在线广告分配综述

JasonnJiang[1] DevinZheng[2] NireyHuang[3] AppleLin[4]

1. 背景介绍

在广告平台中，竞价广告和展示广告是两种比较重要的广告形式。对于竞价广告，广告主设置多个推广计划，每个推广计划包含多个广告和预算限额。广告主在设定预算下，将广告投放到定向人群，并参与竞价，与此同时，希望预算能匀速地消耗完以获取比较好的推广效果。随着广告业务不断增长，以及广告预算和竞争力的差异，在线广告匹配算法-Greedy 算法，即每次选择 ecpm(千次网页展示收入)最高的广告，会引起相对比较突出的预算控制问题，从广告平台来看，主要有：(1) 广告消耗超预算限额；(2) 空结果问题，即某些请求无定向广告参与广告竞价。该问题不但会浪费平台的价值流量，而且不利于建立良好的广告主生态环境。从广告主来看，主要有：(3) 广告预算消耗不尽；(4) 匀速投放问题。该问题影响广告主的广告投放体验以及投资回报率。

对于该预算控制问题，具体分析如下：(1) 对于问题 1，由于广告分配策略未考虑预算消耗信息，当消耗接近预算限额时，未能及时减缓其曝光速度。(2) 对于问题 2，广告量不足是原因之一，但更大的原因可能在于潜在定向广告消耗速度过快。(3) 对于问题 3，主要由于多数中小广告主其竞争力弱，未能获取足够的曝光机会。(4) 对于问题 4，主要是广告按 ecpm 排序，会导致广告消耗速度差异比较大。可以看出，预算控制问题本质上是在线广告分配的问题。

不同于竞价广告，展示广告按千次广告展示计费，其中合约广告是比较特殊的展示广告，其通过广告主和广告平台签订协议，来保证平台在特定售卖时间段内，对于特定人群曝光特定数量的广告。由于合约规模和定向条件的复杂性，展示广告也存在类似的预算控制问题，比如匀速投放问题，广告曝光不足和曝光寡头现象。

由于竞价广告和展示广告的计费方式不同，在线广告分配问题可以分别被抽象看成 Adwords 问题和 Display Ads 问题[2]，其中 Adwords 问题并非指谷歌关键词竞价广告，而是是指 Henzinger[19]提出的竞价广告排序算法的优化问题：在广告预算的限制下，挑选合适的广告进行曝光，以达到广告平台收益最大化或广告预算消耗最大化，并保持广告匀速投放。而 Display Ads 问题是指在曝光次数的限制下，挑选合适的广告进行曝光，以使得最后匹配的所有权重和最大，其中每条匹配的边都有权重。

对于 Adwords 问题，根据不同流量达到顺序模型[2]，可以分成基于流量以最差的顺序到达(Adversarial Order)的算法和基于分布的算法，前一类算法主要存在两种思路[2]：出价缩放算法(bid-scaling)和节流算法(Throttling)，前者是从线性规划原始-对偶问题的角度考虑，而后者是从概率角度考虑。

出价缩放算法主要思想是利用 Adwords 原始对偶问题的“互补松弛性”特征构造其对偶问题的可行解。但是算法非常简单，就是每一次广告请求，对于每一个符合定向要求的广告出价乘以一个缩放因子，选择值最大的广告投放。根据不同的应用场景，缩放因子可以是确定的方法(MSVV[5]，Balance[6])，也可以是随机的方法(Ranking[7]，Perturbed Greedy[2])，还可以是预算使用率的函数(MSVV)。由于该类算法大多有竞争率分析(最差性能分析)，而且其形式相对简单，因此其应用性比较强[2]。

[1] 广点通，jasonnjiang@tencent.com
[2] 广点通，devinzheng@tencent.com
[3] 广点通，nireyhuang@tencent.com
[4] 广点通，applelin@tencent.com

节流算法并非从优化 Adwords 问题的目标函数的角度，而是从概率的角度，通过某种算法设置广告曝光概率来控制其是参与竞价的机会，同时尽量保证广告在投放时间结束时，其广告预算能恰好被消耗完。目前，由于该类方法的输入模型假设条件比较强，其工程实现的难度比较大，比如 Optimized Throttling[17]，其假设给定任何广告的出价，可以准确预估其预算消耗期望值，但其将匹配问题转化为概率问题的思想值得借鉴。

在 Adwords 问题中，基于分布的算法的核心思想是：假设 query 序列分布在一段时间内不变，利用线性规划原始对偶问题 KKT 条件求解出样本线性规划问题的对偶变量，然后基于对偶变量求解 Adwords 问题的近似最优解，例如 Learn-Weights 算法[20]。一般 query 序列是动态变化的，其在一段时间 query 序列分布保持不变是很难满足的，这是该类算法实际应用最大的障碍。

对于 Display Ads 问题，根据不同流量达到顺序模型[2]不同，可以分成基于流量以最差的顺序到达(Adversarial Order)的算法和基于分布的算法。在基于流量以最差的顺序到达算法中，业界存在三种思路：第一种是在线随机算法，指在线性规划框架下，寻找其对偶问题的可行解，比如 Exponential Weighting 算法[12]。第二种是控制论算法，在 query 样本上计算离线线性规划问题的最优解，然后基于控制论方法动态调节最优解，使其适应适应 query 序列和广告信息(广告数，广告日限额，定向条件等等)的动态变化，例如 PID(比例-积分-微分控制器)算法[14]，WaterLevel 算法[4]。第三种是基于流量预测的在线匹配算法，主要针对 Guaranteed Delivery DA 问题(保量展示广告问题)，其试图通过预测的流量，线下生成流量分配计划，从而指导在线的流量分配，例如 HWM 算法[15]和 SHALE 算法[16]。在基于分布的算法中，[23]提出 Two suggested matchings 算法，其利用负载均衡的思想，基于两个不相交的离线可行解指导在线广告分配。

本文在不同流量达到顺序模型[2]中，从竞争率(最差性能分析)的角度，分析 Adwords 问题和 Display Ads 问题中近年来的主要算法，以及简单介绍广点通在预算控制中的实践。

2. 术语说明

为了对比分析各种算法的优劣，我们先对问题定义，流量达到顺序模型和评价指标进行详细说明。首先，我们给出 Adwords 和 Display Ads 问题的详细定义：

- Adwords 问题[5]：Adwords 和 Adwords 问题两者的概念有所不同，前者是指谷歌关键词竞价广告[2]，但后者是指 Henzinger[19]提出的广告排序算法优化问题，具体描述如下[5]：对于在线达到的 query，从定向条件和预算限额的广告集中，挑选出适当的广告进行曝光，以达到广告平台收益最大化或者广告预算消耗最大化的目的。Adwords 问题主要针对竞价广告(例如 CPC 广告)，包括搜索广告的关键词竞价，效果广告的用户竞价(例如，广点通和 Facebook 都是将定向用户展示给参与实时竞价的广告商)等等。为了避免 Adwords 和 Adwords 问题混淆概念，本文中 Adwords 仅仅指 Adwords 问题。
- Online b-Matching 问题[2]：online b-matching 问题是一个特殊的 Adwords 问题，其中，所有的广告预算都是 b，而且所有广告出价都是 1。
- Display Ads 问题[2]：Display Ads 问题是指展示广告问题，具体描述如下：在曝光次数的限制下，挑选合适的广告进行曝光，以使得最后匹配的所有权重和最大，其中每条匹配的边都有权重。Display Ads 问题主要针对 CPM 广告。
- Guaranteed Delivery DA 问题(保量展示广告问题) [18]：其是指特殊 Display Ads 问题，具体是指广告主和广告平台先下达成协议，售卖特定时间段内特定人群特定数量的广告形式，广告主按照曝光付费。

然后，为了能同等条件对比分析算法，我们引入了流量达到顺序模型，这是因为相同的

算法在不同流量达到顺序模型中，其性能可能有所差异，本文中具体涉及到四种流量达到顺序模型[2]：

- 流量以最差的顺序到达(Adversarial Order): Adversarial Order 是指在无任何先验信息的条件下，存在一个对手(Adversary)，其提前知道广告分配算法，从而改变 query 到达的顺序使得分配效果最差。
- 流量以随机的顺序到达(Random Order): 相对于 Adversarial Order，对手对流量达到序列做了随机置换。
- 流量以未知的独立同分布方式到达(Unknown IID): 流量按照一定的分布到达，但该分布我们提前不知道。
- 流量以独立同分布方式到达(IID): 流量按照一定的分布到达，但该分布我们提前知道。

最后，我们给出在线分配算法的评价指标-竞争率，其具体定义如下：

- 竞争率(Competitive Ratio): 竞争率是一种用来分析在线分配算法性能的方法，具体指在线分配算法与最优离线算法的性能比或收益比，表达如下:

$$\text{Competitive Ratio} = \min_{G(U,V,E)} \frac{\text{ALG(G)}}{\text{OPT(G)}}$$

其中，ALG(G)是我们算法的所有广告预算消耗，而 OPT(G)是在已知流量达到顺序和广告信息(广告出价，广告预算和定向条件)的条件下，得到的最优广告预算消耗。

3. Adwords 算法

本章节主要介绍 Adwords 问题相关的算法，其分成若干小节：第一节介绍 Adwords 数学模型；第二节介绍基于 Adversarial Order 的算法；第三节介绍基于 Random Order 和 IID 的算法。其中重点介绍 Adversarial Order 算法，主要理由是：(1) 形式非常简单，容易实现；(2) 能够保证在最坏情况下的竞争率；(3) 算法鲁棒性很强。

3.1 数学模型

Adwords 问题: 在二部图 G(U, V, E)中，其中 U 是广告集，V 是 query 序列，每个广告 u ∈ U 都有一个广告预算B_u，每条定向的边(u, v) ∈ E 都有一个广告出价bid_{uv}，当 v ∈ V 在线到达的时候，我们需要从广告集 U 中找一个预算未消耗完的广告 u 与其匹配，当 v 和 u 匹配成功时，则 u 的预算减少bid_{uv}。如果 u 预算消耗完，则后面将不能做任何匹配，目标是最大化所有广告的预算消耗。图 3.1 就是一个 Adwords 实例，其中$\{u_k\}$是竞价广告集合，每个广告都有定向条件、预算限额和广告出价，$\{v_k\}$是在线到达的用户集合。

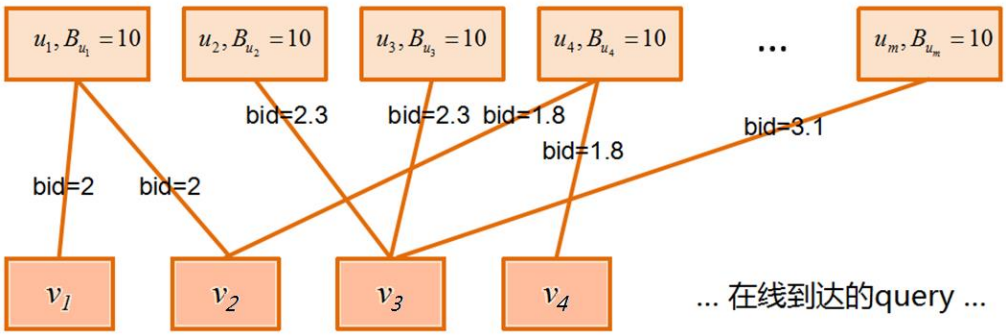

图 3.1 Adwords 实例

3.2 基于 Adversarial Order 的算法

本小节主要介绍基于 Adversarial Order 的算法，即流量以最差的顺序到达，其可以分成出价缩放算法(bid-scaling)和节流算法(Throttling)。

3.2.1 bid-scaling(出价缩放算法)

出价缩放算法主要思想是利用 Adwords 原始对偶问题的“互补松弛性”特征构造其对偶问题的可行解。但是算法非常简单，就是每一次广告请求，对于每一个符合要求的广告出价乘以一个缩放因子，选择值最大的广告投放。

本小节重点介绍 Greedy 算法，Balance 算法和 MSVV 算法[2]，其中 Greedy 算法是众多广告系统(包括广点通，谷歌等等)使用的算法，Balance 算法是匀速投放问题的最优解决方案，而 MSVV 算法集成了两者的优点，当流量以最差的顺序到达时，被认为是 Adwords 问题理论上的最优解决方案[8]。

3.2.1.1 Greedy 算法

Greedy 算法非常简单，就是选择出价最高的广告，其在广告出价差异比较大时非常有效，具体算法如下：

> Greedy 算法：
> 当下一个 query $v \in V$在线到达时：
> - 假设其所有定向的广告的预算消耗完，则无分配；
> - 否则，将 v 分配到出价bid_{uv}最高的广告$u \in U$。

在 Adversarial Order 中，该算法的竞争率仅为 1/2[2]。当广告比较充分时，该算法会引起比较多的广告消耗不尽，而且由于出价直接决定了曝光，当广告定向条件比较宽而且出价比较高时，其消耗速度很容易过快，该现象一方面给广告超限控制带来挑战，另一方面影响了广告平台的用户体验。为了更清楚的说明该算法的不足，这里使用一个比较极端的例子图 3.2：

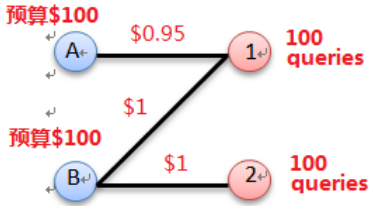

图 3.2 Greedy 算法失效实例

可以看出，图 3.2 中的 bidder A 和 B，每一个 bidder 都有$100 预算，query 这边，首先 100 个 query 1 在线到达，基于 Greedy 算法，则消耗 bidder B 的$100 预算，即 bidder B 预算消耗完了，然后 100 个 query 2 在线到达，由于 bidder B 无预算，而且 query 2 仅定向广告 B，则无法分配。最终预算总消耗为$100，仅为总预算的 1/2。另一方面，可以很明显看出，其消耗速度过快，因为只有 B 有消耗。

3.2.1.2 Balance 算法

为了克服 Greedy 算法的匀速投放问题，[6]提出了 Balance 算法，其忽略单个 bidder 的出价，尽可能平衡所有 bidder 的预算消耗，使得其在线时间尽可能长，即尽量使得所有广告都保持匀速投放，其算法如下：

> Balance 算法:
> 当下一个 query $v \in V$在线到达时：
> - 假设其所有定向的广告的预算消耗完，则无分配；
> - 否则，将 v 分配到消耗与预算比例最小的 bidder。

对于 Balance 算法，其核心思想是平衡所有 bidder 的预算消耗速度，因此其比较好地解决 Greedy 算法的广告匀速投放问题，但在消耗不尽问题上，其仍然非最佳解决方案。在以下实例图 3.3 中，我们可以看出当广告出价不平衡时，其效果也不太理想：

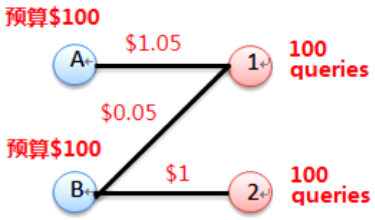

图 3.3 Balance 算法失效实例

图 3.3 中表明 Balance 算法的预算消耗约为$100，从竞争率的角度来看，Balance 算法的竞争率下约为 1/2。对于该现象，Kalyanasundaram 等[6]给出相应的理论分析：

> 理论 3.1[6]. 对于 Adwords 问题，当所有 bidder 的预算都是 b，而且 bid 都是 1，即 online b-matching 问题，Balance 算法的竞争率为 $1-\frac{1}{(1+\frac{1}{b})^b}$，当$b \to \infty$时，其竞争率为$1-\frac{1}{e}$，而且是最优的。

理论 3.1 说明 Balance 算法在 bidder 出价比较接近时表现最优；而对于 bid 差异较大的时候，Balance 算法相反表现较差，但其确实能比较好的解决匀速投放的问题。

3.2.1.3 MSVV 算法

对于 Adwords 问题，Mehta 等人最早意识到广告主日预算的重要性，并且提出目前竞争率最好的算法-MSVV 算法[8]，其同时考虑每个 bidder 的出价和预算未消耗的比例。

3.2.1.3.1 Motivation

前面我们对 Greedy 算法和 Balance 算法分别做了分析，得到各自的优缺点，这里，我们对每个实例分别同时用 Greedy 和 Balance 算法对比分析，如表 3.1：

表 3.1 Greedy 与 Balance 算法对比分析

算法	Greedy 算法失效实例 竞争比	Balance 算法失效例子 竞争比
Greedy 算法	1/2	1
Balance 算法	3/4	1/2

该表说明 Balance 算法在 bidder 出价比较接近时表现最优；而对于 bid 差异较大的时候 Greedy 算法表现较好，Balance 算法相反表现较差。因此当需要同时兼顾竞争率和匀速投放时，很自然的想法是将 Greedy 算法和 Balance 算法融合成新的算法，即同时考虑广告出价和预算限额，这就是 Mehta 提出的思路[8]。

3.2.1.3.2 算法

为了更清楚描述新的算法，先给出一些基本的定义:

- x_u：在广告 u 的总预算中，当前总消费的比例，也就是 当前总消费/广告 u 的总预算。
- $\psi(x_u)$: 权衡函数$\psi(x_u) = 1 - e^{x_u - 1}$，其是一个随着$x_u$增加而单调下降的函数，其在[0,1]的分布，如图 3.4：

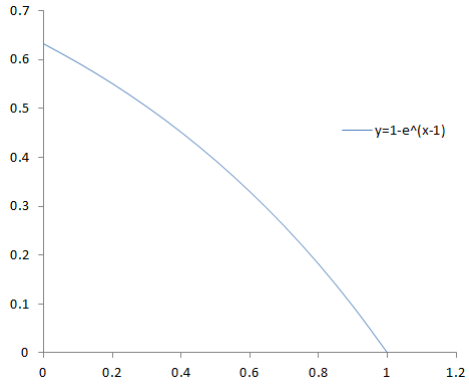

图 3.4 权衡函数分布图

MSVV 算法，其同时考虑 bidder 的出价以及消耗预算比例。与 Greedy 算法相似，MSVV 每次都是将 query 分配给 scaled bid 最大的广告，其中 scaled bid 是指 bid 乘以权衡函数$\psi(x_u)$，具体算法如下：

MSVV 算法[5]:
当下一个 query v ∈ V在线到达时：
- 假设其所有定向的广告的预算消耗完，则无分配；
- 否则，将 query v 分配给 bidder u，其$bid_{uv}\psi(x_u)$值最大。

当所有 bid 相等时，由于权衡函数$\psi(x_u)$是一个单调下降的函数，因此 MSVV 算法就正好退化成了 Balance 算法。另一方面，对于每个 query 的出价如果差异非常大时，MSVV 算法在大多数情况下，都不会颠倒 bidder 的出价顺序，此时 MSVV 表现更像 Greedy 算法。更极端的情况，当所有的广告预算都是无限时，MSVV 算法直接退化成了 Greedy 算法，这是由于所有平衡函数值为：$\psi(0) = 1 - \frac{1}{e}$。

对于 MSVV 算法，[21]给出了优化策略，就是将 MSVV 算法和“某种算法”融合成新算法，具体如下：

新 MSVV 算法[21]:
当下一个 query v ∈ V在线到达时：
- u_1是“某种算法”选择的分配广告。
- u_2是 MSVV 算法选择的分配广告，其中$\psi(x_u) = 1 - e^{\alpha(x_u - 1)}$，$\alpha$为“某种算法”的可信度。
- 当$\alpha bid_{u_1 v}\psi(x_{u_1}) \geq bid_{u_2 v}\psi(x_{u_2})$时，将 u_1 分配给 v，否则将 u_2 分配给 v。

[21]指出，当“某种算法”为基于历史 query 序列的 Adwords 线性规划的最优解时，当该最优解不可靠时，新 MSVV 算法的竞争率至少为 1-1/e；当其可靠时，新 MSVV 算法的竞争率接近 1。

3.2.1.3.3 性能分析

与前面算法分析类似，我们同时给出竞争率和实例分析。

MSVV 算法的竞争率如下：

理论 3.2[5]在 adversarial order 输入模型中，当权重 bid_{uv} 相对于 u 的预算 B_u 非常小时，即满足 The Small-Bids Assumption，MSVV 算法的 competitive ratio 为$1 - \frac{1}{e}$。

理论 3.2 反映出 MSVV 算法相对 Greedy 算法提高了约 34%，为了进一步比较其与前两

种方法的优势，我仍然使用极端例子图 3.1 和图 3.2，进行对比分析，如表 3.2：

表 3.2 Greedy，Balance 和 MSVV 算法对比分析

算法	Greedy 算法失效实例竞争率	Balance 算法失效例子竞争率	竞争率均值
Greedy 算法	1/2	1	0.75
Balance 算法	3/4	1/2	0.62
MSVV 算法	0.73	0.99	0.86

表 3.2 反映出，在这两个极端实例中，MSVV 都接近其它两种方法表现最优的算法，而且从平均竞争率来看，相对 Greedy 算法提升了 14.8%，而相对 Balance 算法则提升了 37.7%。

3.2.1.3.4　如何理解权衡函数$\psi(x) = 1 - e^{x-1}$

权衡函数$\psi(x) = 1 - e^{x-1}$是 Adwords 问题与 Online b-Matching 问题最优解相同的充分条件，更准确的说，是 Adwords 对偶问题和 Online b-Matching 对偶问题最优解相同的充分条件。

为了更好地理解 MSVV 权衡函数的意义，从 LP(线性规划)原始对偶问题角度做简单分析。基于[5]中的引理 5，对于具体的权衡函数ψ和 query 序列π，我们可以建立起 Online b-Matching 问题和 Adwords 问题的对偶问题的联系如下表 3.3：

表 3.3 线性规划原始对偶问题分析

数学模型	原始问题	对偶问题
Online b-Matching	$Max\ c \cdot x$ $s.t.\ Ax \le b\ and\ x \ge 0$	$Min\ b \cdot y$ $s.t.\ A^T y \le c\ and\ y \ge 0$
Adwords	$Max\ c \cdot x$ $s.t.\ Ax \le b + \boldsymbol{\Delta}(\boldsymbol{\psi}, \boldsymbol{\pi})\ and\ x \ge 0$	$Min\ b \cdot y + \boldsymbol{\Delta}(\boldsymbol{\psi}, \boldsymbol{\pi}) \cdot \boldsymbol{y}$ $s.t.\ A^T y \le c\ and\ y \ge 0$

其中，$\Delta(\psi, \pi)$向量中第 i 个元素为$\sum_{k=1}^{i} \alpha_k - \beta_k$，其中α和β分别为 Online b-Matching 原始/对偶问题的最优解。

对比 Online b-Matching 和 Adwords 的对偶问题，可以发现：

- 两者的约束条件完全一致。
- 两者的目标函数仅相差$\boldsymbol{\Delta}(\boldsymbol{\psi}, \boldsymbol{\pi}) \cdot \boldsymbol{y}$，

理论 3.1 表明 Online b-Matching 问题的 Balance 算法的竞争比为 1-1/e，如果需要使得在 Adwords 中的 MSVV 算法竞争比同样为 1-1/e，则从对偶问题角度来看，仅仅需要选择适当的权衡函数ψ，能够使得$\boldsymbol{\Delta}(\boldsymbol{\psi}, \boldsymbol{\pi}) \cdot \beta = 0$，其中β为 Online b-Matching 的最优解，这就是 MSVV 中权衡函数$\psi(x) = 1 - e^{x-1}$的由来。

3.2.2　Throttling(节流算法)

Throttling 算法最初设计目标为解决匀速投放和预算消耗不尽的问题，虽然其思路非常简单，就是设计每个广告的参与竞拍的概率，但其实现难度其实非常大，主要是因为其需要很强 query 输入模型假设条件，但其将广告匹配问题转化成概率问题的思想确实值得借鉴，实际上 Display Ads 中的 HWM(High Water Mark)算法[15]也有相似的思路。

本小节介绍 optimized throttling 算法[17]，其基于最大化投资回报率来控制每个广告参与竞价的概率。该算法需要如下前提条件：

新的输入模型假设：对于x ∈ R，对于 query v ∈ V，当$\mu_u(v) \ge x$时，可以预估广告 u 在 query v 上的期望预算消耗$S_u(x)$，其中$\mu_u(v)$为广告 u 和 query v 的函数。例如，令μ = CTR，当 CTR(点击率)最小为 x 时，可以预估到广告在 query 上的期望预算消耗。

在新的输入模型假设条件下，我们可以将针对广告的 Throttling(节流算法)看成在线背包问题。论文中提出以投资回报率(ROI)为优化目标，相应的背包问题为：

$$\max_{V_u \subseteq V} \sum_{v \in V_u} \nu_u \mathrm{CTR}(u,v)$$

$$\text{s.t.} \quad \sum_{v \in V_u} \mathrm{CTR}(u,v)\mathrm{CPC}(u,v) \le B_u$$

其中，CTR 为点击率，CPC 为每次点击的消耗，v_u为每次点击的价值，也就是转化率。该背包问题的离线贪婪近似解见 Optimized Throttling 算法：

Optimized Throttling 算法： 离线阶段： 1. $\mu_u(v) = \mathrm{CPC(u,v)}$ 2. 基于μ预估 u 的预算消耗分布 3. 基于预算消耗分布和剩余预算，预估一个阈值γ 在线阶段： 当下一个 query v ∈ V在线到达时： 当 $\mathrm{CPC(u,v)}$ > γ时，容许广告 u 参与广告竞拍， 否则限制广告 u

可以看出，Optimized Throttling 算法虽然有很好的理论基础，但其很强输入模型假设条件阻碍了其在实际系统中的应用。

3.3 基于 Random Order 和 IID 的算法

本小节主要介绍基于分布的在线分配算法，该分布包括 query 以随机顺序到达、query 以未知的独立同分布方式到达和 query 以已知的独立同分布方式到达。该方法假设 query 序列分布在一段时间内不变，利用线性规划原始对偶问题 KKT 条件求解出样本线性规划问题的对偶变量，然后基于对偶变量求解 Adwords 问题的近似最优解。

3.3.1 Learn-Weights 算法[20]

Learn-Weights 算法的核心思想是：假设 query 以随机顺序到达，基于历史 query 序列样本构建的线性规划问题，其求解出的对偶变量可以指导剩下 query 序列的在线广告分配，其中图 3.5 为 Adwords 线性规划原始对偶问题，其中$\{\alpha_u, \beta_v\}$为对偶变量。

原始问题	对偶问题
$\max \sum_{u,v} x_{uv} bid_{uv}$	$\min \sum_u B_u \alpha_u + \sum_v \beta_v$
$s.t.\ \forall\ u\ :\ \sum_v x_{uv} bid_{uv} \le B_u$	$s.t.\ \forall\ u,v:\ bid_{uv}\ \alpha_u + \beta_v\ \ge\ bid_{uv}$
$\forall\ v:\ \sum_u x_{uv} \le 1$	$\alpha_u \ge 0\ \ \forall u$
$x_{uv} \ge 0$	$\beta_v \ge 0\ \ \forall v$

图 3.5 Adwords 线性规划原始对偶问题

更正式地描述，基于历史 query 序列样本构建的线性规划对偶问题$\hat{D}$构建如下：

第1步. 抽样出 query 序列中的ε比例，$\hat{V} \subseteq V$(假设 query 序列长度可以预估)。

第2步. 对于所有的广告 u，其广告预算从B_u变成εB_u。

第3步. 仅仅针对(u,v)的约束条件，其中v ∈ $\hat{V}$，u∈ U。

当 $\{\hat{\alpha}_u^*\}$ 为$\hat{D}$的最优解(对偶问题最优解)，则对于剩余 query 序列 v ∈ V − $\hat{V}$，利用$\hat{\alpha}_u^*$和互补松弛性进行广告分配，其中互补松弛性如图 3.6。

$$x_{uv} > 0 \Rightarrow \mathrm{bid}_{uv}(1-\alpha_u) = \beta_v$$
$$\alpha_u > 0 \Rightarrow \sum_v x_{uv}\mathrm{bid}_{uv} = B_u$$
$$\beta_v > 0 \Rightarrow \sum_u x_{uv} = 1$$

图 3.6 Adwords 线性规划原始对偶问题的互补松弛性

“互补松弛性”简单的理解如下：

- 可行解 $\{x_{uv}, \alpha_u, \beta_v\}$ 如果是最优的，则v ∈ V分配的 bidder u 最大化$bid_{uv}(1-\alpha_u)$。这意味着当α_u存在对偶最优时，我们能通过将 v 分配给值最大$bid_{uv}(1-\alpha_u)$的 bidder u, 可以重构对偶问题的最优解。
- α_u为正 当且仅当 bidder u 消耗了所有的预算。

在 online 问题中，我们不知道整个 LP, 所以我们得不到最优解。但是，adwords 原始对偶问题的“互补松弛性”特征可以指导预估α_u，并能获取比较好的竞争率(competitive ratio)。于是利用$bid_{uv}(1-\hat{\alpha}_u^*)$就可以指导在线广告分配，相应的算法如下：

Learn-Weights 算法[20]:
当下一个 query v ∈ V在线到达时：
- 假设其所有定向的广告的预算消耗完，则无分配；
- 否则，将 query v 分配给 bidder u，其$bid_{uv}(1-\hat{\alpha}_u^*)$值最大

[20]同样给出相应的竞争率为 1-ε。虽然 Learn-Weights 算法给出了理论上进一步提升的竞争率，但其应用性不强：第一个原因是很难规定 query 序列以随机顺序到达；第二个原因是离线求解样本 LP(线性规划)对偶问题最优解实际上是一个 NP 问题。

4. Display Ads 算法

本章节主要介绍 Display Ads 问题相关的算法，其分成若干小节：第一节介绍 Display Ads 数学模型；第二节介绍基于 Adversarial Order 的算法；第三节介绍基于 Random Order 和 IID 的算法。与 Adwords 算法理由相同，我们重点介绍基于 Adversarial Order 的算法。

4.1 数学模型

Display Ads 问题: 在二部图 G(U, V, E)中，其中 U 是广告集，而 V 是 query 序列，每个广告u ∈ U都有最多展示次数的预算 n(u)次，每条边都具有权重w_{uv}，目标就是最大化匹配边的权重和。例如图 4.1，第一行是广告集合，其包括定向条件；第二行是 query 集合。

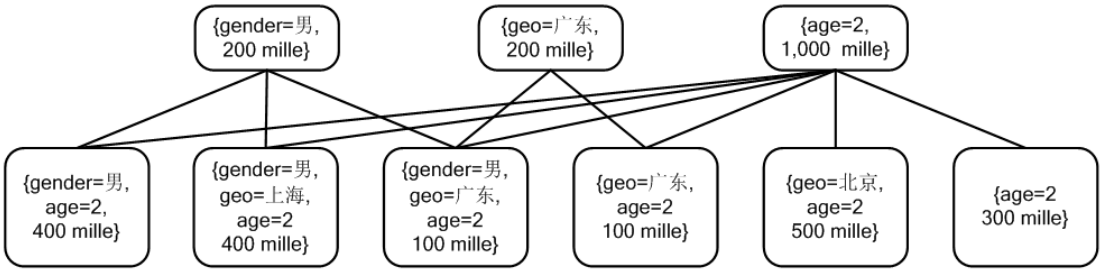

图 4.1 Display Ads 实例

Guaranteed Delivery DA 问题(保量展示广告问题) [18]: 其是指特殊 Display Ads 问题，具体是指广告主和广告平台先下达成协议，售卖特定时间段内特定人群特定数量的广告形式，广告主按照曝光付费。其优化目标如下：

$$argmax_{x_{u,v}}\left(\sum_{u,v} w_{uv}x_{uv} - \sum_v p_v r_v\right)$$

其中，这里省略了约束条件，r 是剩余曝光量，p 是合约的重要度和x_{uv}是我们要求解的分配方案。

4.2 基于 Adversarial Order 的算法

本小节主要介绍基于 Adversarial Order 的算法，即流量以最差的顺序到达，其可以分成在线随机算法(Free Disposal)、基于控制论的算法和基于流量预测的在线匹配算法。

4.2.1 Free Disposal 算法(在线随机算法)

Free Disposal 算法(在线随机算法)主要利用 Display Ads 的原始和对偶可的行解指导线上流量分配。

原始问题

$$max \sum_{v,u} w_{uv}x_{uv}$$
$$\sum_{u} x_{uv} \le 1 \ (\forall v)$$
$$\sum_{v} x_{uv} \le n(u) \ (\forall u)$$

对偶问题

$$min \sum_{u} n(u)\beta_u + \sum_{v} z_v$$
$$\beta_u + z_v \ge w_{uv}(\forall v, u)$$
$$\beta_u, z_v, x_{uv} \ge 0$$

基于 Free Disposal 算法的原始对偶问题，通过以下步骤可以构建对偶问题的可行解：

1) 初始化：∀ u, $\beta_u = 0$。
2) 当 query v 在线到达时，将其分配给$u' \in U$使其最大化$w_{u'v} - \beta_{u'}$。
3) $x_{u'v} = 1$，当 u′已经有 n(u′)个展示时，$x_{u'v} = 0$
4) 在对偶解中，更新$z_v = w_{u'v} - \beta_{u'}$。

其算法框架如下：

> Free Disposal 算法[12]:
> 当下一个 query v ∈ V在线到达时：
> - 假设其所有定向的广告的预算曝光次数已经完成，则无分配；
> - 否则，将 v 分配给 $w_{uv} - \beta_u$最大的广告 u。

表 4.1 描述了β_u的不同策略，以及相应的竞争率。

表 4.1 Free Disposal 及其竞争率分析

β_u	算法	竞争率
Greedy	对于每个广告 u，β_u是分配给 u 的前 n(u)个高权重展示中最低的权重，也即 u 接受一个新的展示需要抛弃的权重	1/2
Uniform Weighting	对于每个广告 u，β_u是分配给 u 的前 n(u)个高权重展示的权重的算法平均。如果分配给 u 的展示少于 n(u)个，β_u是这些展示总权重与 n(u)的比。	1/2
Exponential Weighting	对于每个广告 u，β_u是分配给 u 的前 n(u)个高权重展示的权重的指数加权。当权重排序为$w_1 \ge w_2 \ge \cdots \ge w_{n(u)}$时，$\beta_u = \{n(u)[(1+n(u)^{-1})^{n(u)} - 1]\}^{-1} \sum_{j=1}^{n(u)} w_j(1+n(u)^{-1})^{j-1}$	1-1/e (当 n(u)对每个 u 都充分大)

4.2.2 基于控制论的算法

该方法将流量分配问题转化为一个控制论问题，具体如下：先基于历史数据计算出对偶问题的最优解$\{\beta_a\}$，然后基于控制论方法动态调节β_a以适应 query 序列和广告信息的动态变化。比较常见的方法是 PID 控制算法[14]和 Watelevel 控制算法[4]，其算法如下：

4.2.2.1 PID 控制算法

PID(比例-积分-微分控制)控制法，其控制器定义如下：

$$\beta_a(t+1) \leftarrow \beta_a(t) - k_1 e_a(t) - k_2 \int_0^t e_a(\tau)\, d\tau$$

其中，

- 在 t 时刻时，$e_a(t) = r_a(t) - r'_a(t)$为赢取 bid 的期望概率与实际观察概率的偏差，其中$r_a(t)$为赢取 bid 的期望概率，$r'_a(t)$为赢取 bid 的实际观察概率。例如 $r_a(t) = \frac{x_a(t)}{n(a)}$，$x_a(t)$为 t 时刻中的广告 a 赢得的展示次数。
- k_1 为比例增益参数。
- k_2 为积分增益参数。

4.2.2.2 Waterlevel 控制算法

Watelevel 算法是另一控制方法(广点通多比例复用采用类似的策略)，其具体定义如下：

$$\forall u, \beta_a(t+1) \leftarrow \beta_a(t)\exp(\gamma(^{x_a(t)}/_{n(a)} - {}^{1}/_{T}))$$

其中，γ控制 Waterlevel 对偏差的反馈速度。

4.2.3 基于流量预测的在线匹配算法

针对 Guaranteed Delivery DA 问题(保量展示广告问题)，Yahoo[15]提出了基于流量预测的在线匹配算法，其通过线下生成的 compact allocation plan(紧凑分配计划)指导线上决策，包括广告分配的优先级和曝光概率，例如 HWM 算法[15]和 SHALE 算法[16]， 其中 HWM 算法是 Greedy 算法，更精确的求解可以参考 SHALE 算法，但其在工程实现上和复杂度上远大于 HWM 算法[18]。

4.2.3.1 HWM(High Water Mark)算法[15]，Yahoo!实际系统中用的方法

HWM 离线服务使用了一个简单而非常有效的启发算法来产生分配方案：算法为每个合约 j 生产一个服务率α_j和一个分配顺序。分配顺序是让满足度低的合约可以有较高的优先权得到更多的可行流量，通过对每个合约设置服务率和分配顺序，HWM 算法就可以创建一个紧凑且健壮的分配方案。在线服务时，每个合约得到一次展示的α_j比例，除了非流量不足的情况下，如果流量不足，会通过分配顺序来解决，一个分配顺序靠前的合约 j 会得到α_j比例，下一个合约 j'会得到$\alpha_{j'}$,如果不足$\alpha_{j'}$，新得到剩余所有的比例。具体架构图可以参考图 4.2。

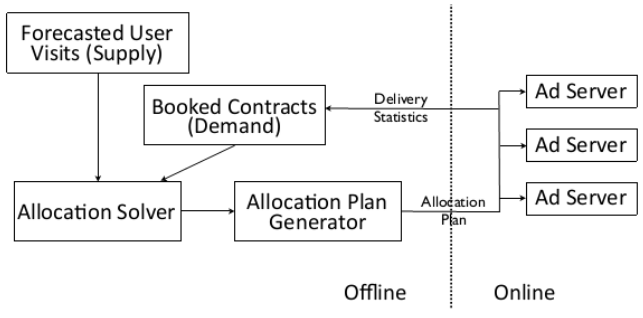

图 4.2 HWM 算法的系统架构

4.2.3.1.1 算法

- 离线服务

基于流量预测模块，预测每个合约 j 的可行流量 s_j，并根据 s_j 来决定分配顺序，较小的 s_j 值的合约、离合约结束时间比较近的合约，或者定向条件比较少的合约，其分配顺序比较靠前，为了决定服务率α_j，HWM 进行如下步骤：

1) 对所有的合约 j，初始化剩余流量$r_j = s_j$;
2) 以分配顺序对每个 j, 进行:
   a) 求解服务率α_j: $\sum_{i\in\Gamma(j)} \min\{r_i, s_i\alpha_j\} = n(j)$，无解时$\alpha_j = 1$。
   b) 更新：$\forall$ i $\in$ $\Gamma(j)$, $r_i = r_i - \min\{r_i, s_i\alpha_j\}$

注意：分配顺序方案依赖于流量预测模块，因此不正确的流量预测会产生次优的结果。

■ 在线服务

基于分配顺序和服务率α_j，在线预测模块的任务就是选择一个满足定向条件的合约，具体算法如下：

1) 给定一个展示 i, 令J = $\{c_1, c_2, \dots, c_{|J|}\}$为满足定向条件的合约广告列表，并且以分配顺序排序。
2) 如果$\sum_{j=1}^{|J|} \alpha_j > 1$，令 l 为满足$\sum_{j=1}^{|l|} \alpha_j \le 1$条件的最大值，最后令$\alpha'_{l+1} = 1 - \sum_{j=1}^{|l|} \alpha_j$。
3) 以α_j的概率选择一个合约j $\in$ [1, l]，并以α'_{l+1}的概率选择 l+1 合约广告。注意如果在$\sum_{j=1}^{|J|} \alpha_j < 1$的情况下，可能没有合约被选中。

4.2.3.1.2 鲁棒性分析

由于 HWM 依赖流量预测模块，因此流量预测的准确性一定程度上影响我们的算法，对于该问题，[15]文中提出两种方法解决该问题：

1) 频繁的再计算

作者提出该观点，并且给出了相应的证明：

定理 4.1：假设我们有 k 次再优化迭代，流量预测错误(错误率为 1-实际流量/预测流量)为 r, 且合约是可完成的，在 r>0 的情况下，under-delivery 是一个正值，且边界为$\frac{r+r^2}{k^{1-r}}$，在 r<0 的情况下，over-delivery 是正值，边界为$\frac{|r|}{k^{1-r}}$。

2) 基于反馈的流量修正

[15]给出一个非常简单的反馈控制器，而且声称在实践中表现很好。这个解决方案用两个参数进行控制：δ：控制展示过程中允许的松弛；(β+, β−)控制对服务率的提升。

控制过程：如果在合约的合约时间内，合约 delivery 的速度落后了δ小时，反馈系统会将剩余未展示的量乘上β −系数；反之，合约 delivery 的速度超过了δ小时，则反馈系统将剩余的展示量乘以β +系数。

4.3 基于 Random Order 和 IID 的算法

对于 Display Ads 问题，在流量以最差方式到达时，其最优竞争率为 1-1/e=0.632，但[23]提出了 Two suggested matchings 算法，当流量以独立同分布方式到达(IID)时，最竞争率可以达到 0.67。该算法简单描述如下：

第1步. 基于样本 query 序列，将广告分配问题转化为最大流问题，并以此构建两个不相交的可行解，其分别为 A 和 B。

第2步. 在线分配时，利用负载均衡的思想：对于在线到达的 query，先使用 A 获取匹配广告；若匹配失败，再考虑 B。

5. 广点通在预算控制中的实践

广点通是腾讯的效果广告系统，依托于腾讯海量优质流量资源，给广告主提供跨平台、跨终端的网络推广方案，并利用腾讯大数据处理算法实现成本可控、效益可观、智能投放的互联网效果广告平台[24]。

广点通支持多种计费类型的广告，其中最主要的是 CPC（按照点击计费）模式，其他计费模式还包括 CPM（曝光次数计费）、CPA（按照转化计费）、以及 CPD（按天计费）。对于 CPC 模式，就是选择 ecpm 最大的广告进行曝光，其中 ecpm = CTR（广告点击率） * BID（广告出价）。

为了使得广告投放更加匀速和广告竞价更加充分，以帮助广告主提高投资回报率，对于 CPC 广告，广点通采用 Throttling 算法，即通过曝光概率控制广告参与竞拍的机会，其主要依托于基于反馈的控制论技术和匀速投放技术。具体来讲，是基于反馈的剩余预算和历史预算消耗速度生成每个广告参与竞拍的概率，以保持其匀速投放。不同于 Optimized Throttling 算法，我们不需要任何的输入模型假设。

对于 CPM 广告，广点通采用 HWM 算法，其依赖预估的流量离线生成 compact allocation plan(紧凑分配计划)[15]，以指导线上流量分配，包括广告分配的优先级和曝光概率。

6. 参考文献

[1] devinzheng, nireyhuang《预算控制效果分析》

[2] A. Mehta, Online Matching and Ad Allocation, in Foundations and Trends, pp.266-304, 2012

[3] Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur, Real-time bidding algorithms for performance-based display ad allocation, in KDD, pp. 1307–1315, 2011.

[4] D. Charles, M. Chickering, N. R. Devanur, K. Jain, and M. Sanghi. Fast algorithms for finding matchings in lopsided bipartite graphs with applications to display ads. In EC 2010.

[5] A. Mehta, A. Saberi, U. V. Vazirani, and V. V. Vazirani, Adwords and gen-eralized online matching, Journal of ACM, vol. 54, no. 5, 2007

[6] B. Kalyanasundaram and K. Pruhs, An optimal deterministic algorithm for online b-matching, Theoretical Computer Science, vol. 233, no. 1–2, pp. 319–325, 2000.

[7] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, An optimal algorithm for on-line bipartite matching, in STOC, pp. 352–358, 1990.

[8] S. Robinson. Computer scientists optimize innovative ad auction. SIAM News, 38:243–273, 2005

[9] N. R. Devanur, Online algorithms with stochastic input, SIG ecom Exchanges, vol. 10, no. 2, pp. 40–49, 2011.

[10] N. R. Devanur, B. Sivan, and Y. Azar, Asymptotically optimal algorithm for stochastic adwords, in Proceedings of the ACM Conference on Electronic Commerce, pp. 388–404, 2012.

[11] J. Feldman, A. Mehta, V. S. Mirrokni, and S. Muthukrishnan, Online stochastic matching: Beating 1-1/e, in FOCS, pp. 117–126, 2009.

[12] J. Feldman, N. Korula, V. S. Mirrokni, S. Muthukrishnan, and M. P´ al, Online ad assignment with free disposal, in WINE, pp. 374–385, 2009.

[13] Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur, Real-time bidding algorithms for performance-based display ad allocation, in KDD, pp. 1307–1315, 2011.

[14] D. Charles, M. Chickering, N. R. Devanur, K. Jain, and M. Sanghi. Fast algorithms for finding matchings in lopsided bipartite graphs with applications to display ads. In EC 2010.

[15] P. Chen, W. Ma, S. Manalapu, C. Nagarajan, S. Vassilvitskii, E. Vee, M. Yu, and J. Zien. Ad

serving using a compact allocation plan. WWW, 2012.
[16] Erik Vee , Sergei Vassilvitskii , Jayavel Shanmugasundaram, Optimal online assignment with forecasts, Proceedings of the 11th ACM conference on Electronic commerce, June 07-11, 2010, Cambridge, Massachusetts, USA
[17] C. Karande, A. Mehta, and R. Srikant, Optimization of budget constrained spend in search advertising, in WSDM, 2013.
[18] Erikhu 《效果广告好声音》
[19] Monika Henzinger. Private communication. 2004
[20] N. R. Devanur and T. P. Hayes, The adwords problem: online keyword match-ing with budgeted bidders under random permutations, in ACM Conference on Electronic Commerce, pp. 71–78, 2009.
[21] M. Mahdian, H. Nazerzadeh, and A. Saberi, Allocating online advertisement space with unreliable estimates, in ACM Conference on Electronic Commerce, pp. 288–294, 2007.
[22] Vijay Bharadwaj, Peiji Chen, Wenjing Ma, Chandrashekhar Nagarajan, John Tomlin, Sergei Vassilvitskii, Erik Vee, Jian Yang: SHALE: an efficient algorithm for allocation of guaranteed display advertising. KDD 2012: 1195-1203
[23] J. Feldman, A. Mehta, V. S. Mirrokni, and S. Muthukrishnan, Online stochas-tic matching: Beating 1-1/e, in FOCS, pp. 117–126, 2009.
[24] meteorli, nelsonliao《新广告新广告选取策略》