GAR Match using multi machine

功能

该算法为GAR Match的简单并行算法。

伪代码

```
GARMatch(gar,data_graph,n){
    //gar: Q[x](X->Y)
    //data_graph:Data Graph
    //n: number of process
    for (i=0;i<n;i++){
        ReadGraphAndGAR(i);  //each process read data_graph and gar
    }
    InitXYTotalMatch(gar,data_graph); //process 0 cal match only nodes in X and Y
    for (int i=1;i<n;i++){
        if (process_id==0)
            SendMessage(i);    //process 0 send message to other process
        else
            RecvMessage(0);    //other process recv message from process 0
    }

    for (int i=1;i<n;i++){
        CalMatch(gar,data_graph,match_result); //each process cal match
    }

    for (int i=1;i<n;i++){
        if (process_id==0)
            RecvMatchResult(i);    //process 0 recv message(match_result) from other process
        else
            SendMatchResult(0);    //other process send match result to process 0
    }
    return match_result;

}
```

相关说明

ReadGraphAndGAR

每个进程都会读取GAR以及DataGraph。

InitXYTotalMatch

进程0会先计算在GAR的X,Y中出现的节点的所有匹配情况，

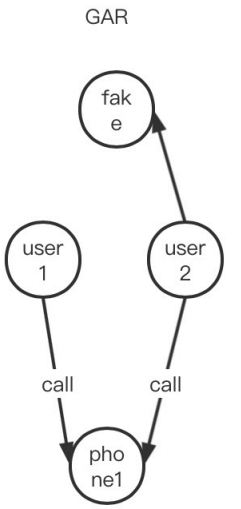
SendMessage RecvMessage

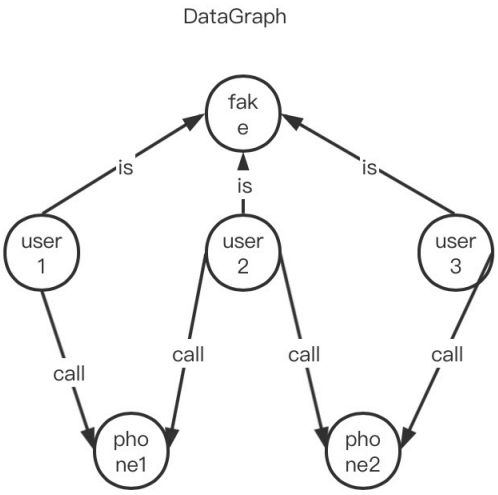
计算完X,Y中相关节点的所有匹配情况后，进程0会将匹配结果发送给其他进程。

信息格式

进程0先给每个进程发送进程0给该进程的匹配数量match_size，接着发送$|X,Y|$条消息，其中$|X,Y|$代表gar的X,Y中出现的节点去重后的个数。每条信息对应$|X,Y|$中的一个vertex u,并发送一个长度为match_size的vector $matchlist$，$matchlist$中每个元素$matchlist[i]$表示在第i个匹配结果中u对应的data graph节点编号。

下面举例说明
考虑以下的GAR以及DataGraph





X:空
Y:EdgeLiteral(user1,fake,is)

经过InitXYTotalMatch过程后，进程0得到的结果为：
$(Queryuser1, Targetuser1), (Queryfake, Targetfake)$
$(Queryuser1, Targetuser2), (Queryfake, Targetfake)$
$(Queryuser1, Targetuser3), (Queryfake, Targetfake)$

若n=4，则1号–3号进程均分得一个匹配
以进程1为例，则发送的消息分别为：
$Message_{match}$：1
$Message_{user1}$：$\{TargetUser1\}$
$Message_{Queryfake}$：$\{Targetfake\}$

CalMatch

每个进程的到对应匹配后，根据已有匹配进行剩余部分匹配

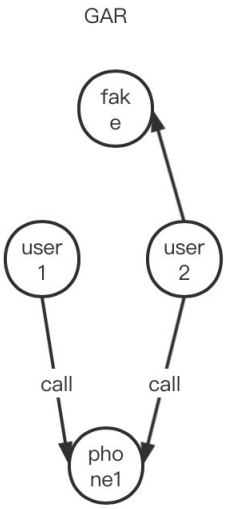
SendMatchResult RecvMatchResult

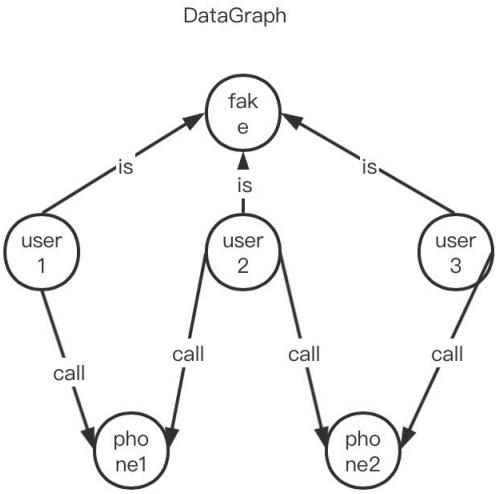
进程1–(n–1)计算完全部匹配之后，将匹配发送給进程0，由进程0进行整合并输出

消息格式

其他进程给进程0发送其他进程给进程0的匹配数量match_size，接着发送$|V|$条消息，其中$|V|$代表gar的pattern点数。每条信息对应V中的一个vertex u,并发送一个长度为match_size的vector $matchlist$，$matchlist$中每个元素$matchlist[i]$表示在第i个匹配结果中u对应的data graph节点编号。

下面举例说明
考虑以下的GAR以及DataGraph





X:空
Y:EdgeLiteral(user1,fake,is)

经过CalMatch之后，进程1–3的结果为：
进程1:
$(Queryuser1, Targetuser1), (Queryfake, Targetfake), (Queryuser2, TargetUser2), (Queryphone1, Targetphone1)$
进程2:
$(Queryuser1, Targetuser2), (Queryfake, Targetfake), (Queryuser2, TargetUser3), (Queryphone1, Targetphone2)$
进程3:
$(Queryuser1, Targetuser3), (Queryfake, Targetfake), (Queryuser2, TargetUser2), (Queryphone1, Targetphone2)$

则进程1发送給进程0的消息为：

$Message_{match}$: 1
$Message_{user1}$: $\{TargetUser1\}$
$Message_{Queryfake}$: $\{Targetfake\}$
$Message_{user2}$: $\{TargetUser2\}$
$Message_{Queryphone1}$: $\{Targetphone1\}$