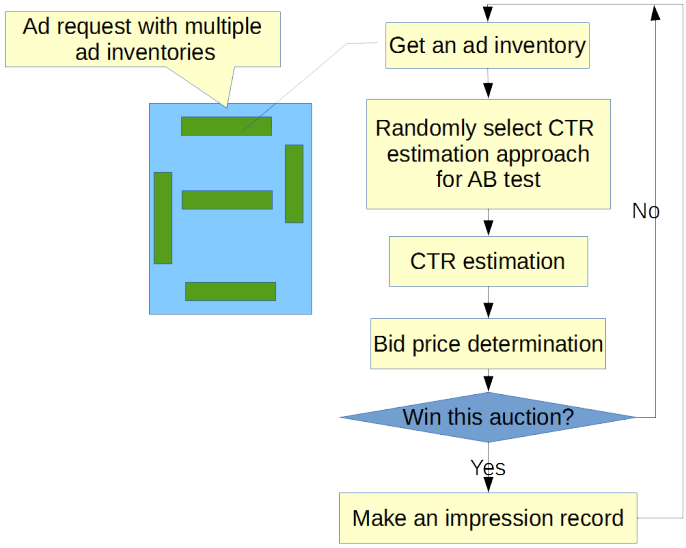

Fig. 7: A conceptual view of the online A/B test. For each Ad request, the DSP will randomly select a CTR estimation approach to estimate the CTR value and determine bidding price. As a result, after running the A/B test for a certain period (we used four consecutive weeks), we can fairly compare the performance of different CTR trees, by using CTR values or gross profit resulted from each CTR approach.

C. KPI Performance Comparisons

Table III reports the KPI values obtained from four consecutive weeks, including performance with respect to each week and across all four weeks. Baseline indicates the CTR estimation approach currently employed by the industry partner, and ULTR-CTR is the proposed approach which uses page URL truncation (using site plus one folder) to build CTR tree for estimation. For confidentiality reasons, GP is normalized to $10,000 for the Baseline, and the relative gain will demonstrate the performance of the proposed method.

The results of A/B testing are very consistent in 4 weeks. While the proposed method, ULTR-CTR, won less number of impressions in each of the 4 weeks, but it increased the number of clicks in 3 out of 4 weeks. That means that ULTR-CTR can effectively decreases effective CPC (eCPC), *i.e.* reduced cost for each click event.

TABLE III: A/B test results within 4 consecutive weeks. Gross Profit (GP) is normalized to $10,000 to prevent leakage of sensitive information. ULTR-CTR is compared with the baseline approach, and resulted in 2.0% improvement in CTR estimation, and 1.4% lift in GP.

Approch	Impressions	Clicks	CTR	GP($)	Date
Baseline	86,871,598	612,381	0.007049	10,000	Week 1
ULTR-CTR	85,388,698	614,299	0.007194	10,247	01/16-01/22
Improvement	-	-	2.1%	2.5%	
Baseline	94,105,419	738,744	0.007850	10,000	Week 2
ULTR-CTR	92,254,755	740,846	0.008030	10,142	01/23-01/29
Improvement	-	-	2.3%	1.4%	
Baseline	80,492,102	607,185	0.007543	10,000	Week 3
ULTR-CTR	78,228,341	607,106	0.007761	10,048	01/30-02/05
Improvement	-	-	2.9%	0.5%	
Baseline	63,269,740	390,792	0.006177	10,000	Week 4
ULTR-CTR	62,587,576	607,106	0.006293	10,098	02/06-02/12
Improvement	-	-	1.9%	1.0%	
Baseline	324,738,859	2,349,102	0.007234	10,000	Weeks 1-4
ULTR-CTR	318,459,370	23,56,134	0.007376	10,138	01/16-02/12
Improvement	-	-	2.0%	1.4%	

Overall, the CTR values of the ULTR-CTR consistently increase in all 4 weeks, and it also results in a higher Gross Profit in all 4 weeks. In total, the number of impressions we bought decreased by 1.9 percent, the number of clicks increased by 0.3 percent, CTR increased by 2 percent and Gross Profit increased by 1.4 percent. Such consistent enhancement and performance gain proved that using page truncating to cluster page URLs will provide better CTR estimations as well as better KPI in practice.

V. Conclusion

CTR estimation is one of the most important steps in Real-time Bidding for computational advertising. While most research in CTR estimation utilize well-studied machine learning models, industry commonly relies on generative models, such as simple feature based tree hierarchy, mainly because CTR trees are easy to implement, easy to interpret, and easy to update by using big data processing and storing tool, in real-time.

For existing CTR tree based approaches, a page URL is directly removed when there are insufficient number of impressions at the page level for CTR estimation. As a result, it will lead to a complete loss of information about site and pages, and deteriorates CTR estimation accuracy. To address this issue, we proposed a new method, ULTR-CTR, to partially keep the page URL information, by truncating the page URL and making it more general. Our main objective is to keep partial page information and increase the number of impressions for page level CTR estimation. To achieve the goal, we added an extra tier into the current CTR tree hierarchical structure. If CTR estimation at a page level does not have sufficient number of impressions for a statistical reliable estimation, ULTR-CTR will truncate the page URLs and only preserves site and folder information for CTR estimation. Such a truncation procedure will increase the number of impressions for CTR estimation

using partially preserved page information. Our hypothesis is that pages under the same domain and folders have similar CTR values, so we can use URL truncation to quickly cluster pages into groups for reliable CTR estimation. To verify our hypothesis, we designed a loop test on 14 day historical data, and the offline test confirmed that our proposal resulted in a more accurate CTR estimation.

After that, we carried out online A/B test and validated the performance of ULTR-CTR on a commercial bidding engine for four consecutive weeks. The A/B test confirmed that ULTR-CTR resulted in 2% performance gain in CTR estimation, and 1.4% gain in gross profit. Overall, ULTR-CTR delivers a light-weight page clustering approach for real-time accurate CTR estimation.

REFERENCES

[1] D. W. Roth and D. Salisbury, "Internet advertising system," *US Patent*, vol. US 6285987 B1, 2001.

[2] X. Zhu, H. Tao, Z. Wu, J. Cao, K. Kalish, and J. Kayne, *Fraud Prevention in Online Digital Advertising*. SpringerBriefs in Computer Science, 2017.

[3] D. C. Fain and J. O. Pedersen, "Sponsored search: a brief history," *Bulletin of the American Society for Information Science and Technology*, vol. 32, pp. 12–13, 2006.

[4] B. Edelman, M. Ostrovsky, and M. Schwarz, "Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords," *American Economic Review*, vol. 97, pp. 242–259, 2007.

[5] R. Ormandi, H. Yang, and Q. Lu, "Scalable multidimensional hierarchical bayesian modeling on spark," *Proceedings of the 4th International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, no. 41, pp. 33–48, 2015.

[6] C. Li, Y. Lu, D. Wang, and S. Pande, "Click-through prediction for advertising in twitter timeline," *Proc. of ACM SIG KDD*, 2015.

[7] K. Dembczynski, W. Kotowski, and D. Weiss, "Predicting ads click-through rate with decision rules," *WWW 2008*, April 2008.

[8] F. Wang, W. Suphamitmongkol, and B. Wang, "Advertisement click-through rate prediction using multiple criteria linear programming regression model," *Elsevier Procedia Computer Science*, no. 17, 2013.

[9] W.-Y. S. W.-C. P. Wen-Yuan Zhu, Chun-Hao Wang and J.-L. Huang, "Sem: A softmax-based ensemble model for ctr estimation in real-time bidding advertising," *2017 IEEE International Conference on Big Data and Smart Computing*, pp. 5–12, 2017.

[10] S. W. L. W. Qiang Liu, Feng Yu, "A convolutional click prediction model," *CIKM '15 Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1743–1746, 2015.

[11] M. Richardson, E. Dominowska, and R. Ragno, "Predicting clicks: Estimating the click-through rate for new ads," *WWW 2007*, May 2007.

[12] M. C. Andrzej Szwabe, Pawel Misiorek, "Logistic regression setup for rtb ctr estimation," *ICMLC 2017 Proceedings of the 9th International Conference on Machine Learning and Computing*, pp. 61–70, 2017.

[13] IAB, "Openrtb api specification version 2.5," *https://www.iab.com/guidelines/real-time-bidding-rtb-project/*, 2016.

[14] T. Kohonen, S. Kaski, K. Lagus, J. Salo, J. Honkela, V. Paatero, and A. Saarela, "Self organization of a massive document collection," *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 11, no. 3, May 2000.

[15] K. Smith and A. Ng, "Web page clustering using a self-organizing map of user navigation patterns," *Elsevier Science*, 2002.

[16] A. C. P. and V. M. S., "Click through rate prediction for display advertisement," *International Journal of Computer Applications*, vol. 136, no. 1, 2007.