效果平台细节设计

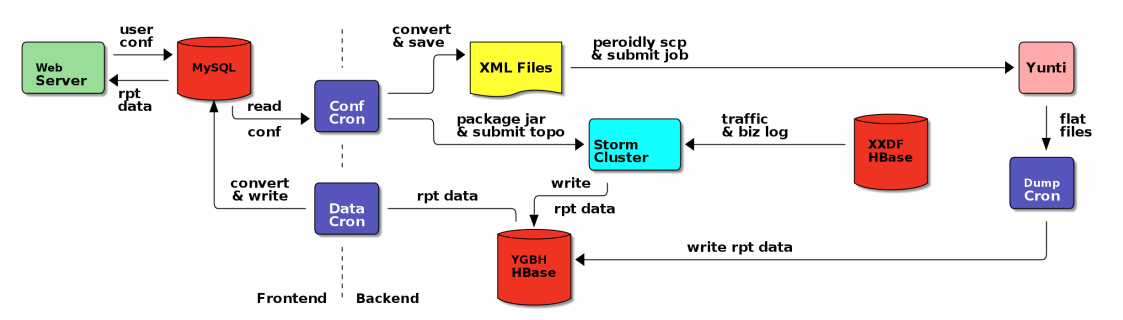
清无

2012-06-11

Contents

1 系统架构与部署

1.1 系统整体数据流程



其中:

- **前端使用 MySQL 保存用户界面输入的简单配置数据及后台计算的报表结果数据**
- **Conf Cron 定期将 MySQL 中人工审核通过的简单配置数据转换扩充为完整的计算方案配置文件,写为供离线计算使用的 XML 文件并将其同实时计算逻辑打包提交到 Storm 集群进行实时计算**
- **离线计算部分定期将 XML 配置文件 scp 到中转机并提交为云梯任务,其每天计算完毕后产出若干 flat file,由 Dump Cron 定时转换为合适的格式写入月光宝盒的 HBase 表**
- **实时计算拓扑读取吸星大法写入 HBase 的标准化日志进行计算,并将结果实时写入月光宝盒的 HBase 表**
- **Data Cron 定期将月光宝盒 HBase 中的报表数据转换为适合前端展现的形式写入前端 MySQL 数据库**

1.2 数据分区方案

- **背景知识**
  - 用户行为统计事实
    1. 有很多用户是先不登录,等拍下宝贝时才登录(小于 20%)
    2. 同一 session 中用户切换不同帐号登录的概率很低
    3. 统计来源时通常只考虑单一用户的访问路径,当同一 session 中用户在不同帐号间切换时,理论上说应该相应截断访问路径以保证准确性
  - atpanel 日志行为

1. atpanel 日志的 uid 和 mid 之间是多对多关系
2. atpanel 日志在未登录情况下 uid 为空、登录后 uid 为用户数字 ID，uid 为空的比例约 20%
3. 只要用户不关浏览器，atpanel 日志的 sid 字段（session id）就不会改变，即便用户换帐号重新登录也一样

- acookie 日志行为
  1. acookie 日志在未登录情况下会将 uid 填为上次相同 acookie 登录时的 uid，uid 为空的比例小于 1%
- 业务日志行为
  1. 点击日志同 atpanel 日志格式一样
  2. 成交日志中的 buyer_id 可以同 atpanel 日志的 uid 直接关联（前提是用户已登录）
  3. 除点击日志外的大部分业务日志（成交、收藏、拍下等）只含有用户标识，与 atpanel 日志的 uid 才能关联

- **切分方案**
  - 离线计算建树使用 atpanel 日志的 `mid+sid` 作为切分 key，这是考虑到同一 session 中切换帐号的用户比例较小，而先不登录直等到拍下时登录的用户比例较高。为了容忍前一情况，切分 key 中就不能包含 uid
  - 实时计算为了尽量分散计算量，选择基于 atpanel 日志的 uid 和 sid 构造出的 puid 作为切分 key，然后在切分到的 storm bolt 实例中再根据 sid 分组进行建树操作。这样分组后建树结果同离线计算结果不一致，但考虑到实时数据仅用于展现并不保存，且数据仍有前后对比价值，应该可以接受。另外当 atpanel 和 acookie 日志整合完毕后，uid 为空的情况会大大减少，结果差异也会进一步减少。
    * 伪 uid (即 puid) 构造方法为（这里为了避免存储扁平文件时产生分隔符混淆，使用可见字符分隔各字段）：
      1. 若 uid 非空，则 puid 为 "<uid>:"
      2. 若 uid 为空，则 puid 为 ":<sid>"
    * 点击日志同 atpanel 日志一样以 puid 作为切分 key，保证同一用户的点击和流量数据总是发往同一 storm bolt 实例
    * 成交日志以 buyer_id 构造出的 puid 作为切分 key，保证同一用户的成交和流量数据总是发往同一 storm bolt 实例

1.3 离线部署与计算

目前云梯尚不支持自动提交和删除任务，故离线效果计算系统需要每日 **人工提交** 已审批的报表计算任务，原本 PRD 中设计的 90 天报表计算有效期目前也不能自动完成，只能到时候人工去删除任务。

admin 机器上运行的定时任务每隔 5 分钟从前端 MySQL 的 `plan_share` 表中取得已审批的报表配置 ID 并获取相关数据后，将其转换为效果平台后端 XML 格式文件，按照报表审批通过日期以如下结构保存在 admin 机器上：

- 根目录 `/home/lz/effect_platform/conf/report/<YYYY>/<MM>/<DD>/`
  - 这里 `<YYYY>/<MM>/<DD>` 即为报表审批通过日期
- 报表配置 XML 文件名为 `report_<plan id>.xml` ，这里 `<plan id>` 为前端生成的全局唯一报表编号

离线系统的定时任务从 admin 机器当天的目录下 scp 出所有的 XML 文件，以便进行人工任务提交。

这里离线系统需要的配置信息为：

配置项	含义	示例
admin_host	整个效果平台内唯一的管理机主机名	admin_host=10.232.36.4

[TODO] **后续需要同云梯负责人员讨论任务自动提交和删除方案（士诚跟进）**

1.4 实时部署与计算

实时计算逻辑全部打包为 effect_core.jar 包。构建新任务时，将封装了拓扑创建逻辑入口函数的 EffectTopology.class 、计算方案 XML 配置文件、effect_core.jar 包的内容和其他配置文件一同打包为最终的计算任务包，再通过 storm jar 命令提交。创建出的拓扑名统一为 effect_plan_<plan id> ，例如 effect_plan_12345 。

admin 机器上运行的定时任务每隔 5 分钟从前端 MySQL 的 plan_share 表中取得已审批的报表配置 ID 并获取相关数据后，将其转换为效果平台后端 XML 格式文件，按照上述过程打包出报表相关的计算任务 jar 包提交到 storm 集群。同时定时任务还会从同一张表中取得所有要停止计算的报表 ID，使用 storm kill effect_plan_<plan id> 的形式终止对应的计算拓扑。

v1.0 版中每个报表计算任务拓扑都包含 3 个计算任务：详情计算、汇总计算和外投广告指标计算。其中详情计算任务计算按来源路径汇总的指标信息，汇总计算任务计算指定效果页的全来源汇总指标信息，外投广告指标计算任务计算外投广告点击数和点击人数指标。

本来若对 UV 等去重计数指标使用概率算法计算出 bitmap 后，即可在前端对所有来源进行汇总计算，但目前的时间压力和资源都暂时不能满足需要，故需要汇总计算。

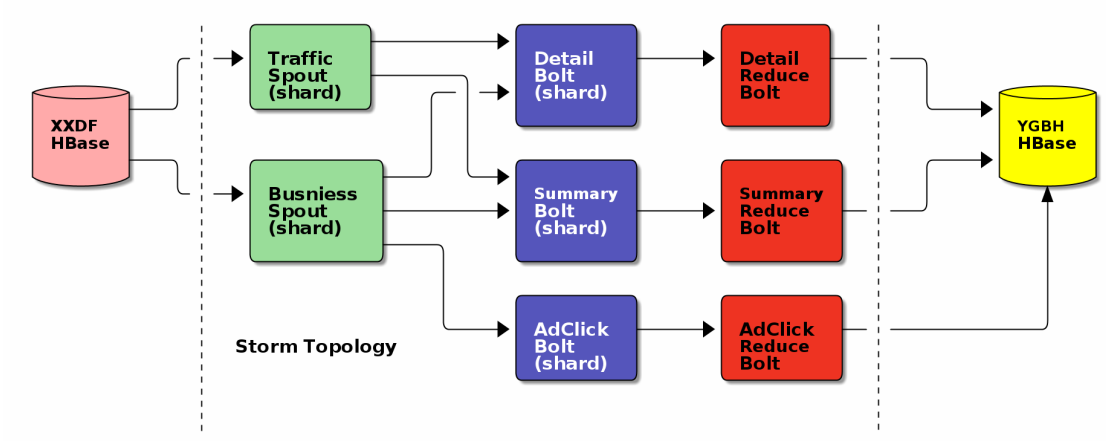
外投广告计算逻辑同路径规则完全无关，故使用独立逻辑单独计算。

汇总计算任务中使用的路径匹配规则为单跳，页面类型为用户指定的一个或多个效果页类型，展开级别为规则级，例如 10000,10001[e=rule] ；其余规则同详情计算一致。 **v1.0 版中暂不考虑用户指定通配效果页的情况（即最后一跳为 *）！**

最终计算任务包内容如下所示：

```
META-INF/
META-INF/MANIFEST.MF
com/etao/lz/EffectToplogy.class
com/etao/lz/effect/**.class
config/plan.xml
config/topo.xml
```

部署结构示意图如下：



其中：

- `Traffic Spout`：用来分区读入 HBase 中吸星大法产出的 atpanel 流量日志
- `Business Spout`：用来分区读入 HBase 中吸星大法产出的成交/拍下/点击日志
- `Detail Bolt`：按 puid 进行 sharding，在独立来源路径级别进行建树、来源路径识别及初始指标计算的任务
- `Summary Bolt`：按 puid 进行 sharding，在效果页汇总级别进行建树及初始指标计算的任务
- `AdClick Bolt`：按 ad_id 进行 sharding，计算外投广告点击量的任务
- `Detail Reduce Bolt`：在独立来源路径级别进行跨 shard 指标汇总的任务，产出最终每独立来源路径指标数据并定时刷新到 HBase 中
- `Summary Reduce Bolt`：在效果页汇总级别进行跨 shard 指标汇总的任务，产出最终效果页汇总指标数据并定时刷新到 HBase 中
- `AdClick Reduce Bolt`：在外投广告 ad_id 级别进行跨 shard 外投广告点击量汇总的任务，产出最终每外投广告 ad_id 数据并定时刷新到 HBase 中

2 URL 类型匹配

2.1 URL 类型匹配逻辑

- 每条访问日志记录有 2 个页面类型属性：
  - rtype - 当前访问页面前一跳的页面类型 ID，根据不同的来源和应用场景，该属性可能由 referer（如站外来源）或 url（如绝大部分广告来源）匹配的结果决定；设置该属性是为了方便构建全息树时判断确切的页面类型并在必要时插入虚拟页面类型节点；
  - ptype - 当前访问页的页面类型 ID，该属性仅由 url（如用户自定义页面）匹配的结果决定
- URL 类型匹配的执行流程为：
  - 读取所有 url_type 匹配规则，按照 (`<match_field>`, `<target_type>`) 组合分成以下 3 组，每组内再按照规则的 `<priority>` 排序：
    1. (`referer`, `referer`) - 即匹配目标是当前访问日志的 referer 字段，匹配成功时设置当前访问日志的 `rtype` 属性为规则对应的 `type_id`；
    2. (`url`, `referer`) - 即匹配目标是当前访问日志的 url 字段，匹配成功时设置当前访问日志的 `rtype` 属性为规则对应的 `type_id`；
    3. (`url`, `url`) - 即匹配目标是当前访问日志的 url 字段，匹配成功时设置当前访问日志的 `ptype` 属性为规则对应的 `type_id`；**v1.0 版中所有用户定义的 URL 模式都属于此范畴**
  - 对于每条访问日志记录，按照以上顺序执行这 3 组匹配规则，每组规则内按优先级顺序找到首个匹配规则即停止，后执行的规则组可覆盖先执行的规则组设置的属性；
  - 在没有相匹配的规则时，rtype 或 ptype 默认值都是 0，即表示未知类型页面，例如直接访问页面的 rtype 就是 0。

2.2 预定义 URL 类型匹配规则

预定义来源 URL 类型匹配规则及对应 ID 见 redmine 上澄苍的文档

2.3 匹配后默认输出的附加属性

URL 类型匹配后除了方案中的自定义属性之外，还要产出如下附加属性：

属性名	含义	示例
ali_corp	当前访问页面属于阿里集团的哪个子公司（依靠 URL 中的主机名判定）	1

- `ali_corp` 字段取值为：
  - 0 - 未知或非阿里系
  - 1 - 淘宝
  - 2 - 天猫
  - 3 - 一淘
  - 4 - 聚划算

TBD

3 全息树构建与来源路径识别

3.1 对外接口定义

离线和实时效果计算在全息树构建和来源路径识别部分逻辑相同，可以抽出公共组件供双方使用。组件接口需要满足以下条件:

- 使用流式日志增量建树
- 可随时获取当前已建立的部分树
- 可方便地获得树中每个节点的 index root path (即从树根到该节点的节点序号路径)，供离线计算使用
- 可方便地获得树中每个节点的 type root path (即从树根到该节点的节点类型路径)，供来源路径识别使用

现定义组件接口形式如下:

```
1  /** URLMatcher 产出的带有页面类型标识的日志对象
2   */
3  public interface PTLogEntry extends Map<String, Object> {
4      /** 返回当前访问日志 url 对应的页面类型 ID,若无法识别则返回 0 */
5      public long getPType();
6      /** 返回当前访问日志 referer 对应的页面类型 ID,若无法识别则返回 0 */
7      public long getRType();
8  }
9
10 /** 来源路径元数据（归属点时戳和优先级）
11  */
12 public class SourceMeta {
```

```
13      public SourceMeta();
14      /** 同一条来源路径在树中可能先后出现多次,
15       * 本方法获取当前来源路径在全息树中首次出现时的归属点时戳
16       */
17      public long getFirstOpTS();
18      /** 同一条来源路径在树中可能先后出现多次,
19       * 本方法获取当前来源路径在全息树中末次出现时的归属点时戳
20       */
21      public long getLastOpTS();
22      /** 获取来源路径对应规则模板优先级
23       */
24      public int getPriority();
25  }
26
27  /** 全息树节点对象
28   */
29  public class HoloTreeNode {
30      public HoloTreeNode(PTLogEntry logEntry);
31      /** 获取节点的序号路径,为点分隔的各级节点序号,
32       * 序号为节点在当前全息树内的顺序编号
33       */
34      public String getSerialRootPath();
35      /** 获取节点的页面类型路径,为直接串接的各级节点页面类型编号(+256),
36
37       * 用于来源路径识别,该路径可能包含虚节点
38       */
39      public String getPTypeRootPath();
40      /** 获取节点属性集合,不包括原始日志中已有的信息
41       */
42      public PTLogEntry getPTLogEntry();
43      /** 获取当前节点已匹配的来源路径列表,结果集中键
44       * 为来源标识串,值为来源路径元数据
45       */
46      public Map<String, SourceMeta> getSources();
47
48      /** 获取全息树中当前节点的父节点
49       */
50      public HoloTreeNode getParent();
51      /** 获取全息树中当前节点的子节点
52       */
53      public List<HoloTreeNode> getChildren();
54      /** 当前节点是否是效果页？
55       */
56      public boolean isEffectPage();
57  }
58
59  /** 全息树接口
60   * 使用独立接口而非直接用 SortedMap 表达全息树是为了方便添加树级别信息
61   */
62  public interface HoloTree extends SortedMap<Long, HoloTreeNode> {
63  }
64
65  /** 全息树构建器对象,同时会进行来源路径识别
```

```java
 */
public class HoloTreeBuilder {
    public HoloTreeBuilder(HoloConfig conf);
    /** autoCompact 参数控制是否在完成单个树的构建后自动进行删减压缩,
     * 实时计算时需要开启该功能,离线无需开启
     */
    public HoloTreeBuilder(HoloConfig conf, boolean
        autoCompact=false);
    /** 传入构建树所需的日志,内部建树时有最大节点数限制,超过时自动截断为多棵树
     */
    public void appendLog(PTLogEntry logEntry);
    /** 获取当前树列表,每棵树为按时间顺序排序的节点集合,树结构通过节点
     * 相关方法获取。注意 SortedMap 中的键不是原始的日志时戳,因为其不
     * 保证唯一,不能做 map key!
     * 仅离线计算使用
     */
    public List<HoloTree> getCurrentTrees();
    /** 清空当前全息树数据
     * 仅离线计算使用
     */
    public void flush();
}

/** 效果平台配置解析类
 */
public class HoloConfig {
    /** 解析指定的配置文件内容
     */
    public HoloConfig(String confFilePath) throws Exception;
}
```

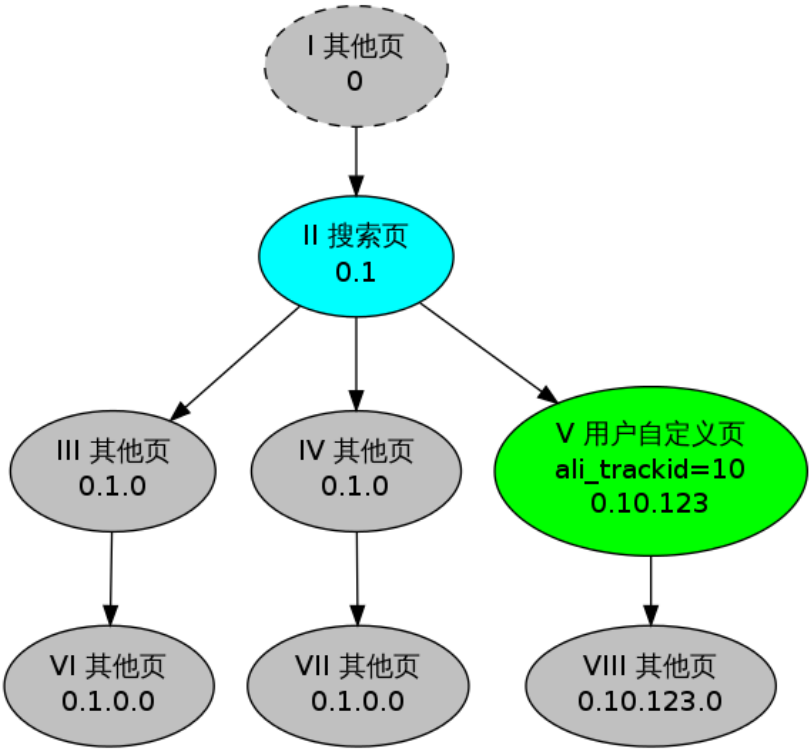
3.2 实时/离线计算方案 XML 配置文件

格式与定义见 redmine 上清无的文档

3.3 全息树构建

发现树根节点时,其页面类型 root path 构建为 \u<rtype+256> \u<ptype+256> ,即引入一个虚拟的 referer 节点。例如若树根节点 url 为用户自定义页面,页面类型 ptype 为 10000,而其 referer 类型识别为直通车广告,页面类型 rtype 为 104,则树根节点的页面类型 root path 将为 \u0168\u2810 。

中间节点存在非 0 的 rtype 属性,且同其父节点的 ptype 属性不同时,中间节点的页面类型 root path 中上一跳页面类型将以自己的 rtype 而非父节点的 ptype 构建。这是因为大部分广告来源的识别需要通过当前页面判定上一跳的类型,当页面同时存在广告和非广告链接且都被用户点击时,会让页面自身的类型识别产生歧义,通过上述处理方式可以消除该歧义,也很容易从直观角度说明含义:



这里第 II 跳搜索页是首条访问日志，而虚线框出的第 I 条节点是根据其 referer 类型在 root path 上插入的虚拟节点。第 V 跳是点击搜索页上直通车广告带来的访问，故在其 root path 中将上一跳搜索页的页面类型改为了直通车对应 ID。其后的子节点都会继承这一新的 root path。

TBD

3.4 来源路径规则向正则表达式的转换

- 每条来源路径规则将转换为一条正则表达式，方法为：
  1. 将路径规则中每个节点要匹配的 type_id 转换为对应数值 +256 的 Unicode 字符（+256 是为了避免转换结果同正则元字符重叠），如 type_id=10 转换为 `\u010a`；
  2. 将路径规则中设为 `*` 的节点转换为正则通配算符 `.`；
  3. 单个节点中要匹配多个逗号分隔 type_id 的，将对应转换后的 Unicode 字符用正则 union 算符 `|` 连接起来；
  4. 路径跳数 `n` 和 `*` 转换为量词修饰的正则通配算符 `.{n-1}` 或 `.*`，若 `n` 为 1 则不插入任何算符；
  5. 每个节点转换出的正则片段外增加捕获算符 `()`；
  6. 路径规则的归属点节点到路径末尾的正则片段外增加捕获算符 `()`；
  7. 最后在结尾增加 `$` 表示从尾到头匹配；
- 例子：
  1. 来源路径规则（第 2 跳为归属点 o）：`1-1->2,3[o]-2->2-1->*`
  2. type_id 转换为 Unicode 字符后：`\u0101-1->\u0102,\u0103-2->\u0102-1->*`
  3. 节点 `*` 转换为正则通配算符后：`\u0101-1->\u0102,\u0103-2->\u0102-1->.`
  4. 逗号分隔 type_id 转换后：`\u0101-1->\u0102|\u0103-2->\u0102-1->.`

5. 路径正则化并增加每个节点的捕获算符后: `(\u0101)(\u0102|\u0103).{1}(\u0102)(.)`
6. 增加归属点路径捕获算符并增加 $ 后: `(\u0101)((\u0102|\u0103).{1}(\u0102)(.))$`

3.5 来源路径识别匹配

“其他来源”识别规则:

- 用户自定义效果页(路径中最后一条)是具体的页面类型时,所有踩过该类型页面但不匹配任何一条用户定义的路径规则的路径上所发生的效果都归到“其他来源”上;
- 用户自定义效果页(路径中最后一跳)为通配符 * 时,不存在“其他来源”,此时在不匹配任何一条用户定义的路径规则的路径上所发生的效果都会被 **直接丢弃!**

上述“其他来源”识别规则实现时可按如下方式处理:

1. 取出计算方案配置中效果页(即路径最后一跳)不为通配符 * 的路径规则,将所有效果页具体页面类型组合为单个匹配正则表达式(参考 来源路径规则向正则表达式转换)。
   - 例如所有具体的页面类型集合为 `{1,2,3}` 时,构建出的表达式为 `\u0101|\u0102|\u0103` ;
2. 对于全息树上新增的节点,使用该正则判定其 url 是否是用户指定的效果页,若是则为该节点增加一个规则 ID 为 -1 的来源,表示“其他来源”
3. 若不匹配该正则则不做任何动作

TBD

4 效果指标计算

4.1 效果指标体系和算法

参考 redmine 上士诚的文档 ,其中同时给出了指标 ID。
注意:

- 所有 UV 统一使用 atpanel 日志中的 `uid_mid` 字段值去重计数来计算。

4.2 业务数据关联算法

- 成交事件同流量日志的关联和归属现在是一起完成的,参考 效果归属逻辑
- 外投广告点击事件同来源路径无关,不需要同流量日志关联,可以另起一个任务直接接收外投广告点击日志计算相关指标。

4.3 效果归属逻辑

成交效果归属逻辑如下:

1. 按照成交宝贝 ID 从成交时间开始向前寻找归属有效周期内所有已识别出来源路径(即已染色)的同 ID 宝贝页
2. 若有此类页面,则将其中所有来源路径取出,按照如下原则进行归属:

(a) 来源路径对应规则优先级不同时，归属至规则优先级高的来源路径；

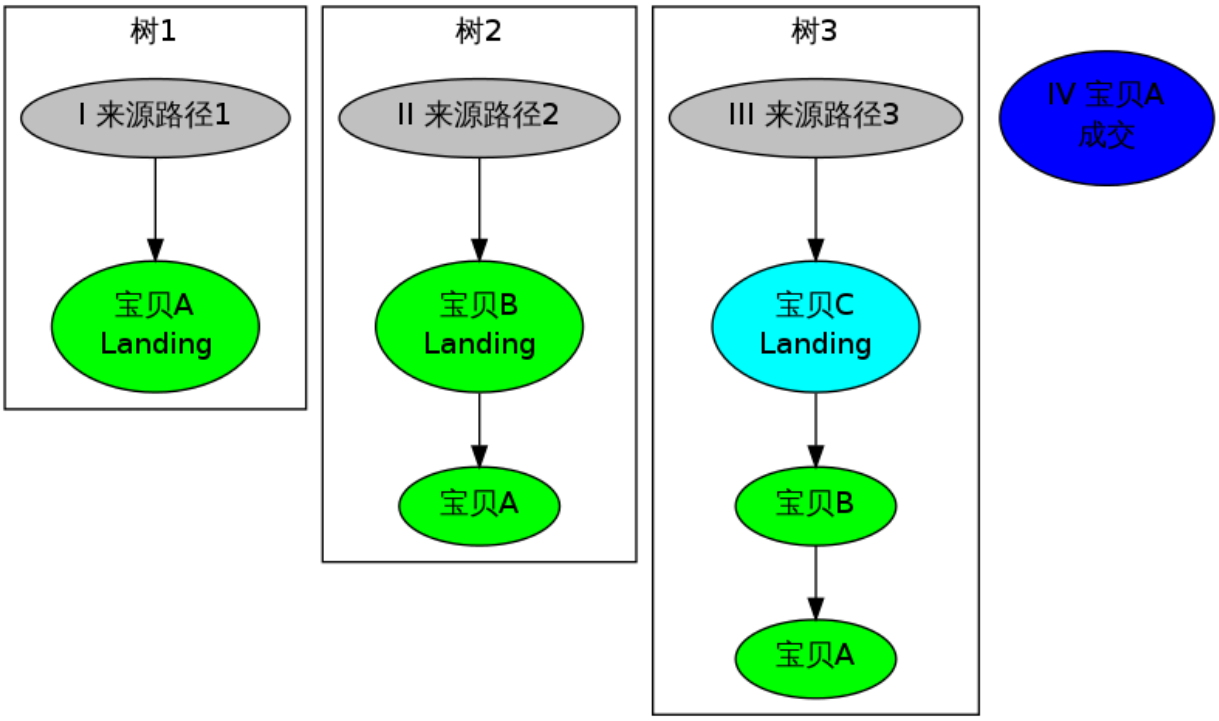
(b) 来源路径对应规则优先级相同时，按照用户选择的同优先级归属方法进行归属；

(c) 同优先级归属方法为 first/last 时，需要根据各来源路径的归属点出现时刻先后顺序决定归属给谁

3. 若无此类页面，则 **丢弃该成交** ！

   - 这里的假定是某宝贝成交必然意味着在归属有效周期内访问过该宝贝页

例如如下访问顺序：



宝贝 A、B 同店，宝贝 C 与它们异店。树 1、2、3 在归属有效期内先后被构建出来，最后宝贝 A 成交。按照前述来源归属逻辑，宝贝 A 应该在来源路径 1、2、3 之间进行归属。归属结果可能为：

1. 若来源路径 1 对应规则优先级高，则成交归属为来源路径 1 的直接成交，因其 Landing 宝贝就是 A

2. 若来源路径 2 对应规则优先级高，则成交归属为来源路径 2 的间接成交，因其 Landing 宝贝 B 不是 A 但与 A 同店

3. 若来源路径 3 对应规则优先级高，则成交归属为来源路径 3 的其他成交，因其 Landing 宝贝 C 不是 A 且与 A 异店

4. 若来源路径 1、2、3 优先级相同，则成交按同优先级归属方法归属：

   (a) 归属方法为 first，则成交归属为来源路径 1 的直接成交

   (b) 归属方法为 last，则成交归属为来源路径 3 的其他成交

   (c) 归属方法为 all，则成交分别归属到来源路径 1 的直接成交、2 的间接成交和 3 的其他成交

4.3.1 Landing 宝贝页的定位

为了判定直接/间接/其他成交，需要确定 Landing 宝贝页的位置。这有 2 种情况：

1. Landing 宝贝页就是效果页，如用户关注广告直接引导到单品页面的路径，此时效果页是单品页
2. Landing 宝贝页是效果页的下一跳，如用户关注广告引导到活动页，而活动页上再放置单品页面链接，此时效果页是活动页

因此在查找 Landing 宝贝页时需要先查看效果页，当效果页非宝贝页时再看其下一跳页面。

4.4 特殊情况处理

- “其他来源”已经标识在全息树上，只有在没识别出其他来源时才会选择该来源进行归属，若成交关联的节点没有标识任何来源则直接丢弃该成交（具体见 其他来源识别规则 ）
- 成交和流量指标需要根据具体发生效果页面的域名，分别统计其在四淘（淘宝、一淘、天猫、聚划算）内的分布情况，由于聚划算和一淘没有直接成交，故成交指标只可能分布在淘宝和天猫内，流量指标则四淘都有。
- v1.0 版本中不允许用户指定路径上归属点位置，由系统默认设置路径中最后一跳为归属点，这样会导致所有路径规则的归属点位置相同，无法起到其原有的路径区分效果。故这里约定在一条路径同时匹配了最后一跳相同的多条同优先级路径规则时，**优先选择规则 ID 较小的来源路径进行归属**；同时匹配的多条路径规则最后一跳不同时则可以正常用同优先级归属规则进行处理。

5 数据交互

5.1 定时任务：转换前端用户配置到后端 XML 配置文件

5.1.1 转换过程

v1.0 版本中 XML 配置文件中有部分固定值：

XML XPath	属性值
effect_plan/ver	1
effect_plan/update_interval	60
effect_plan/tree_split	[method]=”none”
effect_plan/src_path/rule/priority	10
effect_plan/src_path/rule/limit	/num=1000, /effect_id=0
effect_plan/src_path/rule/effect_owner	总是规则中路径最后一个节点的 ID
effect_plan/url_type/rule/priority	10
effect_plan/url_type/rule	澄苍的文档 中每个预设来源都要生成 URL 类型匹配规则

v1.0 版本中不实现用户自定义来源，故前端 MySQL 中的 `src_custom_config` 表这里不会使用。

前端 MySQL `plan_config` 表中，各字段同 XML 配置文件对应关系为：

字段名	对应 XML XPath	转换规则
plan_id	effect_plan/plan_id	直接复制
user_id	effect_plan/analyzer_id	直接复制
effect_url	在路径配置部分引用其类型 ID	见 (1)
src_type	在路径配置部分引用其内容	见 (2)
path_type	同 path_config 表关联使用	见 (3)
belong_id	effect_plan/attr_calc	[method]= 数值对应的枚举串,见 (4)
ind_ids	effect_plan/effects/ind	[id]=seqno, [ind_id]=ind_ids 中的每项,见 (5)
period	effect_plan/period	取值为字段值对应的周期时间,1-1 天
expire_type	effect_plan/ttl	取值为字段值对应的过期时间,1-90 天

1. 应为 effect_url 分配一个全新的页面类型 ID:epid,以便后续引用,epid 在方案内唯一且大于等于 10000,另外还要在 URL 匹配规则中为 effect_url 另加一条 type_id 为 epid 的规则,其匹配目标总是 url 字段
2. src_type 含有外投广告/SPM 类型时,需要根据 配置转换特例处理逻辑 进行处理
3. 用户不限定路径时,path_type 为 0,此时需生成一条默认路径模板规则 `<src_type>-1-><effect page type>` ,且规则 ID 也为 0;用户限定路径时,path_type 为 1,此时需要用 plan_id 在 `path_config` 表中查出所有路径记录,并生成对应的路径模板规则
4. belong_id 枚举值:见 `plan_config` 表说明(1-first,2-last,3-equal,4-all)
5. ind_ids 中含有中间页指标时,需要根据 配置转换特例处理逻辑 进行处理

前端 MySQL `path_config` 表中,各字段同 XML 配置文件对应关系为:

字段名	对应 XML XPath	转换规则
path_id	effect_plan/src_path/rule/path_id	直接复制
data	effect_plan/src_path/rule/path/node	见 (1)

1. 为 data 中 JSON 数组的每一项顺序生成对应的 node 配置,为此要给 `data[*].url` 分别创建 URL 类型匹配规则并分配全新的页面类型 ID: `dpid[*]` ,以便路径中作为 type_ref 引用,dpid[*] 和 epid 一样在方案内唯一且大于等于 10000;URL 类型匹配规则中的 `match_field` 固定为 `url` ; `data[*].step` 作为 `node["next"]` 属性值, `node["expand"]` 属性值固定为 `rule`

5.1.2 转换特例

- 计算中间页指标时,用户定义的来源路径模块规则后要增加 `-1->*[e=rule]` ,因为实际效果页是用户指定效果页的下一跳。
- 用户选择来源分类中包含“外投广告”时,需要将所有用户自定义路径规则单独复制一份,指定其首跳页面类型为外投广告且按 `adid` 属性展开,并将原路径规则 ID 加 10000 作为外投广告路径规则专有 ID。这样将最终产出数据转换至前端 MySQL 库时不需要再查找对应的报表配置数据,只要看到 src 标识中路径规则 ID 在 10000 以上,即可知道是外投广告,直接将规则 ID 减去 10000 即可得到实际应回填的原规则 ID,首跳展开数据就是应回填的 adid 值。
- 用户选择来源分类中包含“SPM”时,和“外投广告”一样要单独复制一份用户自定义路径规则,指定其首跳页面类型为 SPM 且按 `spm` 属性展开,并将原路径规则 ID 加 20000 作为 SPM 路径规则专有 ID。最终产出数据中只要看到 src 标识里的路径规则 ID 在 20000 以上,即可知道是 SPM,将其减去 20000 即可得到实际应回填的原规则 ID,首跳展开数据就是应回填的 spm 值。

5.1.3 转换示例 I

本示例中用户在来源中只选择了联盟广告、直通车和淘宝客,没有限定访问路径,归属规则选择了首跳归属,计算指标选择了单品 PV、单品 UV,其他均为默认值

- 前端 MySQL 库内容
  plan_config 表内容为:

字段	值
plan_id	123
user_id	10
effect_url	http://detail.tmall.com/item.htm?id=15656723521
src_type	101,104,105
path_ids	0
belong_id	1
ind_ids	200,201
period	1
expire_type	1

  path_config 表为空。

- 转换过程
  生成 XML 不变配置部分:

```
<ver>1</ver>
<update_interval>60</update_interval>
<tree_split method="none" />
```

  根据 plan_config 表中内容生成基本配置:

```
<analyzer_id>10</analyzer_id>
<plan_id>123</plan_id>
<ttl>90</ttl>
<period>1</period>
<attr_calc method="first" />
```

  根据 plan_config 表中 ind_ids 字段内容生成指标配置:

```
<effects>
   <ind id="0" ind_id="200" />
   <ind id="1" ind_id="201" />
</effects>
```

  取出 plan_config 表中的 effect_url,为其分配唯一且大于等于 10000 的页面类型 ID:10000,然后为 src_type 中出现的预设类型和 effect_url 分别生成 URL 类型匹配配置:

```
<url_type>
   <rule>
      <priority>10</priority>
      <type_id>101</type_id>
      <match_field>url</match_field>
      <match_regexps>
         <match_regexp>
```

```
            <regexp><![CDATA[pid=mm_]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>104</type_id>
      <match_field>url</match_field>
      <target_type>referer</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[ali_trackid=1_]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>105</type_id>
      <match_field>url</match_field>
      <target_type>referer</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[ali_trackid=2(:|%3A)]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>10000</type_id>
      <match_field>url</match_field>
      <target_type>url</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[http://detail.tmall.com/item.htm?id=15656723
         </match_regexp>
      </match_regexps>
   </rule>
</url_type>
```

由于用户未设置路径限定规则,故只有一条默认路径 `<src_types> -1-> <effect_url>` ,且其 ID 为 0,生成的路径配置如下:

```
<src_path>
   <rule>
      <priority>10</priority>
      <limit>
         <num>1000</num>
         <effect_id>0</effect_id>
      </limit>
      <path_id>0</path_id>
      <path>
         <node id="0" type_refs="101,104,105" next="1"
            expand="ptype" />
```

```
            <node id="1" type_refs="10000" expand="rule" />
        </path>
        <effect_owner>1</effect_owner>
    </rule>
</src_path>
```

将各部分合并放置在 <effect_plan></effect_plan> 标签对中即构成完整的 XML 配置文件:

```
<effect_plan>
<ver>1</ver>
<update_interval>60</update_interval>
<tree_split method="none" />
<analyzer_id>10</analyzer_id>
<plan_id>123</plan_id>
<ttl>90</ttl>
<period>1</period>
<attr_calc method="first" />
<effects>
    <ind id="0" ind_id="200" />
    <ind id="1" ind_id="201" />
</effects>
<url_type>
    <rule>
        <priority>10</priority>
        <type_id>101</type_id>
        <match_field>url</match_field>
        <match_regexps>
            <match_regexp>
                <regexp><![CDATA[pid=mm_]]></regexp>
            </match_regexp>
        </match_regexps>
    </rule>
    <rule>
        <priority>10</priority>
        <type_id>104</type_id>
        <match_field>url</match_field>
        <match_regexps>
            <match_regexp>
                <regexp><![CDATA[ali_trackid=1_]]></regexp>
            </match_regexp>
        </match_regexps>
    </rule>
    <rule>
        <priority>10</priority>
        <type_id>105</type_id>
        <match_field>url</match_field>
        <match_regexps>
            <match_regexp>
                <regexp><![CDATA[ali_trackid=2(:|%3A)]]></regexp>
            </match_regexp>
        </match_regexps>
    </rule>
```

```
45      <rule>
46          <priority>10</priority>
47          <type_id>10000</type_id>
48          <match_field>url</match_field>
49          <match_regexps>
50              <match_regexp>
51                  <regexp><![CDATA[http://detail.tmall.com/item.htm?id=15656723
52              </match_regexp>
53          </match_regexps>
54      </rule>
55  </url_type>
56  <src_path>
57      <rule>
58          <priority>10</priority>
59          <limit>
60              <num>1000</num>
61              <effect_id>0</effect_id>
62          </limit>
63          <path_id>0</path_id>
64          <path>
65              <node id="0" type_refs="101,104,105" next="1"
                    expand="ptype" />
66              <node id="1" type_refs="10000" expand="rule" />
67          </path>
68          <effect_owner>1</effect_owner>
69      </rule>
70  </src_path>
71  </effect_plan>
```

5.1.4 转换示例 II

本示例中用户在来源中选择了外投媒体、SPM 和旺旺广告,限定有两条访问路径,归属规则选择了首跳归属,计算指标选择了中间页 PV、中间页 UV,其他均为默认值

- 前端 MySQL 库内容
  `plan_config` 表内容为:

字段	值
plan_id	124
user_id	10
effect_url	http://brand.tmall.com/?spm=1.1000386.221827.22&prt=1332757125471&prc=1
src_type	100,103,114
path_ids	1,2
belong_id	1
ind_ids	400,401
period	1
expire_type	1

`path_config` 表内容为:

字段	值
path_id	1
plan_id	124
data	[{name:"…", url:"http://www.taobao.com", step:1}]
path_id	2
plan_id	124
data	[{name:"…", url:"http://www.tmall.com", step:1}]

- 转换过程
  生成 XML 不变配置部分：

```
<ver>1</ver>
<update_interval>60</update_interval>
<tree_split method="none" />
```

根据 plan_config 表中内容生成基本配置：

```
<analyzer_id>10</analyzer_id>
<plan_id>124</plan_id>
<ttl>90</ttl>
<period>1</period>
<attr_calc method="first" />
```

根据 plan_config 表中 ind_ids 字段内容生成指标配置：

```
<effects>
   <ind id="0" ind_id="400" />
   <ind id="1" ind_id="401" />
</effects>
```

取出 plan_config 表中的 effect_url，为其分配唯一且大于等于 10000 的页面类型 ID：10000；取出 path_config 表中的 data 里所有的 url，为其分配唯一且大于等于 10000 的页面类型 ID：10001、10002，然后为 src_type 中出现的预设类型、effect_url 和 path_config 表中的 url 分别生成 URL 类型匹配配置：

```
<url_type>
   <rule>
      <priority>10</priority>
      <type_id>100</type_id>
      <match_field>url</match_field>
      <target_type>referer</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[tb_market_id=]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>103</type_id>
      <match_field>url</match_field>
      <target_type>referer</target_type>
      <match_regexps>
         <match_regexp>
```

```
            <regexp><![CDATA[spm=([^?&]+)]]></regexp>
            <props>
               <prop field="spm" value="$1" />
            </props>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>114</type_id>
      <match_field>url</match_field>
      <target_type>referer</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[ali_trackid=10_]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>10000</type_id>
      <match_field>url</match_field>
      <target_type>url</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[http://brand.tmall.com/?spm=1.1000386.221827
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>10001</type_id>
      <match_field>url</match_field>
      <target_type>url</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[http://www.taobao.com]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>10002</type_id>
      <match_field>url</match_field>
      <target_type>url</target_type>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[http://www.tmall.com]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
</url_type>
```

由于用户设置了 2 条限定路径,且选中来源中包含了外投广告和 SPM 这 2 个特殊来源,故总共会有 3 组共 6 条路径规则,分别为一般来源/外投广告/SPM 生成。其中一般来源对应的 2 条路径规则 ID 就是 path_config 表中的 path_id 字段值,外投广告对应的 2 条路径规则 ID 是该值加 10000、expand 属性为:adid,SPM 对应的 2 条路径规则 ID 是该值加 20000、expand 属性为:spm。最终产生的路径规则配置如下:

```
<src_path>
   <rule>
      <priority>10</priority>
      <limit>
         <num>1000</num>
         <effect_id>0</effect_id>
      </limit>
      <path_id>1</path_id>
      <path>
         <node id="0" type_refs="114" next="1" expand="ptype" />
         <node id="1" type_refs="10001" next="1" expand="rule" />
         <node id="2" type_refs="10000" expand="rule" />
      </path>
      <effect_owner>1</effect_owner>
   </rule>
   <rule>
      <priority>10</priority>
      <limit>
         <num>1000</num>
         <effect_id>0</effect_id>
      </limit>
      <path_id>2</path_id>
      <path>
         <node id="0" type_refs="114" next="1" expand="ptype" />
         <node id="1" type_refs="10002" next="1" expand="rule" />
         <node id="2" type_refs="10000" expand="rule" />
      </path>
      <effect_owner>1</effect_owner>
   </rule>
   <rule>
      <priority>10</priority>
      <limit>
         <num>1000</num>
         <effect_id>0</effect_id>
      </limit>
      <path_id>10001</path_id>
      <path>
         <node id="0" type_refs="100" next="1" expand=":adid" />
         <node id="1" type_refs="10001" next="1" expand="rule" />
         <node id="2" type_refs="10000" expand="rule" />
      </path>
```

```
42        <effect_owner>1</effect_owner>
43     </rule>
44     <rule>
45        <priority>10</priority>
46        <limit>
47           <num>1000</num>
48           <effect_id>0</effect_id>
49        </limit>
50        <path_id>10002</path_id>
51        <path>
52           <node id="0" type_refs="100" next="1" expand=":adid"
                />
53           <node id="1" type_refs="10002" next="1"
                expand="rule" />
54           <node id="2" type_refs="10000" expand="rule" />
55        </path>
56        <effect_owner>1</effect_owner>
57     </rule>
58
59     <rule>
60        <priority>10</priority>
61        <limit>
62           <num>1000</num>
63           <effect_id>0</effect_id>
64        </limit>
65        <path_id>20001</path_id>
66        <path>
67           <node id="0" type_refs="103" next="1" expand=":spm"
                />
68           <node id="1" type_refs="10001" next="1"
                expand="rule" />
69           <node id="2" type_refs="10000" expand="rule" />
70        </path>
71        <effect_owner>1</effect_owner>
72     </rule>
73     <rule>
74        <priority>10</priority>
75        <limit>
76           <num>1000</num>
77           <effect_id>0</effect_id>
78        </limit>
79        <path_id>20002</path_id>
80        <path>
81           <node id="0" type_refs="103" next="1" expand=":spm"
                />
82           <node id="1" type_refs="10002" next="1"
                expand="rule" />
83           <node id="2" type_refs="10000" expand="rule" />
84        </path>
85        <effect_owner>1</effect_owner>
86     </rule>
87  </src_path>
```

将各部分合并放置在 <effect_plan></effect_plan> 标签对中即构成完整的 XML 配置文件:

```
<effect_plan>
<ver>1</ver>
<update_interval>60</update_interval>
<tree_split method="none" />
<analyzer_id>10</analyzer_id>
<plan_id>124</plan_id>
<ttl>90</ttl>
<period>1</period>
<attr_calc method="first" />
<effects>
   <ind id="0" ind_id="400" />
   <ind id="1" ind_id="401" />
</effects>
<url_type>
   <rule>
      <priority>10</priority>
      <type_id>100</type_id>
      <match_field>url</match_field>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[tb_market_id=]]></regexp>
          </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>103</type_id>
      <match_field>url</match_field>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[spm=([^?&]+)]]></regexp>
            <props>
               <prop field="spm" value="$1" />
            </props>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>114</type_id>
      <match_field>url</match_field>
      <match_regexps>
         <match_regexp>
            <regexp><![CDATA[ali_trackid=10_]]></regexp>
         </match_regexp>
      </match_regexps>
   </rule>
   <rule>
      <priority>10</priority>
      <type_id>10000</type_id>
```

```
51          <match_field>url</match_field>
52          <match_regexps>
53              <match_regexp>
54                  <regexp><![CDATA[http://brand.tmall.com/?spm=1.1000386.221827
55              </match_regexp>
56          </match_regexps>
57      </rule>
58      <rule>
59          <priority>10</priority>
60          <type_id>10001</type_id>
61          <match_field>url</match_field>
62          <match_regexps>
63              <match_regexp>
64                  <regexp><![CDATA[http://www.taobao.com]]></regexp>
65              </match_regexp>
66          </match_regexps>
67      </rule>
68      <rule>
69          <priority>10</priority>
70          <type_id>10002</type_id>
71          <match_field>url</match_field>
72          <match_regexps>
73              <match_regexp>
74                  <regexp><![CDATA[http://www.tmall.com]]></regexp>
75              </match_regexp>
76          </match_regexps>
77      </rule>
78  </url_type>
79  <src_path>
80      <rule>
81          <priority>10</priority>
82          <limit>
83              <num>1000</num>
84              <effect_id>0</effect_id>
85          </limit>
86          <path_id>1</path_id>
87          <path>
88              <node id="0" type_refs="114" next="1" expand="ptype"
                    />
89              <node id="1" type_refs="10001" next="1"
                    expand="rule" />
90              <node id="2" type_refs="10000" expand="rule" />
91          </path>
92          <effect_owner>1</effect_owner>
93      </rule>
94      <rule>
95          <priority>10</priority>
96          <limit>
97              <num>1000</num>
98              <effect_id>0</effect_id>
99          </limit>
100         <path_id>2</path_id>
101         <path>
```

```
102             <node id="0" type_refs="114" next="1" expand="ptype"
                    />
103             <node id="1" type_refs="10002" next="1"
                    expand="rule" />
104             <node id="2" type_refs="10000" expand="rule" />
105         </path>
106         <effect_owner>1</effect_owner>
107     </rule>
108
109     <rule>
110         <priority>10</priority>
111         <limit>
112             <num>1000</num>
113             <effect_id>0</effect_id>
114         </limit>
115         <path_id>10001</path_id>
116         <path>
117             <node id="0" type_refs="100" next="1" expand=":adid"
                    />
118             <node id="1" type_refs="10001" next="1"
                    expand="rule" />
119             <node id="2" type_refs="10000" expand="rule" />
120         </path>
121         <effect_owner>1</effect_owner>
122     </rule>
123     <rule>
124         <priority>10</priority>
125         <limit>
126             <num>1000</num>
127             <effect_id>0</effect_id>
128         </limit>
129         <path_id>10002</path_id>
130         <path>
131             <node id="0" type_refs="100" next="1" expand=":adid"
                    />
132             <node id="1" type_refs="10002" next="1"
                    expand="rule" />
133             <node id="2" type_refs="10000" expand="rule" />
134         </path>
135         <effect_owner>1</effect_owner>
136     </rule>
137
138     <rule>
139         <priority>10</priority>
140         <limit>
141             <num>1000</num>
142             <effect_id>0</effect_id>
143         </limit>
144         <path_id>20001</path_id>
145         <path>
146             <node id="0" type_refs="103" next="1" expand=":spm"
                    />
147             <node id="1" type_refs="10001" next="1"
```

```
                expand="rule" />
148             <node id="2" type_refs="10000" expand="rule" />
149          </path>
150          <effect_owner>1</effect_owner>
151       </rule>
152       <rule>
153          <priority>10</priority>
154          <limit>
155             <num>1000</num>
156             <effect_id>0</effect_id>
157          </limit>
158          <path_id>20002</path_id>
159          <path>
160             <node id="0" type_refs="103" next="1" expand=":spm"
                   />
161             <node id="1" type_refs="10002" next="1"
                   expand="rule" />
162             <node id="2" type_refs="10000" expand="rule" />
163          </path>
164          <effect_owner>1</effect_owner>
165       </rule>
166    </src_path>
167    </effect_plan>
```

5.2 后端输入数据

5.2.1 离线计算系统输入数据

离线系统直接扫描原始日志，自行进行 ETL 清洗工作。

5.2.2 实时计算系统输入数据

实时系统依赖于吸星大法输出结果，该结果已经经过了 ETL 清洗和标准化工作。吸星大法输出结果保存在 HBase 表中，atpanel 流量日志表名为 `browse_log`，外投广告点击日志表名为 `ad_ex_click`，成交/拍下日志需要同时使用表 `pay_order` 和 `biz_order`（其中 `pay_order` 中为成交的实际总金额，`biz_order` 中同时包括了拍下和成交信息）。实时系统需要建立 storm spout 任务从 HBase 中定期扫描当前时刻开始的所有日志记录，以流数据形式转发给后级计算任务。

数据格式参考 redmine 上 太奇的文档

为了避免实时系统从 HBase 中读取日志数据的操作成为瓶颈，吸星大法产出日志的 rowkey 中要包含一个用于 sharding 的字段；为了避免不必要的日志乱序，该 sharding key 粒度应该保证单个用户的数据集合是不被切开的，同时 sharding 数量也不应太多，因为实时系统读取 HBase 用的 storm spout 任务数量也就是 10 的量级；由于日志时间粒度为秒级，为了避免同一秒内日志数据无法区分造成重叠，还需要给 rowkey 附加用户标识和随机数来降低重叠概率。最终吸星大法产出日志的 rowkey 定为如下格式：

```
<CRC32(puid)%32: 1 byte> <ts: 4 bytes> <puid: varlen> <rand: 1 byte>
```

这里：

- `CRC32(puid)%32`：sharding key，最多切分 32 个分区已经可以满足需要；
- `ts`：32 位整数表示的日志秒级时间戳；

- puid : 变长字符串,即文档开始时定义的伪 uid,用来区分不同用户的日志数据;
- rand : 用来区分同一秒内用户日志的随机数,0~255;

5.3 后端产出报表数据存储与交互

实时和离线效果分析系统的统计结果都输出到 HBase 中。

HBase 中的统计结果由定时任务每隔 5 分钟转存到前端 MySQL 的详情表和汇总表中,前端 MySQL schema 设计为包含 PRD 中所有已知指标的宽表,定时任务在转存时根据统计结果对应计算方案配置决定填充宽表中哪些具体字段。前端 MySQL 报表详情表和汇总表 schema 附后。

5.3.1 HBase schema 设计过程

- 报表数据的查询行为由 dump 数据的定时任务决定:首先确定查询日期范围(以天为单位),然后将该日期下所有记录(包括汇总数据和维度数据)全部取出写入前端 MySQL 库
- 实时系统对报表数据的插入修改行为是:每隔一段固定时间(根据业务需要可能是秒级或分钟级),首先确定方案提交者和效果计算方案,然后修改或插入当天指定来源路径下的指标数据
- 离线效果计算系统对报表数据的插入修改行为是:每天固定时刻(一般为 0 点),取出前一天数据计算后的结果,按照方案提交者和效果计算方案,将相关指标数据全部插入 HBase
- 根据该查询行为,可以将 date_ts + user_id + plan_id + dim_id + MD5(src) 定为 HBase 详情表的 rowkey(这里 dim_id 指指标拆分维度,目前有 5 个取值:0(不拆分)/1(淘宝内)/2(天猫内)/3(一淘内)/4(聚划算内); date_ts 为精度到日的整数 epoch 时戳,时分秒都取 0; user_id 就是配置 XML 文件中的 analyzer_id 值),然后将完整来源路径标识和指标数据作为多个 column 保存;HBase 汇总表的 rowkey 与其类似,只是没有 src 相关信息,且其指标数据是跨来源汇总后的结果。

5.3.2 HBase schema 设计小结

1. HBase 详情报表表名定为 effect_rpt ,汇总报表表名定为 effect_rpt_sum ,外投广告点击指标表表名定为 effect_rpt_adclk
2. effect_rpt 表的 rowkey 为: <date_ts:4 字节 > <user_id:4 字节 > <plan_id:4 字节 > <dim_id:2 字节 > <MD5(src):16 字节 >
3. effect_rpt_sum 表的 rowkey 为: <date_ts:4 字节 > <user_id:4 字节 > <plan_id:4 字节 > <dim_id:2 字节 >
4. effect_rpt_adclk 表的 rowkey 为: <date_ts:4 字节 > <user_id:4 字节 > <plan_id:4 字节 > <ad_id: varlen>
5. effect_rpt 表的数据列为:
   (a) d:src 列:存储来源路径标识,仅详情表 effect_rpt 中存在,其格式为 <path id> ^B <node val1> ^C <node val2> ^C ... ^C <node valn> ,这里 <path id> 为效果计算方案配置 XML 文件中对应路径模板规则的 id, <node val?> 为对应路径模板规则中从前往后每个路径节点的具体展开值,其取值内容含义由模

板规则中对应节点的展开选项决定(rule 级别展开时取值为模板规则 ID, ptype 级别展开时取值为匹配到的页面类型 ID,属性级别展开时取值为匹配时对应页面的指定属性值)。

- 对于“其他来源”,本列为 -1^B ,rowkey 中也用对应的 MD5 标识

(b) d:i<indicator id> 列:存储 ID 为 <indicator id> 的指标在当前维度内的对应结果

6. effect_rpt_sum 表的数据列为:

(a) d:i<indicator id> 列:含义同上

7. effect_rpt_adclk 表的数据列为:

(a) d:i<indicator id> 列:目前仅有 ID 为 101 和 108 的指标(即外投广告位点击量和点击人数)

5.3.3 HBase 中产出报表数据存储格式示例

设 user_id 为 13959,plan_id 为 123,当前日期为 2012-5-28,指标效果页 pv 和 uv 的 ID 分别为 103 和 104,模板规则 1 为 t1[e=rule]-1->search[e=:keyword]-1->(t1,t2,t3)[e=ptype]

HBase 详情表 effect_rpt 内容可能为:

rowkey	d:src	d:i103	d:i104
0x4fc24f80 0x00003687 0x0000007b 0x0000 ···	1^B1^C 男装 ^C2	1239	282
0x4fc24f80 0x00003687 0x0000007b 0x0001 ···	1^B1^C 男装 ^C2	388	182
0x4fc24f80 0x00003687 0x0000007b 0x0002 ···	1^B1^C 男装 ^C2	851	100
0x4fc24f80 0x00003687 0x0000007b 0x0000 ···	1^B1^C 女装 ^C3	82911	1828
0x4fc24f80 0x00003687 0x0000007b 0x0001 ···	1^B1^C 女装 ^C3	82000	1800
0x4fc24f80 0x00003687 0x0000007b 0x0002 ···	1^B1^C 女装 ^C3	911	28
···	···	···	···

HBase 汇总表 effect_rpt_sum 内容可能为:

rowkey	d:i103	d:i104
0x4fc24f80 0x00003687 0x0000007b 0x0000	84150	2110
···	···	···

HBase 外投广告指标表 effect_rpt_adclk 内容可能为:

rowkey	d:i101	d:108
0x4fc24f80 0x00003687 0x0000007b 583838282810294_20120606	19292	182
0x4fc24f80 0x00003687 0x0000007b 580038282810294_20120606	9349	18

5.4 定时任务:转换后端报表数据到前端 MySQL

HBase 报表详情表 effect_rpt 的内容转换至前端 MySQL 的 rpt_detail 表中:

1. rowkey 中的 <plan_id> 作为 int 写入 rpt_detail.plan_id
2. rowkey 中的 <date_ts> 转换为本地时间后,将年月日取出拼成 8 位十进制 int 写入 rpt_detail.day
3. rowkey 中的 <dim_id> 作为 int 写入 rpt_detail.dim

4. 将 d:src 按 ^B 切分为 2 部分：
   (a) 前半部分为匹配的路径规则 ID(参考 配置转换特例处理逻辑)：
      i. 若值小于 10000 则为普通规则，将其作为 int 写入 rpt_detail.path_id
      ii. 若值大于等于 10000 小于 20000 则为外投广告规则，将其减去 10000 后写入 rpt_detail.path_id ，同时将外投广告来源 ID (100) 写入 rpt_detail.src_id
      iii. 若值大于等于 20000 小于 30000 则为 SPM 规则，将其减去 20000 后写入 rpt_detail.path_id ，同时将 SPM 来源 ID(103)写入 rpt_detail.src_id
   (b) 后半部分为匹配展开的路径实例，将其按 ^C 切分后：
      i. 若是普通规则，取首字段内容作为 int 写入 rpt_detail.src_id
      ii. 若是外投广告或 SPM 规则，取首字段内容作为 string 写入 rpt_detail.src_ext
5. d:i<ind id> 各列对应写入 rpt_detail.ind_<ind id>

HBase 报表汇总表 effect_rpt_sum 的内容转换至前端 MySQL 的 rpt_summary 表中：

1. rowkey 中的 <plan_id> 作为 int 写入 rpt_summary.plan_id
2. rowkey 中的 <date_ts> 转换为本地时间后，将年月日取出拼成 8 位十进制 int 写入 rpt_summary.day
3. rowkey 中的 <dim_id> 作为 int 写入 rpt_summary.dim
4. d:i<ind id> 各列对应写入 rpt_summary.ind_<ind id>

HBase 外投广告指标表 effect_rpt_adclk 的内容转换至前端 MySQL 的 rpt_extend_ad 表中：

1. rowkey 中的 <plan_id> 作为 int 写入 rpt_extend_ad.plan_id
2. rowkey 中的 <date_ts> 转换为本地时间后，将年月日取出拼成 8 位十进制 int 写入 rpt_extend_ad.day
3. rowkey 中的 <ad_id> 作为 string 写入 rpt_extend_ad.ad_id
4. d:i101 和 d:i108 列分别作为 double 写入 rpt_extend_ad.ind_101 和 rpt_extend_ad.ind_108

5.5 web 相关数据库设计

参考 redmine 上少侠的文档

5.5.1 报表计算方案配置相关

- 计划配置表 plan_config
  保存前端配置界面用户填写的主要配置信息。

字段	类型	长度	说明	示例
plan_id	int	11	计划 id (auto_increment)	1
user_id	int	11	用户 id	1
report_name	varchar	255	报表名称	test_report_name
effect_name	varchar	255	效果页名称	test_effect_name
effect_url	text	-	效果页 URL	http://lz.taobao.com
src_type	text	-	效果页流量来源 (1)	1,2,3
path_type	int	3	到达效果页的访问路径类型 (2)	1
belong_id	int	3	效果归属顺序 (3)	2
ind_ids	text	-	效果指标 ids	3,203
period	int	3	效果归属周期类型:1-1 天	1
expire_type	int	3	过期类型:1-90 天	1
expire_date	int	11	过期日期	20120606
status	int	3	计划状态 (4)	1
ctime	int	11	创建时间	1234567890
mtime	int	11	更新时间	1234567891

1. 效果页流量来源:0 为不限定,目前有 1-17 种,具体配置在来源配置表中
2. 访问路径类型:0:不限定,1:限定
3. 效果归属顺序枚举量:1-首跳 (first),2-末跳 (last),3-平均 (equal),4-全部 (all)
4. 计划状态:1:未提交;2:审核中;3:审核通过;4:审核未通过;5:禁用;6:删除;

- 计划共享表 `plan_share`
  作为队列使用,让定时任务不需要扫表即可获取最新变动的配置方案信息。

字段	类型	长度	说明	示例
plan_id	int	11		
type	int	3	状态 (1)	1

1. 计划状态:1:审核完毕;2:禁用或者删除。当发生上述计划状态变更时,会往该表中插入一条记录

- 自定义来源表 `src_custom_config`
  v1.0 不需要该表!

字段	类型	长度	说明	示例
src_id	int	11	来源 id(auto_increment)	1
plan_id	int	11	计划 id	1
src_name	varchar	255	来源名称	xx
src_url	varchar	255	来源 url	http://lz.taobao.com

  - 注意:自定义来源表 src_id 为 30000 – ….

- 路径配置表 `path_config`

字段	类型	长度	说明	示例
path_id	int	11	路径 id(auto_increment)	1
plan_id	int	11	计划 id	1
name	varchar	255	名称	xx
desc	text	-	描述	…
data	text	-	数据 (1)	…
deleted	int	3	删除标识:1 删除,0 未删除	0

- 数据:存储该路径的详细信息,json 格式。不存储效果页,使用时自动扩展

```
[{
name:"xx", // 名称
url:"http://lz.taobao.com", // url
step: 1   // 步长
},
...
]
```

5.5.2 报表数据相关

- 数据展示表(汇总) rpt_summary

字段	类型	长度	说明	示例
plan_id	int	11	计划 id	1
day	int	11	日期	20120510
dim	int	3	维度	1
ind_<ind_id>	double		指标 (1)	…

- 指标:根据 <ind_config> 配置的指标展开的字段,很多很多

- 数据展示表(详情) rpt_detail

字段	类型	长度	说明	示例
plan_id	int	11	计划 id	1
day	int	11	日期	20120510
dim	int	3	维度	1
src_id	int	11	来源 id(1)	1
src_ext	int	11	来源扩展 id(2)	1
path_id	int	11	路径 id	1
ind_<ind_id>	double		指标 (3)	…

1. 来源 id:当来源是外投广告时,src_ext 是 ad_config 中的 ad_id;当来源是 SPM 时,src_ext 是 spm 值
2. 来源扩展 id:如上
3. 指标:根据 <ind_config> 配置的指标展开的字段,很多很多

- 数据展示表(外投广告指标) rpt_extend_ad

字段	类型	长度	说明	示例
plan_id	int	11	计划 id	1
day	int	11	日期	20120510
ad_id	int	11	外投广告位 id	
ind_<*ind_id*>	double		指标 (1)	…

1. 外投广告指标目前仅使用 id 为 101 和 108 的指标(即外投广告位点击量/点击人数)

5.5.3 辅助前端展现相关

- 指标配置表 `ind_config`

字段	类型	长度	说明	示例
ind_id	int	11	指标 id(auto_increment)	1
ind_name	varchar	255	指标名称	pv
ind_type	int	3	指标类型 (1)	1
ind_desc	text	-	指标描述	…
ind_race	int	3	指标族群 (2)	0
ind_group_id	int	11	指标组 id	1
enabled	int	11	是否启用:1 可用,0 不可用	1
weight	int	11	排序 (3)	1

1. 指标类型:0-int,1-float,2-money,3-percent
2. 指标族群:0-默认,1-外投广告
3. 排序:数值小的在前,可接收负值

- 指标配置表 -组信息 `ind_group_config`

字段	类型	长度	说明	示例
ind_group_id	int	11	指标组 id(auto_increment)	1
ind_group_name	varchar	255	指标组名称	xx
ind_group_desc	text	-	指标描述	…
ind_group_parent_id	int	11	指标组父节点 id,默认 root 为 0	0
enabled	int	11	是否启用:1 可用,0 不可用	1
weight	int	11	排序:数值小的在前,可接受负值	0

- 归属配置表 `belong_config`

字段	类型	长度	说明	示例
belong_id	int	11	归属 id(auto_increment)	1
belong_name	varchar	255	归属名称	xx
belong_desc	text	-	归属描述	…
enabled	int	11	是否启用:1 可用,0 不可用	1
weight	int	11	排序:数值小的在前,可接受负值	0

- 来源配置表 `src_config`

字段	类型	长度	说明	示例
src_id	int	11	来源 id(auto_increment)	1
src_name	varchar	255	来源名称	xx
weight	int	11	排序:数值小的在前,可接受负值	0

  - 注意:src_id 为 1 – 10000

- ad 配置信息 `ad_config`

字段	类型	长度	说明	示例
ad_id	int	11		1
ad_site_name	varchar	255		xx
ad_page_name	varchar	255		xx
ad_position_name	varchar	255		xx
ad_creative_name	varchar	255		xx
ad_activity_name	varchar	255		xx
ad_activity_id	int	11		xx

- 指标切分维度配置表 `dim_config`

字段	类型	长度	说明	示例
dim_id	int	11	指标切分维度 ID	1
name	varchar	255	维度名称	…
desc	varchar	255	维度描述	…
enabled	int	11	是否启用:1 可用,0 不可用	1
weight	int	11	排序:数值小的在前,可接受负值	0