

(19) 中华人民共和国国家知识产权局



## (12) 发明专利申请

(10) 申请公布号 CN 104102576 A

(43) 申请公布日 2014. 10. 15

---

(21) 申请号 201310127577. 4

(22) 申请日 2013. 04. 12

(71) 申请人 阿里巴巴集团控股有限公司

地址 英属开曼群岛大开曼岛资本大厦一座  
四层 847 号邮箱

(72) 发明人 欧舟

(74) 专利代理机构 北京集佳知识产权代理有限  
公司 11227

代理人 任苏亚 王宝筠

(51) Int. Cl.

G06F 11/36(2006. 01)

G06F 17/30(2006. 01)

---

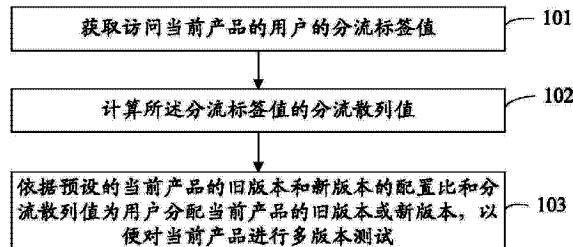
权利要求书2页 说明书14页 附图4页

(54) 发明名称

一种多版本测试方法和装置

(57) 摘要

本申请提供了一种多版本测试方法和装置，所述方法包括：获取访问当前产品的用户的分流标签值，所述分流标签值用于唯一标识用户；计算所述分流标签值的分流散列值；依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本，以便对所述当前产品进行多版本测试。采用本申请实施例提供的方法或装置，能够提供一种通用的基于分流散列值实现的方便的多版本测试方法；进一步的，还能解决用户差异性所带来的多版本测试结果的有效性和准确性的问题，提供有效的验证方法，还能解决同一产品在并行进行多个多版本测试时出现的分流互相干扰的问题。



1. 一种多版本测试方法,其特征在于,该方法包括:

获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户;

计算所述分流标签值的分流散列值;

依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品进行多版本测试。

2. 根据权利要求 1 所述的方法,其特征在于,所述获取访问当前产品的用户的分流标签值,具体包括:

判断用户的 Web 请求中的访问数据 Cookie 中是否存在所述分流标签值,如果是,则直接从用户的 Cookie 中提取所述分流标签值,如果否,则依据预设的分流标签值的生成策略为所述用户生成分流标签值,并将所述分流标签值添加至所述用户的 Cookie 中。

3. 根据权利要求 2 所述的方法,其特征在于,所述依据预设的分流标签值的生成策略为所述用户生成分流标签值,具体包括:

获取所述用户的 IP 地址、第一次访问所述当前产品的时间和一随机数;

依据所述 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值。

4. 根据权利要求 1 所述的方法,其特征在于,所述计算所述分流标签值的分流散列值,包括:

将所述分流标签值转换为哈希 hash 码值;

计算所述 hash 码值对应的初始散列值;

对所述初始散列值进行分流优化处理后得到分流散列值。

5. 根据权利要求 4 所述的方法,其特征在于,所述对所述初始散列值进行分流优化处理后得到分流散列值,具体包括:

对所述初始散列值按照预设的平移区间进行平移后得到平移的分流散列值;或者,

对所述初始散列值按照预设的反转规则进行反转后得到反转的分流散列值;或者,

对所述初始散列值与预设的乘积因子进行相乘后得到相乘的分流散列值;或者,

对所述初始散列值按照预设的时间参数进行散列后得到散列的分流散列值。

6. 根据权利要求 1 所述的方法,其特征在于,所述对所述当前产品进行多版本测试,具体包括:

依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。

7. 根据权利要求 6 所述的方法,其特征在于,所述依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试,具体包括:

分别依据访问旧版本和新版本的用户的转化率,获取访问旧版本的所有用户的旧版本转化率,以及访问新版本的所有用户的新版本转化率;

依据所述旧版本转化率和新版本转化率对所述旧版本和新版本进行测试。

8. 一种多版本测试装置,其特征在于,该装置包括:

获取模块,用于获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户;

计算模块,用于计算所述分流标签值的分流散列值;

分配模块,用于依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品进行多版本测试。

9. 根据权利要求 8 所述的装置,其特征在于,所述获取模块具体包括:

判断子模块,用于判断用户的 Web 请求中的访问数据 Cookie 中是否存在所述分流标签值;

提取子模块,用于在所述判断子模块的结果为是的情况下,直接从用户的 Cookie 中提取所述分流标签值;

生成子模块,用于在所述判断子模块的结果为否的情况下,依据预设的分流标签值的生成策略为所述用户生成分流标签值;

添加子模块,用于将所述生成子模块生成的分流标签值添加至所述用户的 Cookie 中。

10. 根据权利要求 9 所述的装置,其特征在于,所述生成子模块具体包括:

获取参数子模块,用于获取所述用户的 IP 地址、第一次访问所述当前产品的时间和一随机数;

组合子模块,用于依据所述 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值。

11. 根据权利要求 7 所述的装置,其特征在于,所述计算模块包括:

转换子模块,用于将所述分流标签值转换为哈希 hash 码值;

计算子模块,用于计算所述 hash 码值对应的初始散列值;

优化子模块,用于对所述初始散列值进行分流优化处理后得到分流散列值。

12. 根据权利要求 11 所述的装置,其特征在于,所述优化子模块,具体包括:

平移子模块,用于对所述初始散列值按照预设的平移区间进行平移后得到平移的分流散列值;或者,

反转子模块,用于对所述初始散列值按照预设的反转规则进行反转后得到反转的分流散列值;或者,

相乘子模块,用于对所述初始散列值与预设的乘积因子进行相乘后得到相乘的分流散列值;或者,

散列子模块,用于对所述初始散列值按照预设的时间参数进行散列后得到散列的分流散列值。

13. 根据权利要求 8 所述的装置,其特征在于,还包括:

测试模块,用于依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。

14. 根据权利要求 13 所述的装置,其特征在于,所述测试模块包括:

获取转化率子模块,用于分别依据访问旧版本和新版本的用户的转化率,获取访问旧版本的所有用户的旧版本转化率,以及访问新版本的所有用户的新版本转化率;

测试子模块,用于依据所述旧版本转化率和新版本转化率对所述旧版本和新版本进行效果测试。

## 一种多版本测试方法和装置

### 技术领域

[0001] 本申请涉及互联网数据处理领域,特别涉及一种多版本测试方法和装置。

### 背景技术

[0002] 在互联网中,产品在不断发展,例如,在淘宝网中,如果对旧版本进行优化之后,其优化后的版本就是相对于旧版本的新版本。在网站的设计中,为了提高用户的使用体验,更会时常对网站进行改版或者优化。例如,网站上的某个按钮是用红色还是用蓝色,网站上的某块区域排版布局是横排还是竖排,几种权重不同的调整算法的取舍或者某些关键性位置文案的选择,等等。而为了测试网站的新版本和旧版本的效果差异,一个比较好的方法就是对这两个版本进行分流、记录和效果分析的多版本测试过程。

[0003] 多版本测试一般也称为A/B测试,一般情况下,A指的是网站的一个旧版本,而B则表示网站的新版本。多版本测试是一种新兴的产品优化方法,简单来说,就是同时将用户分为浏览旧版本和浏览新版本的两部分,分给旧版本的用户就看到旧版本,而分给新版本的用户就会看到新版本,并记录下旧版本和新版本下用户的行为,最后根据旧版本和新版本的用户行为的数据分析,来对旧版本和新版本进行比较测试,以测试出哪个版本的产品更好、更优。

[0004] 可见,一个A/B测试典型的过程包含三部分:分流、记录和数据分析。具体分流的方式具有多样性,例如:有一种是按用户比例分流的方式:是在进行A/B测试时,按用户维度进行分流,如果一个产品在A/B测试期间访问的用户总数为20万,而分流比例为50%和50%,那么会有10万用户分配给新版本,另外10万用户分配给老版本。还有一种方式是按用户请求比例分流:这是在进行A/B测试时,按用户请求维度进行分流,如果一个产品在A/B测试期间访问的用户为20万,一个用户可能会访问该产品多次,假设平均每个用户访问7次,分流比例为50%和50%,那么会有70万次请求分配给新版本,另外70万次请求分配给老版本。可见,“按用户比例分流”和“按请求比例分流”比较大的区别在于前者的同一用户始终看到的是相同的版本,并且用户数的比例和分流比例一致,请求数的比例可能和分流比例不一致。而后者的同一用户可能看到该产品的不同版本,并且请求数的比例和分流比例一致,用户数的比例和可能和分流比例不一致。当然,A/B测试还存在其他的分流方式,一般情况下会采用按用户比例分流的方式,因为与按照用户请求分流的方式相比这样能够使同一个用户一直访问同一个版本,带来更好的用户体验。

[0005] 但是发明人在研究过程中发现,目前现有技术中,一般情况下各个用户的行为都会存在差异,而用户之间存在行为差异就可能会导致多版本测试结果的可信度和有效性较低,如果后续采用数据验证的方式对测试结果进行验证,例如采用基于统计学的验证,则验证的实现过程非常复杂。因此,现有技术并没有一种通用的方便易行的进行多版本测试的技术方案。

### 发明内容

[0006] 为了解决现有技术的问题,本申请提供一种通用的基于分流散列值的多版本测试方法,基于 cookie 和散列数就能够方便易行地实现多版本测试,适用于不同的业务系统。

[0007] 进一步的,本申请实施例通过分流优化还解决了用户差异性所带来的多版本测试结果的有效性和准确性的问题,提供一种简单易行的有效的验证方法。

[0008] 进一步的,本申请实施例还能通过分流优化解决同一产品在并行进行多个多版本测试时出现的分流互相干扰的问题。

[0009] 本申请还提供了一种多版本测试装置,用以保证上述方法在实际中的实现及应用。

[0010] 为了解决上述问题,本申请公开了一种多版本测试方法,包括:

[0011] 获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户;

[0012] 计算所述分流标签值的分流散列值;

[0013] 依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品进行多版本测试。

[0014] 优选的,所述获取访问当前产品的用户的分流标签值,具体包括:

[0015] 判断用户的 Web 请求中的访问数据 Cookie 中是否存在所述分流标签值,如果是,则直接从用户的 Cookie 中提取所述分流标签值,如果否,则依据预设的分流标签值的生成策略为所述用户生成分流标签值,并将所述分流标签值添加至所述用户的 Cookie 中。

[0016] 优选的,所述依据预设的分流标签值的生成策略为所述用户生成分流标签值,具体包括:

[0017] 获取所述用户的 IP 地址、第一次访问所述当前产品的时间和一随机数;

[0018] 依据所述 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值。

[0019] 优选的,在需要对访问所述当前产品的用户进行分流优化处理的情况下,则所述计算所述分流标签值的分流散列值之前,还包括:

[0020] 将所述分流标签值转换为哈希 hash 码值;

[0021] 计算所述 hash 码值对应的初始散列值;

[0022] 对所述初始散列值进行分流优化处理后得到分流散列值。

[0023] 优选的,所述对所述初始散列值进行分流优化处理后得到分流散列值,具体包括:

[0024] 对所述初始散列值按照预设的平移区间进行平移后得到平移的分流散列值;或者,

[0025] 对所述初始散列值按照预设的反转规则进行反转后得到反转的分流散列值;或者,

[0026] 对所述初始散列值与预设的乘积因子进行相乘后得到相乘的分流散列值;或者,

[0027] 对所述初始散列值按照预设的时间参数进行散列后得到散列的分流散列值。

[0028] 优选的,所述对所述当前产品进行多版本测试,具体为:

[0029] 依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。

[0030] 优选的,所述依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的

旧版本和新版本进行测试,具体包括:

[0031] 分别依据访问旧版本和新版本的用户的转化率,获取访问旧版本的所有用户的旧版本转化率,以及访问新版本的所有用户的新版本转化率;

[0032] 依据所述旧版本转化率和新版本转化率对所述旧版本和新版本进行测试。

[0033] 本申请公开了一种多版本测试装置,包括:

[0034] 获取模块,用于获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户;

[0035] 计算模块,用于计算所述分流标签值的分流散列值;

[0036] 分配模块,用于依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品进行多版本测试。

[0037] 优选的,所述获取模块具体包括:

[0038] 判断子模块,用于判断用户的 Web 请求中的访问数据 Cookie 中是否存在所述分流标签值;

[0039] 提取子模块,用于在所述判断子模块的结果为是的情况下,直接从用户的 Cookie 中提取所述分流标签值;

[0040] 生成子模块,用于在所述判断子模块的结果为否的情况下,依据预设的分流标签值的生成策略为所述用户生成分流标签值;

[0041] 添加子模块,用于将所述生成子模块生成的分流标签值添加至所述用户的 Cookie 中。

[0042] 优选的,所述生成子模块具体包括:

[0043] 获取参数子模块,用于获取所述用户的 IP 地址、第一次访问所述当前产品的时间和一随机数;

[0044] 组合子模块,用于依据所述 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值。

[0045] 优选的,所述计算模块包括:

[0046] 转换子模块,用于将转换为哈希 hash 码值;

[0047] 计算子模块,用于计算所述 hash 码值对应的初始散列值;

[0048] 优化子模块,用于对所述初始散列值进行分流优化处理后得到分流散列值。

[0049] 优选的,所述优化子模块,具体包括:

[0050] 平移子模块,用于对所述初始散列值按照预设的平移区间进行平移后得到平移的分流散列值;或者,

[0051] 反转子模块,用于对所述初始散列值按照预设的反转规则进行反转后得到反转的分流散列值;或者,

[0052] 相乘子模块,用于对所述初始散列值与预设的乘积因子进行相乘后得到相乘的分流散列值;或者,

[0053] 散列子模块,用于对所述初始散列值按照预设的时间参数进行散列后得到散列的分流散列值。

[0054] 优选的,还包括:

[0055] 测试模块,用于依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。

[0056] 优选的,所述测试模块包括:

[0057] 获取转化率子模块,用于分别依据访问旧版本和新版本的用户的转化率,获取访问旧版本的所有用户的旧版本转化率,以及访问新版本的所有用户的新版本转化率;

[0058] 计算子模块,用于依据所述旧版本转化率和新版本转化率对所述旧版本和新版本进行测试。

[0059] 与现有技术相比,本申请包括以下优点:

[0060] 在本申请中,在获取到访问当前产品的用户的分流标签值之后,计算该分流标签值的分流散列值,再依据预设的当前产品的旧版本和新版本的配置比和分流散列值,为所述用户分配所述当前产品的旧版本或新版本,最后依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。本申请实施例通过对各个用户的分流标签值进行散列从而得到分流散列值,从而提供了一种通用的基于分流散列值实现的方便的多版本测试方法。

[0061] 进一步的,通过对分流标签值的优化过程,可以使得原本应该分配给旧版本的用户可能因为散列值的变化而分配给新版本,而原本应该分配给新版本的用户则可能因为散列值的变化而分配给旧版本,就能够通过上述有效的手段降低用户之间的行为差异给多版本测试结果带来的影响,提高多版本测试的有效性和可信度,提供有效的验证方法。进一步的,通过对分流标签值的优化过程,还能解决同一产品在并行进行多个多版本测试时出现的分流互相干扰的问题。

[0062] 当然,实施本申请的任一产品并不一定需要同时达到以上所述的所有优点。

## 附图说明

[0063] 为了更清楚地说明本申请实施例中的技术方案,下面将对实施例描述中所需要使用的附图作简单地介绍,显而易见地,下面描述中的附图仅仅是本申请的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动性的前提下,还可以根据这些附图获得其他的附图。

[0064] 图 1 是本申请的一种多版本测试方法实施例 1 的流程图;

[0065] 图 2 是本申请方法实施例 1 中步骤 101 的流程图;

[0066] 图 3 是本申请方法实施例 1 中步骤 203 的流程图;

[0067] 图 4 是本申请方法实施例 1 的优选流程图;

[0068] 图 5 是本申请方法实施例 1 中步骤 401 的流程图;

[0069] 图 6 本申请的一种多版本测试方法实施例 2 的流程图;

[0070] 图 7 为本申请的一种多版本测试装置实施例 1 的结构框图;

[0071] 图 8 为本申请的装置实施例 1 中获取模块 701 的结构框图;

[0072] 图 9 为本申请的装置实施例 1 中生成子模块 803 的结构框图;

[0073] 图 10 是本申请装置实施例 1 的优选结构框图;

[0074] 图 11 是本申请的装置实施例 1 中测试模块 1001 的结构框图;

[0075] 图 12 是本申请的一种多版本测试装置实施例 2 的结构框图。

## 具体实施方式

[0076] 下面将结合本申请实施例中的附图,对本申请实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅仅是本申请一部分实施例,而不是全部的实施例。基于本申请中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本申请保护的范围。

[0077] 本申请可用于众多通用或专用的计算装置环境或配置中。例如:个人计算机、服务器计算机、手持设备或便携式设备、平板型设备、多处理器装置、包括以上任何装置或设备的分布式计算环境等等。

[0078] 本申请可以在由计算机执行的计算机可执行指令的一般上下文中描述,例如程序模块。一般地,程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等等。也可以在分布式计算环境中实践本申请,在这些分布式计算环境中,由通过通信网络而被连接的远程处理设备来执行任务。在分布式计算环境中,程序模块可以位于包括存储设备在内的本地和远程计算机存储介质中。

[0079] 本申请的主要思想之一可以包括,在获取到访问当前产品的用户的分流标签值之后,计算该分流标签值的分流散列值,再依据预设的当前产品的旧版本和新版本的配置比和分流散列值,为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品的旧版本和新版本进行测试。其中,本申请所述的产品可以包括网站服务器通过网络传输至用户客户端的网页信息。通过对各个用户的分流标签值进行散列从而得到分流散列值,从而提供一种基于分流散列值实现的方便的多版本测试方法。

[0080] 参考图 1,示出了本申请一种多版本测试方法实施例 1 的流程图,可以包括以下步骤:

[0081] 步骤 101:获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户。

[0082] 本申请在实际中应用于多版本测试系统中,首先需要获取到所有访问当前产品的用户的分流标签值,该分流标签值可以用于唯一标识访问当前产品的当前用户。例如,用户 A 触发了访问当前产品的请求,则接收到该用户 A 的访问请求之后,先不给该用户 A 分配其访问该当前产品的旧版本或者新版本,而是先基于该用户 A 的 IP( 网络之间互连的协议, Internet Protocol) 地址、用户第一次的访问时间和随机数等组成该分流标签值,这样能确保该分流标签值所标示用户的唯一性。

[0083] 参考图 2 所示,所述步骤 101 在实现时具体可以包括:

[0084] 步骤 201:判断用户的 Web 请求中的访问数据 Cookie 中是否存在所述分流标签值,如果是,则进入步骤 202,如果否,则进入步骤 203。

[0085] 在本实施例中,如果多版本测试系统接收到了一个用户的 Web 请求,就先读取该 Web 请求中的访问数据 (Cookie) 中是否已经存在分流标签值,如果已经存在,说明该用户已经不是第一次访问当前产品,此时直接从 Cookie 中提取出该分流标签值,如果没有,说明用户是第一次访问当前产品,那么就按照预先设定好的分流标签值的生成策略为该用户生成其分流标签值。

[0086] 步骤 202:直接从用户的 Cookie 中提取所述分流标签值。

[0087] 在本步骤中,如果该用户的 Cookie 中已经存在分流标签值,则直接从用户的Cookie 中提取所述分流标签值。

[0088] 步骤 203 :依据预设的分流标签值的生成策略为所述用户生成分流标签值。

[0089] 在本步骤中,如果该用户的 Cookie 中不存在分流标签值,则根据预先设定好的分流标签值生成策略生成一个唯一标示该用户的字符串,即是分流标签值。

[0090] 其中,参考图 3 所示,在具体实现时步骤 203 又可以包括以下步骤:

[0091] 步骤 301 :获取所述用户的 IP 地址、第一次访问所述当前产品的时间和一随机数。

[0092] 在本实施例中,用户的分流标签值与用户的 IP 地址、第一次访问所述当前产品的时间和随机数有关系,当然,也可以根据实际场景或者不同的用户需求而设置不同的分流标签值生成策略。其中,用户的 IP 地址和第一次访问当前产品的时间都是多版本测试系统直接可以从 cookie 中得到的,而随机数则可以随机生成。

[0093] 步骤 302 :依据所述 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值。

[0094] 再将步骤 301 获得的 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值即可。例如,假设用户的 IP 地址为 121.0.29.199,而第一次访问该当前产品的时间为“1335163135361”,获取到的一随机数为 3,则可以采用将“.”作为分隔符直接将三个数值组合为一起的方式,则该用户的分流标签值就是:121.0.29.199.1335163135361.3。通过将 IP 地址、第一次访问当前产品的时间和随机数三个参数的叠加,可以使本申请实施例中的分流标签值很好地兼顾了唯一性和散列性。

[0095] 返回图 2,进入步骤 204 :将所述分流标签值添加至所述用户的 Cookie 中。

[0096] 在本步骤中将步骤 203 中得到的分流标签值在多版本测试系统响应用户请求时,写入该用户的 cookie 中,这样下一次该用户再访问该当前产品将直接可以通过步骤 202 获得。

[0097] 接着返回图 1,进入步骤 102 :计算所述分流标签值的分流散列值。

[0098] 在得到用户的分流标签值之后,需要先将该分流标签值转换为哈希 (hash) 码值(其中,如果转换的 hash 码值为负值,需要对其取绝对值以得到为正数的 hash 码值),再对该正 hash 码值进行散列,最终得到该分流标签值对应的分流散列值。分流散列值表示该用户命中当前产品的旧版本还是新版本。例如,假设预先配置好的旧版本和新版本的比例为 50% :50%,则表示 50% 的用户会访问到旧版本,而另外 50% 的用户则会访问到新版本,在这种情况下,分流散列值可以为一个 0 到 99 的数字。在实际中,计算分流散列值可以将该用户的 cookie 值转换为正的 hash 码值,然后再对该正 hash 码值对 100 取余即可将分流标签值散列为 0 ~ 99 的分流散列值。当然,也可以根据需求对 10 取余,则可将分流标签值散列为 0 ~ 9 的分流散列值,等等。

[0099] 步骤 103 :依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品进行多版本测试。

[0100] 如果用户进行一个基于“按用户比例分流”的多版本测试,而预设的当前产品的旧版本和新版本的分流配置比为 20% :80%,则表示将会分流 20% 的用户给旧版本 A,而另外 80% 的用户分流给新版本 B。在实际应用中,可以提供一个图形化界面,用于输入或者显示

当前产品的配置比。

[0101] 在实际应用中,假设预先配置好的旧版本和新版本的比例为 20%:80%,如果用户甲散列的数字为 18,那么这个用户甲始终落在 0 ~ 19 的区间,那么本步骤中为用户甲分配的则是当前产品的旧版本 A,因此该用户甲在访问该当前产品的时候始终看到的是旧版本 A。而另一个用户乙散列的数字为 59,那么这个用户始终落在 20 ~ 99 的区间,那么在本步骤中为用户乙分配的则是当前产品的新版本 B,该用户乙在访问该当前产品的时候始终看到的是新版本 B。

[0102] 需要说明的是,因为在预先设置旧版本和新版本的配置比的时候,可以有很多种方式,例如按照用户个数进行配置,即是配置多少比例的用户分配给旧版本 A,而多少比例的用户分配给新版本 B。而如果按照用户请求数对配置比进行设置的时候,则配置比就表示对于当前产品的若干次请求中,有多少个请求会分配给旧版本 A,而多少个请求会分配给新版本 B。当然,还有其他的配置方式,例如按照业务逻辑进行用户的分流,那么可能会出现该当前产品的正式会员会访问到旧版本 A 而临时用户会访问到新版本 B 的情况;例如按地理位置进行用户的分流,那可能会出现本地用户会访问到旧版本 A 而外地用户会访问到新版本 B 的情况;再例如还可以按黑白名单进行用户分流等。

[0103] 其中,为用户分配当前产品的旧版本或新版本之后,就可以根据分别访问旧版本和新版本的用户行为来对所述当前产品进行旧版本和新版本的多版本测试。例如,可以分别访问旧版本和新版本的用户的转化率,对当前产品进行多版本测试;也可以通过监控用户在当前产品的旧版本和新版本网页上进行鼠标点击的次数或者时间,来对当前产品进行多版本测试;还可以通过记录用户浏览当前产品的旧版本或者新版本网页的时间等信息,来对当前产品进行多版本测试。

[0104] 本申请实施例通过对各个用户的分流标签值进行散列从而得到分流散列值,基于 cookie 和散列数就能够方便易行地实现本技术方案,从而提供了一种通用的基于分流散列值实现的方便的多版本测试方法。

[0105] 优选的,本申请实施例在采用转化率进行多版本测试的时候,参考图 4 所示,在步骤 103 后还可以包括:

[0106] 步骤 401:依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。

[0107] 在结合步骤 103 的分流结果可以得知,对于当前产品哪些用户会访问到旧版本而哪些用户会访问到新版本,而结合访问旧版本的所有用户的转化率,以及访问新版本的所有用户的转化率,就可以对当前产品的旧版本和新版本进行测试,即是通过对旧版本和新版本的转化率来分析当前产品的旧版本和新版本的效果和商业价值等。

[0108] 具体的,参考图 5 所示,所述步骤 401 在实现时具体可以包括:

[0109] 步骤 501:分别依据访问旧版本和新版本的用户的转化率,获取访问旧版本的所有用户的旧版本转化率,以及访问新版本的所有用户的新版本转化率。

[0110] 在本实施例中,因为多版本测试系统会记录每一次用户对当前产品的旧版本或者新版本的访问,那么就可以通过多版本测试系统记录的用户的访问情况来获取到旧版本转化率和新版本转化率。其中,一个网站产品的转化率的计算方式为:进行了相应的动作的访问量 / 总访问量。对于本申请实施例来说,旧版本转化率和新版本转化率即是旧版本的

网站转化率和新版本的网站转化率。网站转化率可以分很多种,对于一个注册页面的网站来讲,其网站转化率可以表示为注册成功率,即是用户访问了该注册页面上有多少用户成功注册。再例如,对于一个电子商务网站的产品详情页面来说,其网站转化率可以表示为下单转化率,即是用户在浏览了该产品详情页面之后,有多少用户触发点击了下单按钮。举例来讲,对于下单转化率的获取来讲,可以统计一共有多少次用户请求(假设有 1000 次用户请求)来请求浏览某个产品的详情页面,而在这么多用户请求对应的总的浏览次数中有多少次用户请求对应(假设有 35 次用户请求)进行了下单,那么下单转化率就是  $35/1000 = 3.5\%$ 。

- [0111] 当然,网站转化率在实际应用中还包括很多种,在此就不再一一举例。
- [0112] 步骤 502 :依据所述旧版本转化率和新版本转化率对所述旧版本和新版本进行测试。
- [0113] 再根据步骤 501 中得到的旧版本整体上的旧版本转化率,以及新版本整体上的新版本转化率,将两个版本的转换率进行比对,从而测试出旧版本和新版本的效果好坏,其中,版本转化率可以表示出该版本的效果优劣。例如,新版本转化率高于旧版本转化率,此时可以认为新版本的效果或者商业价值都高于新版本。一般情况下,因为存在用户之间的个体差异,新版本转化率比旧版本转化率高于一定的数值,那么可以认为该效果测试结果是比较可信的结果。当然不同的产品有不同的特性,也存在不同的经验值,具体对于某个产品其新版本转化率要比旧版本转化率高出多少,才认为效果测试结果可信,可以参考产品转化率提高幅度的历史经验,以作为本领域技术人员一个基本的判断和预期。
- [0114] 当然,根据测试的网站产品的类型不同,例如对于注册页面可以比对其注册转化率,而对于产品详情页面则可以比对下单转化率,其所需比对的网站转化率的类型也是不同的,本领域技术人员在进行新版本和旧版本的效果测试的时候,可以根据实际业务场景或者用户需求选择合适的网站转化率进行比对。
- [0115] 在实际应用中,进行多版本测试时,所得到的测试结果未必一定是新版本的版本转化率高于旧版本的,如果新版本的效果优势不是那么的明显,例如可能只优于旧版本 4% 左右,那么可以暂时认为这个测试的可信度需要论证,因为该测试结果有可能是由于样本个体的差异化所造成的。在这种情况下,可以对用户的分流标签值进行优化,从而对测试结果也起到验证的作用。参考图 6,示出了本申请一种多版本测试方法实施例 2 的流程图,可以包括以下步骤 :
- [0116] 步骤 601 :获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户。
- [0117] 本步骤的实现可以参考实施例 1 的内容,在此不再赘述。
- [0118] 步骤 602 :将所述分流标签值转换为哈希 hash 码值。
- [0119] 在本实施例中,可以采用实施例 1 的方式将步骤 501 获取到的分流标签值转换为正的 hash 码值。
- [0120] 步骤 603 :计算所述 hash 码值对应的初始散列值。
- [0121] 再计算 hash 码值对应的初始散列值。
- [0122] 步骤 604 :对所述初始散列值进行分流优化处理后得到分流散列值。
- [0123] 本实施例与实施例 1 的不同之处在于,本实施例中还对初始散列值进行分流优化

处理从而得到优化后的分流散列值。

[0124] 具体在步骤 604 进行优化时,可以有以下四种方式,第一种方式为:对所述初始散列值按照预设的平移区间进行平移后得到平移的分流散列值。

[0125] 在应用该种优化方式的时候,对于得到的初始散列值需要按照预设的平移区间对其进行平移,以起到变更分配给旧版本和新版本的用户区间的目的。例如,预设的新版本和旧版本的配置比为:20 : 80,预设的平移区间为 50。因此,虽然预设的配置比也是分流 20% 的用户给新版本,但是因为附加了预设的平移区间为 50,那么实际情况变成:用户甲的初始散列值虽然仍旧为 18,如果不对其进行平移,此种情况下会分配给该用户新版本;但是平移预设的平移区间之后就变成 68,最终会分配给该用户旧版本。再例如,用户乙的初始散列值为 59,如果不对其进行平移则此种情况下会分配给该用户旧版本,而如果按照优化的第一种方式,对其平移 50 后得到 109,即其最终的分流散列值是 9,那么在优化的情况下将会分配给该用户新版本。

[0126] 当然,在实际应用中,平移区间可以根据实际场景、用户需求或者新旧版本的配置比等参数而自主设置其数值的。一般情况下,在前一次测试结果可信度值得商榷的情况下,采用第一种优化方式可以实现既能保证多版本测试的随机性,又能采用一批相对新的用户测试一次新版本,从而对多版本测试是否受到个体差异性的影响进行验证。如果基于平移因子的测试结果仍旧是相同的效果趋势,那么可信度进一步提高。反之,则证明前一次多版本测试的结果不准确。如果有需要的话,也可以多平移几次,并且平移的跨度是可以灵活配置的。

[0127] 第二种方式为:对所述初始散列值按照预设的反转规则进行反转后得到反转的分裂散列值。

[0128] 对初始散列值进行优化的第二种方式为反转,此种方式一般适用于只有新旧版本各有一个的场景。例如,将得到的分流标签值落入旧版本的用户全部分配给新版本,而将得到的分流标签值落入新版本的用户全部分配给旧版本,这样就能起到对旧版本和新版本的用户区间进行对换的目的。在进行反转优化之后,如果测试结果仍旧是新版本好于老版本,一般情况下就可以认为本次优化过程是有效的。其中,反转优化是平移优化的一种特例,通常是用在只有旧版本和新版本共两个并需要快速验证多版本测试结果的情况,因为反转的优化方式会影响所有的产品用户,所以在使用第二种方式需要根据实际情况进行适当的选择。

[0129] 第三种方式为:对所述初始散列值与预设的乘积因子进行相乘后得到相乘的分流散列值。

[0130] 对初始散列值进行优化的第三种方式为,将步骤 503 得到的初始散列值与预设的乘积因子进行相乘从而得到相乘的分流散列值,将该乘积作为最终的分流散列值。例如,对于步骤 503 得到的初始散列值都乘以数值“3”,这样也能够起到通过将初始散列值改变成最终的分流散列值进而改变分配给旧版本或者新版本的用户的目的。其中,需要说明的是,预设的乘积因子的数值也是不定的,可以根据实际情况进行调整,只要是一个指定的数字即可。一般情况下,第三种优化方式主要用于解决同产品多个并行多版本测试分流互相干扰的问题。

[0131] 第四种方式为:对所述初始散列值按照预设的时间参数进行散列后得到散列的分

流散列值。

[0132] 对初始散列值进行优化的第四种方式为,将步骤 503 中得到的初始散列值按照预设的时间参数进行散列后得到散列的分流散列值。在本步骤中,预设的时间参数也是一个数值,但是与第三种方式不同的是,只要乘积因子设置成功之后,一直都是同一个指定的数字,但是如果将时间参数预设为当日日期,那么在不同的日期进行的多版本测试中,初始散列值所乘的数字就与当日日期相同,这是一个变量。可以理解的是,按时间参数的优化方式可以让每天测试的用户区间重新打散,因此能够用来作为一个网站产品受用户群行为影响的波动情况的分析依据。其中,时间参数不仅仅可以采用“天”为单位进行优化,也可以采用“小时”或者“星期”等单位进行优化,例如,如果采用“小时”,那么在不同的小时内初始散列值所乘的数字就与当时的小时相同,以此类推,则采用“星期”,那么在不同的星期内初始散列值所乘的数字就与当时的星期相同。

[0133] 需要说明的是,上述的四种方式在优化过程中可以任意选择一种方式进行。而如果进行任意一种优化之后,旧版本和新版本的测试结果的可信度仍然不高,那么可以多进行几次优化,并且其中的平移区间、乘积因子和时间参数等,都是可以灵活配置的。

[0134] 其中,本实施例中的步骤 602 ~ 步骤 604 为实施例 1 中步骤 103 的优选实现方式。

[0135] 步骤 605 :依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本。

[0136] 步骤 606 :依据分别访问旧版本和新版本的用户的转化率,对所述当前产品的旧版本和新版本进行测试。

[0137] 采用本发明实施例,通过对各个用户的分流标签值进行散列从而得到分流散列值,可以使得原本应该分配给旧版本的用户可能因为散列值的变化而分配给新版本,而原本应该分配给新版本的用户则可能因为散列值的变化而分配给旧版本,从而能够降低用户之间的行为差异给测试结果带来的影响,验证多版本测试的可信度和有效性。

[0138] 在实际应用中,因为多版本测试过程中,针对同一个用户请求,可能涉及不同的测试内容,例如可能会先后进入两个多版本测试的流程中,那么一般为了减少时间可以采用并行测试的方式。例如,假设某网站产品同时进行两个并行的多版本测试,采用的是最为常用的“按用户比例分流”的方式。那么对于一个用户访问来讲,将会先进入第一个多版本测试流程(假设其旧版本为 sellpopA, 新版本为 sellpopB), 随后进行第二个多版本测试流程。

[0139] 那么对于需要访问网站的用户来说,假设先进行 sellpop 测试,分配 50% 的用户访问 sellpopA, 另外 50% 的用户访问 sellpopB, 即是 sellpopA : sellpopB 为 :50% :50%。接着在 sellpopB 的用户分支里再进行第二个多版本 popp4p 测试流程,从中分配 20% 的用户访问 popp4pA, 分配另外 20% 的用户访问 popp4pB, 剩余 60% 的用户访问 popp4pC, 即是 popp4pA : popp4pB : popp4pC 的比例为 :20% :20% :60%。在实际运行的过程中, sellpop 测试的实际用户分流比例符合预设的配置比,而 popp4p 测试的实际用户分流比例则不正常, popp4pA 和 popp4pB 实际上没有用户访问,和预期的各有 20% 的用户不符。这就说明在实际中并行多版本测试过程中出现了分流互相干扰的情况。

[0140] 下面将举例说明本实施例中的优化过程是如何克服并行测试中分流互相干扰的问题。

[0141] 分析前述例子中用户分流不正常的原因,在这个例子中,进入网站产品的用户,先进行 sellpop 测试,会分流 50% 的用户访问 sellpopA,另外 50% 的用户访问 sellpopB,接着在 sellpopB 的分支里再进行 popp4p 测试,再从中分流 20% 的用户访问 popp4pA,分 20% 的用户访问 popp4pB,其他 60% 的用户则会访问到 popp4pC。其中,因为 sellpop 测试和 popp4p 测试是在同一时间段内做的多版本测试,所以用户访问网站产品的请求会先经过 sellpop 测试,再经过 popp4p 测试。

[0142] 其中,假设对于用户甲,其分流散列值为 25,属于 0 到 50 的这个区间,那么该用户在 sellpop 测试时就会分配其访问 sellpopA,也即是用户甲根本不会进入第二个多版本测试 (popp4p)。而如果另一个用户乙的分流散列值为 69,属于 50 到 99 的区间,那么该用户乙在 sellpop 测试时就会分配其访问 sellpopB。然后再进入第二个多版本测试 pop4p 时,该用户乙就会分配其访问 pop4pC。可见这样就会造成能够进入 popp4p 测试的用户的散列值必定在 50 ~ 99 区间,也就不属于 0 ~ 19 以及 20 ~ 39 的区间,所以就没有机会访问 pop4pA 版本和 pop4pB 版本。

[0143] 在这种情况下,采用本实施例中优化方式可以解决此问题。在串行化进入的第二个多版本测试 popp4p 中采用第三种方式进行优化,假设预设的乘积因子为 15,那么新版本与旧版本的比例虽然仍是 20 : 80,但是用户的分流标签值都会与一个预设的乘积因子 15 进行相乘。在这种情况下,虽然能够进入 popp4p 测试的,用户的分流标签值的分流散列值一定是在 50 到 99 范围内,假设初始分流标签值为 169(分流散列值为 69,在 50 到 99 范围内),将分流标签值与预设的乘积因子相乘即是  $169 * 15 = 2535$ ,再对 100 取余计算分流散列值,结果为 35,这就说明对应的用户在进入 popp4p 测试时其分流散列值在 20 ~ 39 区间内而会分配 popp4pB 给其访问。

[0144] 假设用户的初始分流标签值为其他数值,例如 169、182、191 和 199,

[0145] 那么优化后的分流散列值与其对应关系如表 1 所示:

[0146] 表 1

[0147]

初 始 分 流 标 签 值 ( sellpop 测 试 )	未 优 化 的 分 流 散 列 值 ( popp4p 测 试 )	采 用 预 设 的 乘 积 因 子 优 化 后 的 散 列 值 ( popp4p 测 试 )
169	69	35
182	82	30
191	91	65
199	99	85

[0148] 可以从表 1 中看到,本来其未优化时的分流散列值属于 50 到 99 的区间的用户,在优化之后其分流散列值重新变为 0 到 99 的区间,这样 pop4pA 版本和 pop4pB 版本都有可能被用户访问到,也就避免了多版本并行测试中分流互相干扰的现象。

[0149] 可见本申请实施例通过对各个用户的分流标签值的初始散列进行优化再得到分流散列值,可以使得原本应该分配给旧版本的用户可能因为散列值的变化而分配给新版本,而原本应该分配给新版本的用户则可能因为散列值的变化而分配给旧版本,从而与现有技术相比,解决用户差异性所带来的多版本测试结果的有效性和准确性的问题,提供有效的验证方法,还能解决同一产品在并行进行多个多版本测试时出现的分流互相干扰的问题。

[0150] 对于前述的各方法实施例,为了简单描述,故将其都表述为一系列的动作组合,但是本领域技术人员应该知悉,本申请并不受所描述的动作顺序的限制,因为依据本申请,某些步骤可以采用其他顺序或者同时进行。其次,本领域技术人员也应该知悉,说明书中所描述的实施例均属于优选实施例,所涉及的动作和模块并不一定是本申请所必须的。

[0151] 与上述本申请一种多版本测试方法实施例 1 所提供的方法相对应,参见图 7,本申请还提供了一种多版本测试装置实施例 1,在本实施例中,该装置可以包括:

[0152] 获取模块 701,用于获取访问当前产品的用户的分流标签值,所述分流标签值用于唯一标识用户。

[0153] 在具体实现时,参考图 8 所示,所述获取模块 701 具体可以包括:

[0154] 判断子模块 801,用于判断用户的访问数据 Cookie 中是否存在所述分流标签值;

[0155] 提取子模块 802,用于在所述判断子模块的结果为是的情况下,直接从用户的 Cookie 中提取所述分流标签值;

[0156] 生成子模块 803,用于在所述判断子模块的结果为否的情况下,依据预设的分流标签值的生成策略为所述用户生成分流标签值;

[0157] 添加子模块 804,用于将所述生成子模块生成的分流标签值添加至所述用户的 Cookie 中。

[0158] 在具体实现时,参考图 9 所示,所述生成子模块 803 具体可以包括:

[0159] 获取参数子模块 901,用于获取所述用户的 IP 地址、第一次访问所述当前产品的时间和一随机数;

[0160] 组合子模块 902,用于依据所述 IP 地址、第一次访问所述当前产品的时间和一随机数组合为所述分流标签值。

[0161] 计算模块 702,用于计算所述分流标签值的分流散列值。

[0162] 分配模块 703,用于依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本,以便对所述当前产品进行多版本测试。

[0163] 本申请实施例的装置通过对各个用户的分流标签值进行散列从而得到分流散列值,基于 cookie 和散列数就能够方便易行地实现本技术方案,从而提供了一种通用的基于分流散列值实现的方便的多版本测试方法。

[0164] 优选的,参考图 10 所示,本申请实施例在采用转化率对当前产品进行多版本测试的时候,除了图 7 所示的模块之外,还可以包括:

[0165] 测试模块 1001, 用于依据分别访问旧版本和新版本的用户的转化率, 对所述当前产品的旧版本和新版本进行测试。

[0166] 在具体实现时, 参考图 11 所示, 所述测试模块 1001 具体可以包括:

[0167] 获取转化率子模块 1101, 用于分别依据访问旧版本和新版本的用户的转化率, 获取访问旧版本的所有用户的旧版本转化率, 以及访问新版本的所有用户的新版本转化率;

[0168] 测试子模块 1102, 用于依据所述旧版本转化率和新版本转化率对所述旧版本和新版本进行测试。

[0169] 与上述本申请一种多版本测试方法实施例 2 所提供的方法相对应, 参见图 12, 本申请还提供了一种多版本测试装置实施例 2, 在本实施例中, 该装置可以包括:

[0170] 获取模块 701, 用于获取访问当前产品的用户的分流标签值, 所述分流标签值用于唯一标识用户。

[0171] 转换子模块 1201, 用于将转换为哈希 hash 码值。

[0172] 计算子模块 1202, 用于计算所述 hash 码值对应的初始散列值。

[0173] 优化子模块 1203, 用于对所述初始散列值进行分流优化处理后得到分流散列值。

[0174] 其中, 所述优化子模块 1203, 具体可以包括:

[0175] 平移子模块, 用于对所述初始散列值按照预设的平移区间进行平移后得到平移的分流散列值; 或者,

[0176] 反转子模块, 用于对所述初始散列值按照预设的反转规则进行反转后得到反转的分流散列值; 或者,

[0177] 相乘子模块, 用于对所述初始散列值与预设的乘积因子进行相乘后得到相乘的分流散列值; 或者,

[0178] 散列子模块, 用于对所述初始散列值按照预设的时间参数进行散列后得到散列的分流散列值。

[0179] 分配模块 703, 用于依据预设的所述当前产品的旧版本和新版本的配置比和所述分流散列值为所述用户分配所述当前产品的旧版本或新版本。

[0180] 测试模块 01001, 用于依据分别访问旧版本和新版本的用户的转化率, 对所述当前产品的旧版本和新版本进行测试。

[0181] 采用本发明实施例, 通过对各个用户的分流标签值的初始散列值进行优化再得到分流散列值, 可以使得原本应该分配给旧版本的用户可能因为散列值的变化而分配给新版本, 而原本应该分配给新版本的用户则可能因为散列值的变化而分配给旧版本, 从而能够降低用户之间的行为差异给测试结果带来的影响, 验证多版本测试的可信度和有效性。进一步, 本申请中的装置还能解决还能解决同一产品在并行进行多个多版本测试时出现的分流互相干扰的问题。

[0182] 需要说明的是, 本说明书中的各个实施例均采用递进的方式描述, 每个实施例重点说明的都是与其他实施例的不同之处, 各个实施例之间相同相似的部分互相参见即可。对于装置类实施例而言, 由于其与方法实施例基本相似, 所以描述的比较简单, 相关之处参见方法实施例的部分说明即可。

[0183] 最后, 还需要说明的是, 在本文中, 诸如第一和第二等之类的关系术语仅仅用来将一个实体或者操作与另一个实体或操作区分开来, 而不一定要求或者暗示这些实体或操作

之间存在任何这种实际的关系或者顺序。而且,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、物品或者设备不仅包括那些要素,而且还包括没有明确列出的其他要素,或者是还包括为这种过程、方法、物品或者设备所固有的要素。在没有更多限制的情况下,由语句“包括一个……”限定的要素,并不排除在包括所述要素的过程、方法、物品或者设备中还存在另外的相同要素。

[0184] 以上对本申请所提供的一种多版本测试方法和装置进行了详细介绍,本文中应用了具体个例对本申请的原理及实施方式进行了阐述,以上实施例的说明只是用于帮助理解本申请的方法及其核心思想;同时,对于本领域的一般技术人员,依据本申请的思想,在具体实施方式及应用范围上均会有改变之处,综上所述,本说明书内容不应理解为对本申请的限制。

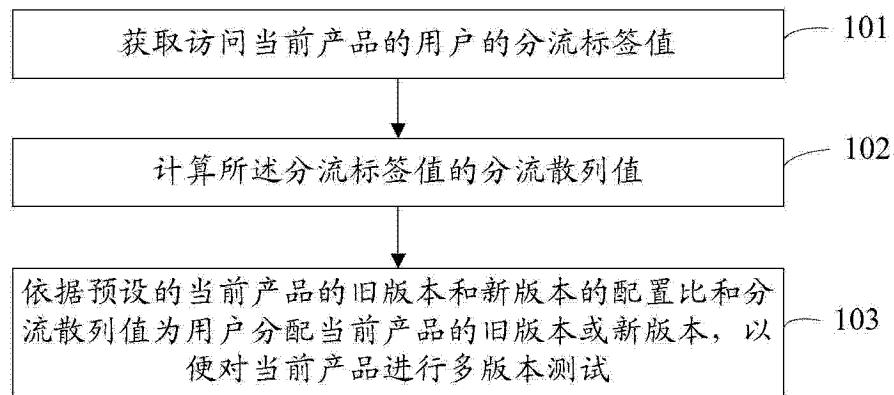


图 1

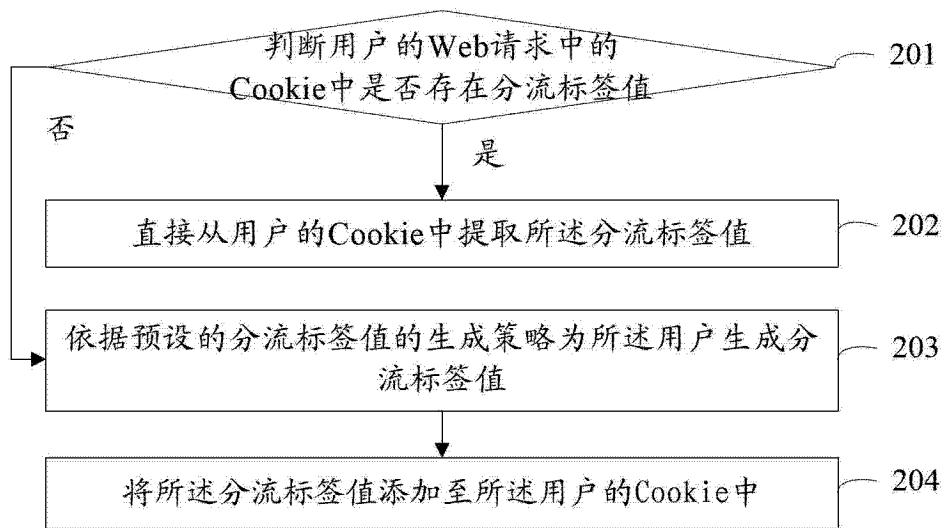


图 2

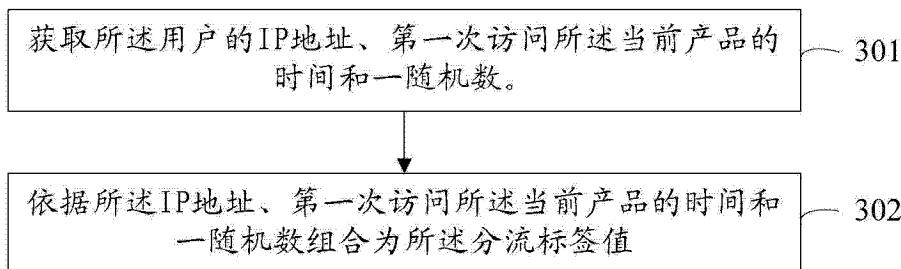


图 3

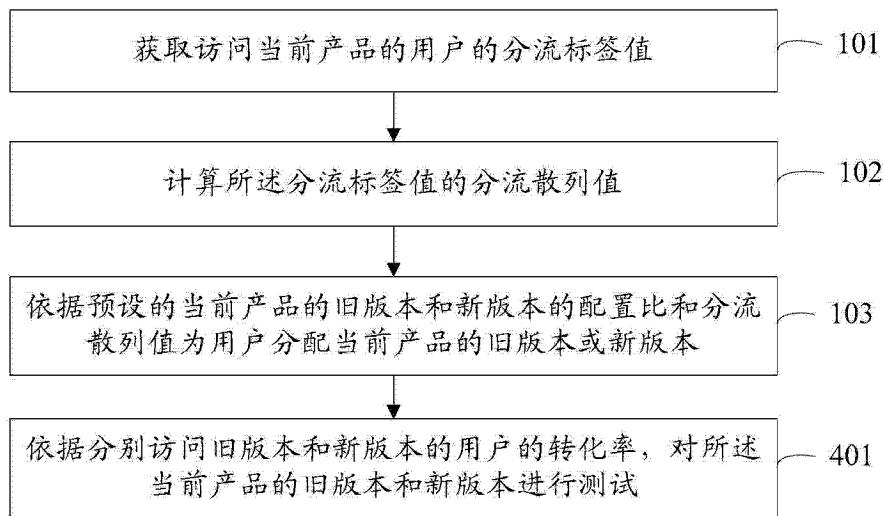


图 4

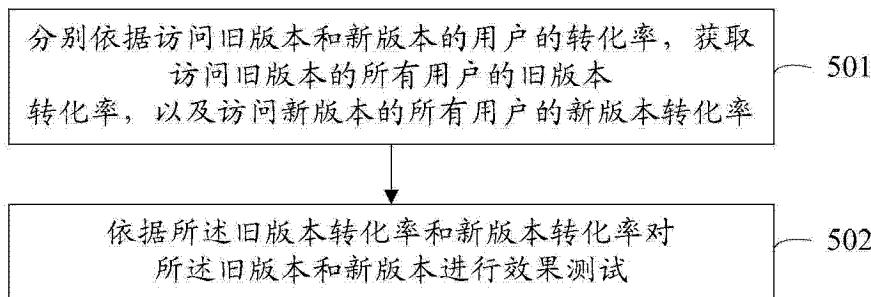


图 5

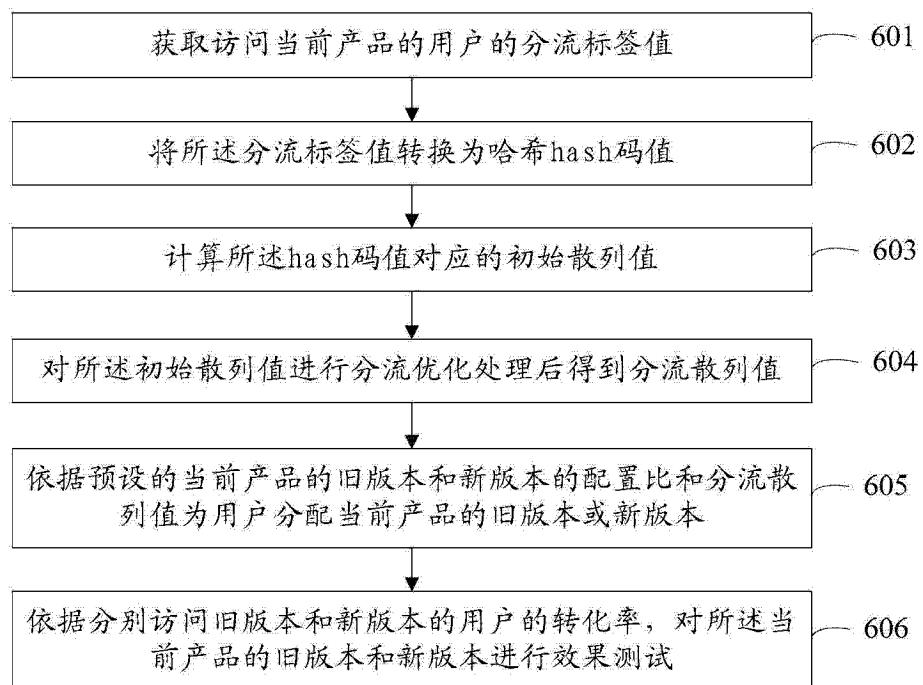


图 6

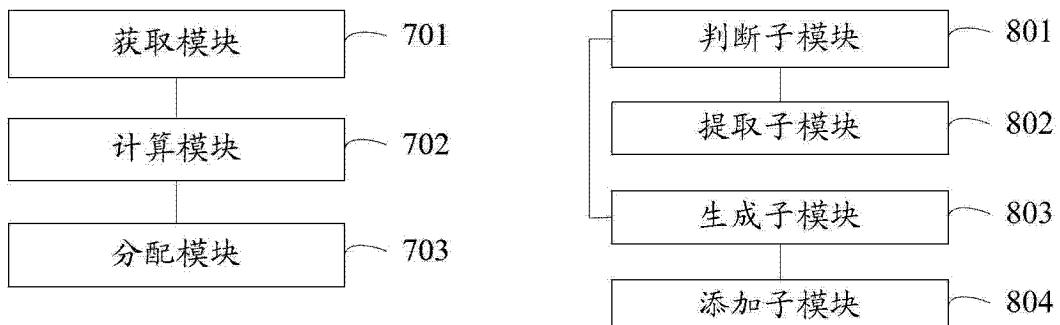


图 7

图 8

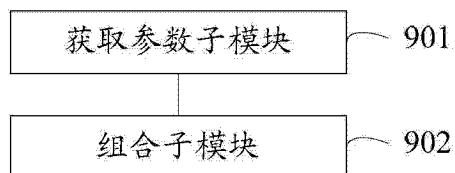


图 9

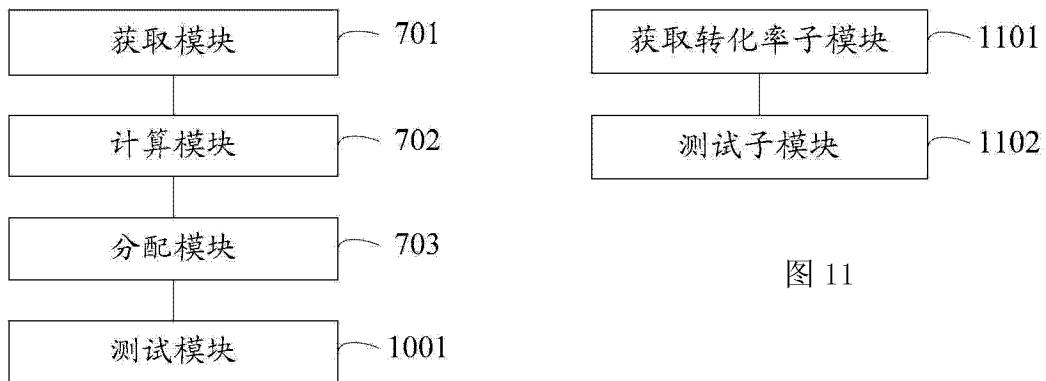


图 10

图 11

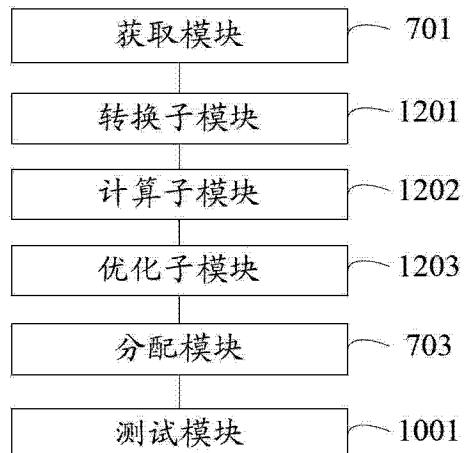


图 12