Pacing Programmatic Job Campaigns by Score-based Ranking

HAO SHA, YIYUN ZHOU, MADHAV SIGDEL, MENGSHU LIU, and MOHAMMED KORAYEM, CareerBuilder, USA

Sponsored job recommendation has become a significant source of revenue for online job marketplaces, where recruiters can set up campaigns with specified budgets to promote their jobs. Campaigns running simultaneously have to compete for the limited number of impressions, and thus high performing campaigns tend to spend out their budgets fast, while underperforming campaigns have difficulty spending their budgets. It is thus desired for the platforms to implement specific pacing mechanisms to coordinate the spending speed across campaigns. In addition, the recruiters often prefer that all the jobs within a campaign can have applicants. Thus, the pacing mechanism should also be able to control the conversions across jobs within a campaign. In this work, we propose a pacing-by-score model that meets both requirements. The experimental evaluations demonstrate the effectiveness of our approach under various circumstances.

CCS Concepts: • **Theory of computation → Computational advertising theory**; • **Information systems** → *Rank aggregation.*

Additional Key Words and Phrases: pacing, programmatic job campaigns, recommender system, online job marketplaces

ACM Reference Format:
Hao Sha, Yiyun Zhou, Madhav Sigdel, Mengshu Liu, and Mohammed Korayem. 2022. Pacing Programmatic Job Campaigns by Score-based Ranking. In *CompJobs-WSDM '22: The First International Workshop on Computational Jobs Marketplace, Feb 25, 2022, Phoenix, AZ*. ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Online recruitment platforms aim to match suitable candidates to the right jobs by showing the most relevant job ads. Recently, there has been growing interest in programmatic job advertising, where job posters pay based upon the performance of the job ad, such as the number of job clicks and applications. Job posters can also promote jobs via campaigns to maximize the exposure of their jobs for higher yield. Generally, campaigns have a fixed budget and an expected running period. The objective of a job campaign would be to get enough job applications for every job at the least cost. Given that multiple campaigns and jobs within a campaign compete against each other, we need to devise a smart pacing mechanism to utilize the campaign budget efficiently.

Various empirical studies have been conducted to solve the pacing problem. One popular line of approaches are based on probabilistic throttling [1, 3, 7]. [3] utilized projected request rate and winning rate to estimate the throttling probability. [1] adjusted throttling probability in an adaptive manner, depending on the relationship between actual and planned spends. [7] associated probabilities with hierarchical groups by conversion rate. Pacing models in [1, 3, 7] also include mechanisms for forecasting impression volume and quality for smart budget allocation. Another line of works resort to bid modification [4, 5]. [4] proposed a tradeoff function to adjust bid based on the fraction of spend and budget. [5] unified response prediction, cost forecasting, and bid optimization. A recent work [2] proposed an impression-based pacing scheme that takes into account the order of recommended jobs within a page.

The pacing approach has become standard for all platforms supporting programmatic transactions, and the sponsored job will be charged based on the job seeker's relevance and remaining budget. However, the traditional pacing algorithms pay little attention to the low-interested sponsor job applications in the campaign. To fill the gap, our campaign pacing engine is designed to maximize platform revenue which more likely promotes the highest bidder, and use the trade-off function to ensure that the low-interested jobs also have enough exposure to candidates at the same time. In section 2, we describe the problem and model details. In section 3, we demonstrate the effectiveness of our model through simulation experiments.

2 METHODOLOGIES

2.1 Problem Definition

Campaign pacing refers to spreading out the spend of the campaign evenly across the term of the campaign (1st requirement). The reason for doing this is to spend budget smoothly in order to reach a broader range of audiences and have a sustainable impact. In addition, the campaign pacing algorithm should function to control distribution of conversions across jobs to ensure that all jobs within that campaign have applicants (2nd requirement). This is important because recruiters want conversions across all their jobs, so a key measure of success for them is to have applicants for as many jobs as possible.

Let us consider the following settings: A campaign i has bid_i, budget B_i, and N_i jobs. The campaign term is discretized into T days. Let $b_{i,t}$ denote the daily budget allocated to day t, with $\sum_{t=0}^{T} b_{i,t} = B_i$. Let $s_{i,t}(\tau)$ denote the cumulative spend from the beginning of the day till time τ of day t, and $s_{i,t}$ denote the daily spend of day t. To meet the 1st requirement, we want $|s_{i,t} - b_{i,t}| \rightarrow 0$ and $|b_{i,t} - B_i/T| \rightarrow 0$. Let $a_{j,t}(\tau)$ denote the number of applies of job j till time $t + \tau$ and $a_j = a_{j,T}$ denote the total number of applies of job j during its life time. So the 2nd requirement can be formulated as $a_j > 0$ for $0 \leq j < N_i$.

2.2 Model Description

Pacing by score We propose a pacing-by-score model that implements pacing by assigning each campaign job a pacing score $S^p \sim \phi(bid_i) \times \psi(s_{i,t}(\tau)/b_{i,t}) \times \eta(a_{j,t}(\tau))$, where ϕ is monotonically increasing, while ψ and η are monotonically decreasing. We can see from this formula that, on the one hand, jobs with higher bid will be promoted more because of ϕ; on the other hand, jobs that nearly spend out their daily budget and have significantly more applies than the others in the same campaign will be penalized because of ψ and η, respectively. The recommender system will then rank the jobs by a combined score $S = S_0 + C \times S^p$, where S_0 is a base score that measures the matching between a candidate and a job [6]. $C > 0$ is a constant that adjusts the weight of pacing score and can be determined via A/B testing. For non-campaign jobs, we have $S = S_0$. Therefore, campaign jobs will have a boost of $C \times S^p$ over non-campaign jobs, provided that $S^p > 0$.

Specifically, we can set $\phi = bid_i$, $\psi = 1 - exp(-1 + \frac{s_{i,t}(\tau)}{b_{i,t}})$, and $\eta = exp(-a_{j,t}(\tau)/\alpha_i)$, where $\alpha_i = \frac{B_i}{bid_i \times N_i}$ is a campaign specific softness. Note that the form of ψ follows the tradeoff function defined in [4]. Alternatively, we can drop ψ and perform probabilistic throttling [7] before ranking. Formally, for a job j of campaign i, the combined score at time τ of the day t is defined as the following:

$$S_{j,t}(\tau) = S_{0,j} + C \times S^p_{j,t}(\tau) = S_{0,j} + C \times \overbrace{bid_i}^{\phi} \times \overbrace{[1 - exp(-1 + \frac{s_{i,t}(\tau)}{b_{i,t}})]}^{\psi} \times \overbrace{exp(-a_{j,t}(\tau)/\alpha_i)}^{\eta}. \quad (1)$$

Table 1. Campaign Data

camp id	0	1	2	3	4	5	6	7	8	9
budget	500	1000	2000	2000	500	2000	500	2000	500	1000
daily cap	26	43	76	76	26	76	26	76	26	43
bid	0.5	0.3	0.3	0.5	0.4	0.3	0.5	0.5	0.3	0.4
no. jobs	28	4	16	4	12	16	2	12	5	1

The intuition behind Eq. 1 is that (1) the ϕ term makes the system promote a higher bidder more, which helps to maximize platform revenue; (2) the tradeoff term (ψ) slows down spending when the cumulative spend is close to the daily budget (1st requirement); (3) the exponential decay term (η) penalizes jobs with significantly more applies over the other jobs in the same campaign, so that jobs with fewer applies can be promoted (2nd requirement); (4) since $C > 0$, campaign jobs are boosted over non-campaign jobs when $\psi > 0$.

3 EXPERIMENTS

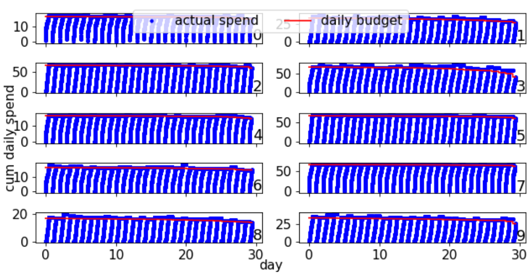
3.1 Poisson Process

In this experiment, we assume that job seekers visit our website following a Poisson process, where the inter-arrival time $\delta t \sim exp(-L\delta t)$ with $L = 1$ (i.e., 1 request per minute). We recommend 5 jobs at once (i.e. 5 impressions) with conversion rate (CVR) decreasing from 0.5 to 0.1. We set the campaign period as 30 days, and assume that all campaigns have a start date at day 0 and an expected end date at day 29. We adopt an adaptive daily budget that is re-calculated at each day t as $b_{i,t} = min(d_i, \frac{B_i - \sum_{t'=0}^{t} s_{i,t'}}{T-t+1})$, where $t \in [0, T]$ and d_i is a daily spend cap provided by recruiters to indicate the maximum number of applications they can process daily. The campaign-related data is presented in Table 1 for 10 campaigns with different budget, daily limit, bids and number of jobs. The pacing engine computes the combined score $S_{j,t}(\tau)$ in Eq. 1 whenever there is a request for impressions. It also monitors whether a campaign has reached its daily budget $b_{i,t}$ every 15 minutes. Once a campaign spends out its daily budget, it will be suspended for the rest of the day.

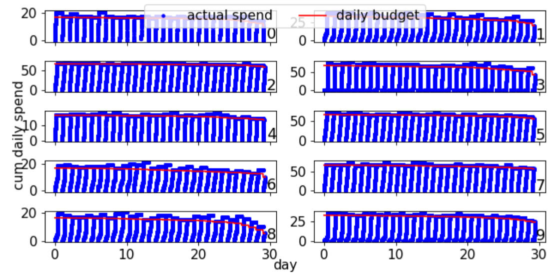
We perform simulations with both $C = 5$ and $C = 0$. We pick $C = 5$ so that base score S_0 and pacing score $C \times S^p$ are on the same scale, although further tuning can be done through A/B testing. We also examine the case where $C = 0$, because from Eq. 1 we can see that this is equivalent to turning off pacing-by-score. Note that although pacing is off, the campaigns will still be suspended for the rest of the day once the daily budget is reached. $C = 0$ can thus be viewed as a baseline with minimum degree of pacing.

In Fig. 1, we plot the daily cumulative spend for the 10 campaigns. From Fig. 1 (a) & (b), we can see that all campaigns last to the end date (day 29) for both $C = 5$ and $C = 0$. This suggests that capping at daily budget alone can prevent campaigns from ending prematurely. We observe that when $C = 0$, the campaigns are more prone to over-spend (blue curves over red curves), especially for campaigns 1, 6, and 8. This is not surprising as the combined score $S_{j,t}(\tau)$ is evaluated every time a search occurs, whereas the daily budget capping is performed every 15 minutes, thus a longer latency.

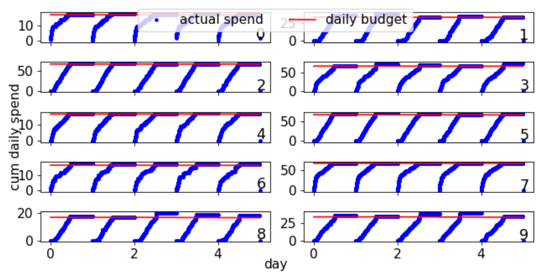
In Fig. 1 (c) & (d), we show a zoom-in version of (a) & (b) (day $0 \sim 4$). For $C = 5$, we can see that the spends of different campaigns grow at a similar pace, and reach the daily budget roughly at the same time (Fig. 1 (c)). In contrast, for $C = 0$, the spending speeds differ significantly across campaigns (Fig. 1 (d)). We can see that strong campaigns (high base score $S_{0,j}$), such as campaigns 0 and 4, spend out their daily budget quickly, whereas weak campaigns (low base score $S_{0,j}$), such as campaigns 3 and 9, start slowly and only receive applies after the strong ones exit the competition.



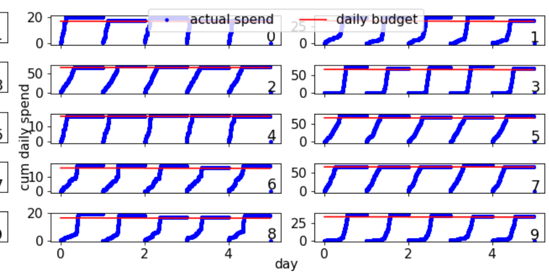
(a) Poisson, day 0 ∼ 29, $C = 5$



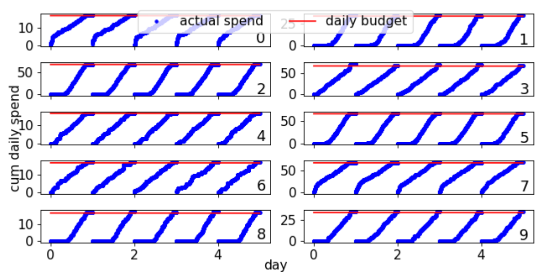
(b) Poisson, day 0 ∼ 29, $C = 0$



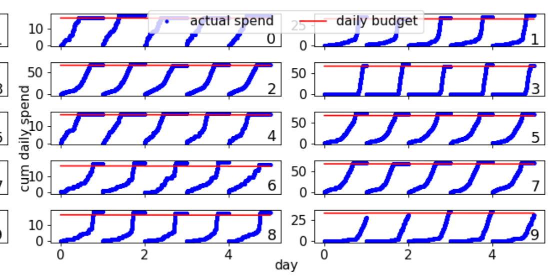
(c) Poisson, day 0 ∼ 4, $C = 5$



(d) Poisson, day 0 ∼ 4, $C = 0$



(e) TOD, day 0 ∼ 4, $C = 5$



(f) TOD, day 0 ∼ 4, $C = 0$

Fig. 1. Daily cumulative spend for campaigns 0 ∼ 9 (blue). The red curves indicate the daily budgets. The ids in the bottom right are campaign ids.

Table 2. Minimum Number of Applies per Job per Campaign

camp id	0	1	2	3	4	5	6	7	8	9
Poisson, $C = 5$	35	631	340	954	94	375	475	317	293	2507
Poisson, $C = 0$	0	235	23	753	0	0	345	0	182	2507
TOD, $C = 5$	35	631	337	966	96	369	480	316	274	2502
TOD, $C = 0$	0	251	25	779	0	0	378	0	177	2501

Therefore, we find that pacing score (S^p), in specific the tradeoff term (ψ), is effective in balancing the spending speed across campaigns and maintaining sustainable competitions.

In Table 2 row 2 & 3, we list the minimum number of applies per job per campaign. For $C = 5$, we can see that all jobs have applies, because the least number of applies per job across campaigns is 35. In contrast, for $C = 0$, the campaigns 0, 4, 5, and 7 have at least one job with zero applies. This shows that pacing score (S^p), specifically the exponential decay term (η), is effective in ensuring that all jobs within a campaign have applies. Note that in this setup we have about $30\ days \times 24\ hrs \times 60\ mins \times 5\ impressions \times 0.3\ avg\ CVR = 64800$ total applies available, which is more than enough for the campaigns (sum of maximum possible applies for all campaigns = 32081), therefore supply > demand. This is one of the reasons why campaigns all over-spend (another reason is latency ~ 15min).

3.2 Time of Day

In this experiment, we adopt the same campaign setup as in the previous experiment, except for assuming that the job search events follow a time-of-day (TOD) pattern with a peak arriving around 3 ~ 4 pm. Compared to the Poisson process model, the TOD approach can better mimic the scenario where traffic is not evenly distributed in a day.

Similar to the Poisson simulations, we test the cases where $C = 5$ (i.e., pacing-by-score is on, and jobs are ranked by $S_0 + C \times S^p$) and $C = 0$ (pacing-by-score is off, but daily budget capping remains). For the TOD model, we set the average number of searches per minute as 4 (avg intensity). We show 5 recommended jobs per search with conversion rate (CVR) decreasing from 0.05 to 0.01. The total number of applies available is roughly 25920 which is less than the total applies the campaigns can afford (32081), thus supply < demand.

From Fig. 1 (e) & (f), we can see that pacing-by-score can make the spending in a more coordinated manner across campaigns. Although slow start can be seen in both cases, it is more severe in $C = 0$, especially in campaigns 3 and 9. Furthermore, in Table 2 row 4 & 5, we can see from the minimum number of applies that when pacing-by-score is on, all jobs have applies, whereas when it is off, jobs can have zero applies in campaigns 0, 4, 5, and 7. This again demonstrates the effectiveness of pacing-by-score in solving the needs in requirements 1 & 2.

4 CONCLUSION

In this work, we propose a pacing-by-score method to pace the spending of programmatic job campaigns. Through simulations, we demonstrate the effectiveness of our method in satisfying the business requirements of spreading the spend and conversions across campaigns and jobs. For future works, we plan to implement our method in production, and evaluate its performance against various state-of-the-art models.

REFERENCES

[1] Deepak Agarwal, Souvik Ghosh, Kai Wei, and Siyu You. 2014. Budget Pacing for Targeted Online Advertisements at LinkedIn. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. 1613–1619.

[2] Sahin Cem Geyik, Luthfur Chowdhury, Florian Raudies, Wen Pu, and Jianqiang Shen. 2020. Impression Pacing for Jobs Marketplace at LinkedIn. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management.* 2445–2452.

[3] Kuang-Chih Lee, Ali Jalali, and Ali Dasdan. 2013. Real time bid optimization with smooth budget delivery in online advertising. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising.* 1–9.

[4] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. 2005. AdWords and generalized on-line matching. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. 264–273.

[5] Kan Ren, Weinan Zhang, Ke Chang, Yifei Rong, Yong Yu, and Jun Wang. 2018. Bidding Machine: Learning to Bid for Directly Optimizing Profits in Display Advertising. *IEEE Transactions on Knowledge and Data Engineering* 30, 4 (2018), 645–659.

[6] Walid Shalaby, BahaaEddin AlAila, Mohammed Korayem, Layla Pournajaf, Khalifeh AlJadda, Shannon Quinn, and Wlodek Zadrozny. 2017. Help me find a job: A graph-based approach for job recommendation at scale. In *2017 IEEE International Conference on Big Data (Big Data).* 1544–1553.

[7] Jian Xu, Kuang-chih Lee, Wentong Li, Hang Qi, and Quan Lu. 2015. Smart Pacing for Effective Online Ad Campaign Optimization. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15).* 2217–2226.