Hierarchical Influence Maximization

Bonin Grégoire, Duguépéroux Joris (directed by Gross-Amblard David and Allard Tristan)

***Abstract*—Nowadays, social networks offer new ways to spread information, ideas, advertising. In many situations, it appears to be important to reach as many users as possible in these networks.**

Since the efficiency of the diffusion (its impact on users) may depend on the center of interest of the users, we propose an approach based on a hierarchy of topics in order to improve performances of current state-of-the-art algorithms in terms of online calculation time.

This approach shall also enable the exhibition of other topics close to the initial one, with better diffusion potential.

Ultimately, these new algorithm are experimentally tested upon some reduced sets of data, and compared to some state-of-the-art algorithms.

I. Introduction

According to We Are Social's **Digital, Social and Mobile** report for 2015, there are more than 3 billion people connected through social networks. The tight connections between communities offer new possibilities for idea dissemination, or targeted marketing. However, these possibilities to promote ideas and products are to be restricted, since the users are not willing to be flooded at all time. From these observations comes a natural question that has received a wide attention recently : how to select a small subset of people from the social network (the seeds), so that a message submitted to them has the maximum impact over the overall social network (because the message will be forwarded, cited, and so on). This question, known as *Influence Maximization* (IM in the sequel) and its solving are more and more important preoccupations for industries (e.g., how to get a more successful publicity campaign).

However, few recent results suggest that taking into account user interests is a more efficient approach than the standard one [ZSY+12]. For instance, if user u influences v mostly on the *football* topic (as illustrated by their recurrent discussion on the social network), advertising u with a message on *football* is likely to activate the interests of v. On the contrary, advertising a message on *French cuisine* to u may not influence v at all. However, taking these preferences into account can only increase the complexity of this issue.

Early results on this question, which consider a simple social network with untyped friendship, show this issue is NP-hard, and so are most of its numerous variations.

In this paper, we will use a topic-aware model to find the most influential users for a given item, but we will refine this model in order to give both new ways of exploring the path, and allow as much offline calculation as possible.

Indeed, the introduction of topic-awareness in [BBM12] allows to take the preferences of users into account, but it does not take into account similarities between topics, and neither do [CFL+15] or [CLY14], while we propose a model which relies on a hierarchy of topics, to support the introduction of a semantic between topics.

An other way of exploring such similarities between items has been dealt by [ABBBy14], the proposition does not insure theoretical warranties, and requires heavy online computations.

Finally, it is also possible that an item might have a very low potential of spreading, while other close items be far more interesting. Then, it appears to be quite interesting at least to suggest an other close item (within a given distance) that is better than the initial one.

This paper will be organized as follows :

- First, we will expose the formalization of the issue, and the model to be used, which allows both the use of a semantic for queries and alternative items, and an aggressive way to perform offline calculation, to get results with a greatly reduced online computation in section II
- After that, we will introduce some algorithms to solve the various issues, and analyze them theoretically in section III
- Some experiments will expose our results on randomly generated networks, with comparison with other state-of-the-art algorithms in section IV
- Finally, we will expose some related work, before concluding on our contribution in section VI and introducing some possible extensions in our future work in section VII

II. Problem statement

A. Initial structure

To avoid issues formerly given, we propose the introduction of a hierarchy of topics, which sets distances between topics, and would allow some reduction on calculations. This hierarchy would both allow to use efficiently the proximity of topics, but also to suggest some items to the user that would both be close to the initial one and give better results.

Hence, we give the following structure. Initially, we have taxonomy of topics T, with $|T| = z$ the number of topics which are numerated as the order of visit of a deep search exploration. This taxonomy also give distances between topics, so that the edges are weighted edges. We also give an oriented graph $G = (V, E)$ assimilated to a social graph (a node is a user, and an edge, the relation between to users). Each edge between two users u and v of V is associated with a weighted taxonomy $T(u, v)$ (each weight corresponds to the influence of the user on its neighbor on the topic) : see Figure 1.

We denote $pp^t_{u,v}$ the weigh of the topic t on the taxonomy between u and v.

In this model, an item will also be be represented by a weighted taxonomy I : for instance, the estimated profile of

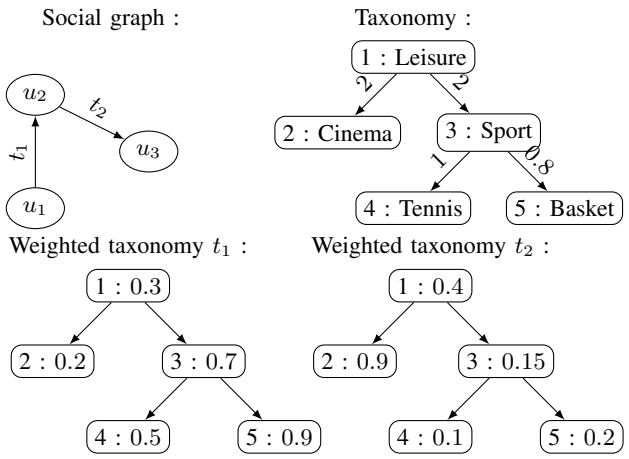

Figure 1. Example of a social graph, a taxonomy, and weighted taxonomies associated

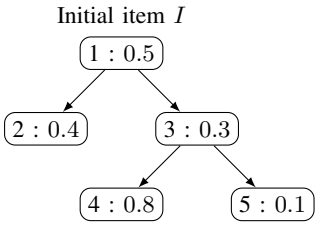

Figure 2. Example of item I initially given

an advertisement (see Figure 2). The weight of the topic t on I will be denote by I^t

From these models, we then define the influence of user on a social graph, for a given item : the activation probability for u to activate its neighbor v for a given item I, denoted by $pp(u, v|I)$ is computed as :

$$pp(u, v|I) = \frac{1}{z} \sum_{t=1}^{z} (pp_{u,v}^t * I^t)$$

B. Influence definition

To compute the influence of a set S of users, we will use the Independent Cascade Model (ICM) [KKT03], which proceeds as follows.

For each query initially given, every user in G is $inactive$, excepted the initial selected set. Then, each active user start activating its neighbors with probability given by the activation probability. If it succeeds, the neighbor becomes $active$, and tries to activate its neighbors too, recursively. Note that a user only activates its neighbors once. The process ends when there are no more inactive user, or when every active user has tried to activate its neighbours.

To evaluate the effect of the process, we will use the the notion of influence spread $\sigma(S|I)$, defined in [ZSY$^+$12] as the expected number of active users after the propagation.

To compute this influence spread, which has been proved to be complex in [CWW10], we will use the maximum influence arborescence (MIA) model ([CWW10]) as an approximation. Under this model, a user u activates an other user v only through the *maximum influence path* between them. A path $P_{u,v}$ between u and v is a non-cyclic sequence of users such that $P_{u,v} = (u = w_0, w_1, ..., w_n = v)$, $(n \geqslant 0)$, where w_n and w_{n+1} are adjacent in G. Through $P_{u,v}$, the activation probability is computed as follows :

$$pp(P_{u,v}|I) = \prod_{k=0}^{n-1} pp(w_k, w_{k+1}|I)$$

Then, the maximal influence path, denoted $u \rightsquigarrow v$ is the maximum of the previous paths :

$$u \rightsquigarrow v = argmax_{P_{u,v}} \{pp(P_{u,v}|I)\}$$

If it is not unique, we select any of the maximum path. MIA model also uses a threshold θ to prune low maximum paths : if $pp(u \rightsquigarrow v|I) < \theta$, v is assumed not to be activated by u.

In order to compute the influence of a set, we notice that we can build a tree structure by assembling the paths from a user u to all users in V, which allows us to compute $pp(u \rightsquigarrow v)$ and $\sigma(u|I) = \sum_{v \in V} pp(u \rightsquigarrow v)$ in general. In paper [CFL$^+$15], similar approaches have been dealt with, in which trees are built from V to a target user v. From these trees, we can deduce the probability that a user v is activated by the set S. To do so, we compute the probability of the complementary event : the probability that none of the user in the set S activates v. Hence, the probability that v is activated by S knowing I, denoted $ap(v|S, I)$ (for activation probability, to follow the notation of [CFL$^+$15]) is computed as

$$ap(v|S, I) = 1 - \prod_{u \in S} (1 - pp(u \rightsquigarrow v|I))$$

Finally, we get the overall influence of a set by summing these set activation probabilities :

$$\sigma(S|I) = \sum_{v \in V} (ap(v|S, I))$$

C. Distance between weighted taxonomies

In order to be able to give some items close to the initial one, we have to use a distance between these items, which are weighted taxonomies.

To take into account the distances between topics, as well as the differences of weights of the topics between the to taxonomies, the idea is to look for the topics for which the weights are different, and then to "move" these weights through the graph until the to taxonomies are similar. Each move cost will be the product of the moved weight by the weight of the edge. Hence, the distance is defined as the minimal sum of all the costs of the moves used to make both taxonomies similar.

One downside of this method is that this distance is only defined when the sum of the weights in both taxonomies are the same, however, this will not be an issue in the following since we will only study these cases.

We can notice that even if many moves are possible, most of them are equivalent, and we will use the one described by the algorithm 1 : for each node which weight is lower in the first taxonomy than in the second, and try to look for the missing

weight by using the lightest paths. Indeed, since we consider cases in which the sum of weights are equals, the balancing of these nodes is sufficient to balance the whole taxonomy.

Algorithm 1: DISTANCE

Input: A taxonomy weighted on its edges $T = (V, E)$, two functions from V to $\mathbb{R}$: $weight_1$ and $weight_2$

Output: The distance d between them

$d \leftarrow 0$;
$Dif \leftarrow \emptyset$;
for $n \in V$ **do**
  if $weight_1(n) < weight_2(n)$ **then**
    $Dif \leftarrow Dif \cup \{n\}$;
while $Dif \neq \emptyset$ **do**
  $n \leftarrow pop(Dif)$;
  $gap \leftarrow weight_2(n) - weight_1(n)$;
  $Open \leftarrow \emptyset$;
  $Close \leftarrow \{n\}$;
  for $u \in neighbours(Close)$ **do**
    $Open \leftarrow insert((u, weight\_edge(n, u)), Open)$;
  while $gap \neq 0$ **do**
    $(u, w) \leftarrow pop(Open)$;
    if $weight_1(u) > weight_2(u)$ **then**
      $gap_u \leftarrow weight_1(u) - weight_2(u)$;
      $min\_gap \leftarrow min(gap_u, gap)$;
      $weight_1(u) \leftarrow weight_1(u) - min\_gap$;
      $weight_1(n) \leftarrow weight_1(n) + min\_gap$;
      $gap \leftarrow gap - min\_gap$;
      $d \leftarrow d + w * min\_gap$;
    $Close \leftarrow Close \cup \{u\}$;
    for $v \in neighbours(u)$ **do**
      if $v \notin (Close \cup Open)$ **then**
        $Open \leftarrow insert((v, w + weight\_edge(u, v)), Open)$;
return d;

D. Problem definition

Hence, from this, we can finally give the following definition.

Definition 1: HIERARCHICAL TOPIC-AWARE INFLUENCE MAXIMIZATION (HTIM)
Given a hierarchical topic-aware social graph $G = (V, E)$, and a HTIM query $Q = (I, k, d)$, we look for a seed set S^* and a weighted item I^*_{out} such that $(S^*, I^*_{out}) = argmax_{(S,I_{out})}\sigma(S|I_{out})$, where $S \subset V$, $|S| = k$, $distance(I, I^*_{out}) \leqslant d$

E. Variations on the problem

However, it is not hard to see that, at is is proposed, the current problem has a problem coming the selection of a maximum between items, which are uncountable, with no specific properties which could allow us to prune them.

Hence, to overcome this issue, we will introduce the following simplifications :

- First, in order to perform as much offline calculation as possible, we will study a single-topic restriction : items shall be restricted to a single topic, which means that both the input taxonomy I and the output I^* will be such that $\exists! t \in [\![1; z]\!], I^t = 1 \wedge \forall t' \in [\![1; z]\!] \neq t, I^t = 0$. Every weigh will be equal to 0 except the more significant one which shall be 1. (III-B and III-C)
- Then, we will extend this limitations, by allowing several topics to be affected to 1, in a discrete multi-topic approach. Hence, each weigh of the taxonomy will be either 0 or 1 : $\forall t \in [\![1; z]\!], I^t = 0 \vee I^t = 1$. (III-D)

These simplifications are all-the-more justified that it is not always clear for a client to know exactly how to specify an item : giving the main topics is often easier to use and sufficient to describe an advertisement or an item.

III. CONTRIBUTION

A. Complexity

However, even with the MIA model, the evaluation of a HTIM query is still very complex. By a reduction of the conventional problem without topics ([KKT03] [CWW10]) inspired by [CFL$^+$15], we can prove that this problem is NP-hard.

Theorem 2: Hierarchical Topic-Aware Influence Maximization under MIA influence computation model is NP-hard.
PROOF: Given an instance of the conventional influence maximization problem, we can build an instance of HTIM with $z = 1$ topics, and $d = 0$ the minimum distance to the original topic such that the solution to our problem is the solution to the conventional problem. As the conventional problem is NP-hard, we prove the lemma.

Despite this issue, [CFL$^+$15] highlights that some properties of $\sigma(S|I)$ enable good approximations to get efficient algorithms. First, the influence is monotonic : for $S_1 \subset S_2, \sigma(S_1|I) \leqslant \sigma(S_2|I)$. Then, the influence is submodular : for $S_1 \subset S_2$ and a user u, $\sigma(S_1 \cup u|I) - \sigma(S_1|I) \geqslant \sigma(S_2 \cup u|I) - \sigma(S_2|I)$.

Based on these properties, a greedy algorithm can achieve an approximation ratio of $1 - \frac{1}{e}$ [KKT03].

B. Online single-topic algorithm

In the case of an item characterized by a simple topics, the general idea of our computation is to compute the optimal seed of each item within the authorized distance to the original item with a greedy algorithm, and to pick up the better result, as illustrated in the algorithm 2. In the case of a single-topic item, this computation will then be performed in z times the cost of the greedy algorithm.

To perform the computation of the greedy algorithm, we will first define the marginal influence denoted $\Delta\sigma(u|S, I)$ of a node u which does not belong to a set of seeds S, knowing an item I. This marginal influence is defined as $\Delta\sigma(u|S, I) = \sigma(S \cup \{u\}|I) - \sigma(S|I)$.

To compute $margin(u|S, I)$, we first compute $\sigma(S \cup \{u\}$, before subtracting $\sigma(S)$. To do so, we simply use the primary

Algorithm 2: ONLINE SINGLE TOPIC ALGORITHM

Input: A single-topic item I, a distance d, a hierarchy of topics H, a number of seeds k, a graph $G = (V, E)$

Output: A single-topic item I_{out}, a queue of seeds S, the value of the influence $\sigma(S|I)$

$I_{out} \leftarrow null$;
$S \leftarrow \emptyset$;
$\sigma \leftarrow 0$;
/* We choose the best topic among the interesting ones */
for t *in* H **do**
  if $distance(t, I) \leqslant d$ **then**
    if $Greedy(t, k, G).\sigma > \sigma$ **then**
      $I_{out} \leftarrow t$;
      $S \leftarrow Greedy(t, k, G).S$;
      $\sigma \leftarrow Greedy(t, k, G).\sigma$;
    else
      continue;
return I_{out}, S, σ;

definition of σ as the sum of the $ap(v|S \cup \{u\}, I)$ for all v in V.

However, we can notice that it is easy to compute $ap(v|S \cup \{u\}, I)$ if we already know $ap(v|S, I)$ and $pp(u \rightsquigarrow v|I)$ for all v in V, thanks to the following equality :

$$\begin{aligned} ap(v|S \cup \{u\}, I) &= 1 - \prod_{w \in S \cup \{u\}} (1 - pp(w \rightsquigarrow v|I)) \\ &= 1 - (1 - pp(u \rightsquigarrow v|I)) * \prod_{w \in S} (1 - pp(w \rightsquigarrow v|I)) \\ &= 1 - (1 - pp(u \rightsquigarrow v|I) * (1 - ap(v|S, I)) \end{aligned}$$

Therefore, to get more efficient computation of $margin$, we will consider that $\Delta\sigma(u|S, I)$ also takes in input $\sigma(S|I)$ and $ap(v|S, I)$, and gives $ap(c|S \cup \{u\}, I)$ for all v in V in output.

In the following, we will use the notation $ap(S, I)$ to denote the knowledge of $ap(v|S, I)$ for all v in V.

Hence, we get an algorithm linear in time on the number of nodes V, as described in the algorithm 3.

Algorithm 3: MARGIN

Input: A seed u, a queue of seed S, a graph $G = (V, E)$, an item I, $\sigma(S)$ the influence of S, the activation probability $ap(S, I)$

Output: $\Delta\sigma(u|S, G, I)$ the value of the marginal influence of u, the activation probability $ap(S \cup \{u\}, I)$

$\sigma(S \cup u) \leftarrow 0$;
for v *in* V **do**
  $ap(v|S \cup \{u\}) \leftarrow (1 - (1 - pp(u \rightsquigarrow v|I)) * (1 - ap(v|S, I))$
  $\sigma(S \cup u) \leftarrow \sigma(S \cup u) + ap(v|S \cup \{u\})$;
return $\sigma(S \cup \{u\}) - \sigma(S), ap(S \cup \{u\}, I)$;

From there, we can build the greedy algorithm proposed in [KKT03] and [CFL$^+$15] that will proceed in $|S|$ iterations and will, for each iteration, take the node with the higher marginal influence knowing the seeds that have already been selected, as presented in the algorithm 4.

Algorithm 4: GREEDY

Input: A topic t, a number of seeds k, a graph $G = (V, E)$

Output: A queue of seeds S of size k, the value of the influence $\sigma(S|I)$

$S \leftarrow \emptyset$;
$\sigma \leftarrow 0$;
/* Initialization of influence of S on each node */
for v *in* V **do**
  $ap(v|\emptyset, I) \leftarrow 0$
/* iterate for all seeds */
for $i = 1$ *to* k **do**
  /* find the seed with maximum marginal influence */
  $maxSeed \leftarrow null$;
  $maxMargin \leftarrow 0$;
  for u *in* $V \setminus S$ **do**
    if $margin(u|S, G, t, ap(S, I)).\Delta\sigma > maxMargin$ **then**
      $maxSeed \leftarrow u$;
      $maxMargin \leftarrow margin(u|S, G, t, \Delta\sigma, ap(S, I)).\Delta\sigma$;
      $ap(S \cup \{u\}, I) \leftarrow margin(u|S, G, t, \sigma, ap(S, I)).ap$
  /* Add the seed to the set and sum the influence */
  $S \leftarrow add(maxSeed, S)$;
  $\sigma \leftarrow \sigma + maxMargin$;
return S, σ;

As for the complexity, this algorithm performs k iterations, which all use $|V|$ times the algorithm 3. Hence, this greedy algorithm runs in $\mathcal{O}(k * |V|^2)$.

C. Offline single-topic optimization

With the previous algorithms, all the computation is performed online, so that a user will have to wait for a long time before having the results of his query. Therefore, we try to diminish the online computation, by making most of it offline, and consequently lighter for the user.

The main idea is that, as we consider single-topic items, the number of available topics is quite small, and we can therefore consider performing the greedy algorithm on each possible input in offline computation, and store the results. Hence, the online computation would mainly consist in applying the algorithm 2, considering every greedy call as a look in memory.

This idea is quite simple, but it does not yet take into account the number of seeds required. However, as we perform a greedy algorithm, this parameter is no big issue thanks to the absence of back-tracking.

Indeed, as we perform the greedy algorithm, a selected seed cannot be unpicked, so that a given run $Greedy(t, k, G)$ also performs the runs $Greedy(t, k', G)$ for all $k' \leqslant k$.

This property allows us make a single run of the greedy algorithm, with $k = |V|$, instead of making multiple runs for each possible k. During this run, we will then store the nodes in the order they are selected as seeds, and associate, for each of them, the influence of the seed in which they first appear.

Hence, by performing this greedy algorithm for each topic, we finally get an offline phase that time complexity is $\mathcal{O}(z * |V|^3)$ where z is the number of topics. By doing so, the online complexity in time is then reduced to $\mathcal{O}(z)$, since the only computation we have to do is to check distances, and check values for each topic (such a checking is available in constant time in hash table for instance).

We can also notice that, if we do apply this offline phase, there is no more need to keep the social graph in memory as every information is stored in the offline structure. In particular, there is no more need to store the edges and their associated weighted taxonomies, it is now enough to store the result of the offline phase, which is an associative list of size $|V|$ for each topic : $\mathcal{O}(z * |V|)$.

D. Multi-topic optimization

The previous vision of an item is however a huge reduction of what a user could want to spread in a network. Indeed, reducing an item to a single topic introduces both a bias in the computation (because an item often touches more than one topic) and an additional choice for the user, to select one topic : for instance, if a soda firm wants to spread an advertising dealing with football, should it select "football" or "soda" as the unique topic ?

To solve this issue, we extend our vision of an item to a set of topics. For this variation of the HTIM query, we shall only consider as valid output the items with as many topics as the input item.

With this extension, the online procedures previously developed still work, but the offline approach is much more compromised. Indeed, in order to take every item into account in an offline approach, it would require to compute the greedy algorithm for each possible item, which means 2^z times. Even in an offline approach, such a complexity in time ($\mathcal{O}(2^z * |V|^3)$) or in space ($\mathcal{O}(2^z * |V|)$) isn't possible, and we have to conceive other way to lighten online computation.

To do so, we will use the $best - effort$ framework, developed in [CFL$^+$15], which allows us to keep theoretical warranties while introducing heuristics to speed the computation up. This framework algorithm will then be used as a substitute for the greedy algorithm in the algorithm 2.

We can also notice that the extension from single-topic items to multi-topic items also increase the complexity of the algorithm 2. Indeed, the number of possible items increase from z to $\binom{z}{n}$, where n is the number of topics of the initial item.

1) Best-effort framework: The best-effort scheme relies on an heuristic which will be called h, which is an upper approximation of the marginal influence of a node over a network.

The idea is similar to the greedy in the fact that once a node is selected as seed, it is definitive. However, the main difference is that we would like not to compute the marginal influence at each iteration for each node. Hence, at each step, we consider that each which is not in the seed node already has a score which can be either the real value of the marginal influence (the state of the node is $exact$), an over approximation from this step (the state of the node is $bounded$), or an over approximation from a previous step (the state of the node is $initial$). Then, we select the node with the best score. If the state of this node is $exact$, then it is added to the seed. If it is $bounded$, we compute the real marginal influence and change its state to $exact$, before putting it back to the sorted list of nodes. If it is $initial$, we compute an over approximation of its marginal influence and change its state to $bounded$, before putting it back to the sorted list of nodes. Each time a node is added to the seed, the state of each node which is not in the seed is set to $initial$. This whole method results in the algorithm 5.

This algorithm will hence only compute the marginal influence of the nodes which over estimation is above the maximal marginal influence. This method works thanks to the submodularity of the influence ($\Delta\sigma(u|S) \geqslant \Delta\sigma(u|S \cup \{v\})$), which allows to keep the score of the previous step as over approximation so long as the real heuristic has not been computed.

As for the complexity in worst case in time for this algorithm, we compute k iterations which would all compute, on the worst case, the marginal influence of each node, and their heuristics (which cost is supposed to be smaller than the real computation), so that we end up with $\mathcal{O}(k * |V|^2)$, similarly to the greedy algorithm, if the heuristic is very poorly informative.

2) Offline heuristic: In our case, we want to get an heuristic which would be as accurate as possible, and work in very short online time. However, taking into account the whole complexity of the possible items is not possible due to the high complexity of the input. Hence, we consider using the sum of the influence over the topics selected in the item.

However, as it is impossible to store the marginal influence for each node and each possible set ($\mathcal{O}(|z| * |V| * 2^{|V|})$ space), the solution we propose is to store a value for each topic, each node, and each size of set possible, which means a storage in $\mathcal{O}(|z| * |V|^2)$.

Hence, for a given topic t, size of seed k and node u, the value stored will be the marginal influence of the u over set of seed selected with the greedy algorithm on t, with a slight modification. Indeed, if the u appeared to be in the resulting set, the marginal influence would be meaningless. That's why the computation of the greedy algorithm will select seeds in $V \setminus \{u\}$ instead of V. In practice, it is enough to use $V \setminus (S \cup$

Algorithm 5: BEST EFFORT ALGORITHM

Input: A multi-topic item I, a number of seeds k, a graph $G = (V, E)$
Output: A queue of seeds S of size k, the value of the influence $\sigma(S|I)$

$\sigma \leftarrow 0$;
$S \leftarrow \emptyset$;
for u *in* V **do**
  $H \leftarrow insert((u, \sigma(u|I)), H)$;
for v *in* V **do**
  $ap(v|\emptyset, I) \leftarrow 0$;
for $i = 1$ *to* k **do**
  for $u \in H$ **do**
    $u.state \leftarrow initial$;
  $u \leftarrow H.head$;
  while $u.state \neq initial$ **do**
    $u \leftarrow H.pop()$;
    if $u.state = initial$ **then**
      $u.state \leftarrow bounded$;
      $H \leftarrow insert((u, h(u)), H)$;
    else if $u.state = bounded$ **then**
      $u.state \leftarrow exact$;
      $H \leftarrow insert((u, margin(u|S, G, I, \sigma, ap(S, I)).\Delta\sigma), H)$;
  $S \leftarrow S \cup \{u\}$;
  $\sigma \leftarrow \sigma + margin(u|S, G, I, \sigma, ap(S, I)).\Delta\sigma$
  $ap(S \cup \{u\}, I) \leftarrow margin(u|S, G, I, \sigma, ap(S, I)).ap$
return S, σ;

$\{u\}$) instead of $V \setminus S$ line 8 in the algorithm 4. In practice, we will perform only one computation of the greedy algorithm by node (and not $|V|$ for all the possible size of set), by storing the marginal influence at each step.

Thanks to these computations, there is now a value of each node, each topic and each possible set. However, this computation appears to be very long : we have compute the greedy algorithm for each node in $|V|$, which leads to $\mathcal{O}(|z| * |V|^4)$.

This heuristic would have been very interesting if it worked, however, we realized to late that our assumption according to which the sum of the influence on each topic was greater than the influence of the overall item wasn't necessarily right. Indeed, if we consider a network in which the weights on every edge and for each topic are very low, it could appear that the influence is null for each topic due to the MIA threshold, even if the influence for more than one topic appeared to be greater than the same threshold. Then, our heuristic would give a null result, which is lower than the non-null real value.

IV. RESULTS

A. Data

In order to test our algorithms, we first tried to collect real social graphs. However, the lack of data which would be both openly accessible and easy to use, taxonomies and weights for each topic for each relation, lead us to focus on social graphs based on research and relations between researchers.

Indeed, in research, using publications as traces for influence and network is very interesting since we can easily get both a taxonomy, which is built upon some classification used in revues such as ACM Computing Classification System ([Coub], [Coua]), or Web of Science.

Due to the absence of correct indexing to retrieve data, we haven't been able to collect more than a few hundreds of exploitable articles, manually, from the Web of Science search engine ([Lee]). Thanks to this data, we have built a social graph, using the following balancing : for a given user u and its neighbor v, and a topic t, we use $pp^t_{u,v}$ equal to the number of articles from v in which an article from u is quoted about the topic t, divided by the amount of quoted articles from v about t. Furthermore, as we didn't have any way to get an objective distance between the topics, we arbitrary set the weight of every edge of the initial taxonomy 1.

However, this approach didn't give any good result, due to the little amount of articles collected : the number of articles which are both quoted and present in our data being very low, only little weights were not null.

As a consequence, we decided to test our algorithms on randomly generated graphs, with randomly generated weights. Indeed, as there is no state-of-the-art comparison for our algorithm and the problem we formulated, the only interesting point of these experiments is to determine the time improvement introduced by the offline computation.

B. Experimental measures

The network we built contains 400 nodes, all connected to a random amount of neighbors between 0 and 10, and a MIA threshold of 0.5, which is the maximum size we could run on the machines at our disposal.

As a taxonomy, we took a tree of depths 3, with 56 topics, inspired by a subtree of the taxonomy proposed by Web of Science.

With these parameters, we have run experiments with three different topics and for three different sizes of seed : 25, 50, 100.

Even if the choice of the topic does not matter much since the data is randomly generated, we applied the computation on three different topics chosen to be as different as possible : one of them is close to the root of the taxonomy, on of them is the deepest leaf of the taxonomy, and the last one is right between the two others, at the median depth.

The diagrams 3, 4 and 5, are organized as follows : each of them represents a topic, and the results of both greedy online algorithm and greedy offline algorithm for the three seed sizes.

For the online time, we can notice the tremendous profit brought by the offline computation, which would both be useful if there were to be many demands, or for a bigger network.

We have also measured the time required to build the data required to have the offline version of the greedy algorithm work, which both independent of the topic and of the seed size

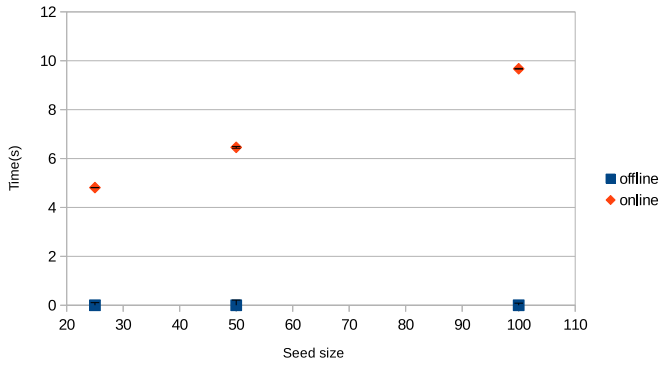

Figure 3. Online time of calculation for the topic in deepest position in the taxonomy

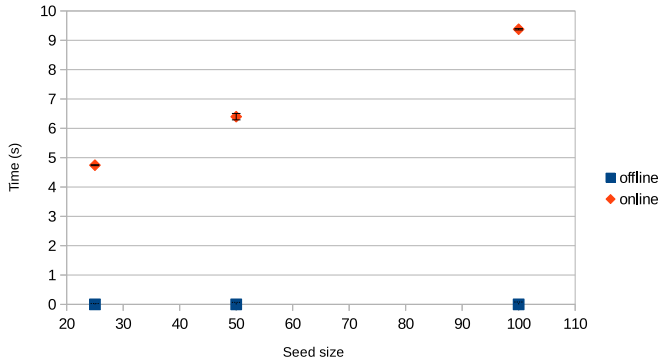

Figure 4. Online time of calculation for the topic in medium position in the taxonomy

: 1750.02 ± 30.565 seconds (roughly half an hour). This value may well look very big considering the gain of few seconds for each query, but remembering the fact that it includes computation for all topics, for all number of seeds may put things into an better perspective.

To perform the whole algorithm 2 to answer an HTIM query, and not only a TIM query, we have to multiply the number of topics within the distance between to the original one. In particular, for a large distance, the computation of a HTIM query with the online algorithm may well take 56 times more time than indicates on the graphics, up to few minutes according to the size seed expected, while the offline version would still give an answer within a very little time.

However, it is to be noticed that these small-sized experiments on relatively weak computers may not be representative for a real-size network, and that this computation is to be completed by further studies.

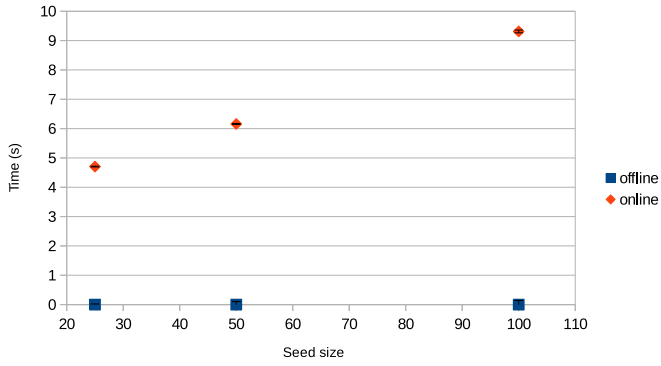

Figure 5. Online time of calculation for the topic in root of the taxonomy

V. Related work

Maximizing the spread of influence through a social network [KKT03]: first proposes the Linear Threshold model and the Influence Cascade model as a diffusion model through social networks seen as graphs. Furthermore, this it also first introduces the greedy algorithm and its $1 - \frac{1}{e}$ approximation to solve the letter. However, it does not take any kind of topics at all in this model.

Topic-Aware Social Influence Propagation Models [BBM12]: proposes a description of a topic-aware model, and the Topic-Aware Influence Cascade (TIC) model, and some way to gather information on a network to fill the model, and be able to create such a graph.

Online Topic-Aware Influence Maximization [CFL$^+$15]: introduces the greedy algorithm, the best-effort algorithm and many heuristics in the TIC model, so as to allow an efficient online approach of the solving of the Topic-Aware Maximization queries, while keeping the theoretical warranties of the greedy algorithm developed in [KKT03].

Efficient Topic-aware Influence Maximization Using Preprocessing [CLY14]: gives some way to perform as much offline computation as possible. Even if these propositions are very efficient, the do not keep as much warranties as the greedy algorithm on the quality of the answers.

Online topic-aware influence maximization queries [ABBBy14]: similarly propose some offline computation, which uses similarities between close topics, to extract some representative items and perform computation upon them. This approach is interesting in the fact that it looks further into the similarities between topics, but it loses nevertheless theoretical warranties.

VI. Conclusion

In this work, we introduced a new problematic, the HTIM in order to allow more flexibility in the choice of an initial item than a simple TIM query.

From this problematic, we gave a first simplification in order to have some result with both acceptable time computation and theoretical warranties. To allow some improvement in the online time, we also introduced a way to perform most of the computation offline. Then, we lightened the initial simplification, so as to have more precision thanks to a better flexibility in the input.

Because of the lack of material, we have been forced to generate network and connection randomly. Furthermore, due to poor computation power, an evaluation of our algorithm on real-size data appeared to be impossible.

Yet, the experiments we managed to do on small data are encouraging, and already show how promising are our offline

approach is compared to online exclusive computation. Even if these performance may not be representative for larger scales we are still confident that these performance will keep being a good approach on real-size networks.

VII. Future Work

In the future, many other approaches could be interesting. A first approach would be to exploit the heuristic we developed in section III-D2. Indeed, even if this heuristic is not an upper approximation of the marginal influence, it may still give a good approximation of the influence, such that the computation would be fully offline, with only little loss on the performances in precision. However, such a work would still require to give some further experiments, and above all, some theoretical warranties about this approximation.

Some other approaches using the influence of nodes on the network could also be interesting, so as to give some rating to each node, depending on the topic or item. Indeed, the creation of such a rating could also give some good approximation with a combination of the better rated nodes on the input item. Some further computation would still be required in order not to take nodes which impact the same users, but this could be far lighter than the initial computation or the greedy algorithm.

References

[ABBBy14] Cigdem Aslay, Nicola Barbieri, Francesco Bonchi, and Ricardo Baeza-yates. Online topic-aware influence maximization queries. In *In EDBT*, pages 295–306, 2014.

[BBM12] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. Topic-aware social influence propagation models. In Zaki et al. [ZSY+12], pages 81–90.

[CFL+15] Shuo Chen, Ju Fan, Guoliang Li, Jianhua Feng, Kian-lee Tan, and Jinhui Tang. Online topic-aware influence maximization. *Proc. VLDB Endow.*, 8(6):666–677, February 2015.

[CLY14] Wei Chen, Tian Lin, and Cheng Yang. Efficient topic-aware influence maximization using preprocessing. *CoRR*, abs/1403.0057, 2014.

[Coua] James; Glinert Ephraim; Horton Thomas; Mead Nancy; Ralston Anthony; Rada Roy; Rodkin Craig; Rous Bernard; Tucker Allen; Wegner Peter; Weiss Eric; Wierzbicki Carol Coulter, Neal ; French. Computing classification system 1998: Current status and future maintenance report of the ccs update committee.

[Coub] Neal Coulter. Acm's computing classification system reflects changing times.

[CWW10] Wei Chen, Chi Wang, and Yajun Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 1029–1038, New York, NY, USA, 2010. ACM.

[KKT03] David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In Lise Getoor, Ted E. Senator, Pedro M. Domingos, and Christos Faloutsos, editors, *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003*, pages 137–146. ACM, 2003.

[Lee] Sul H. Lee. Citation indexing and isi's web of science (discussion of finding literature manually. description of citation indexing, and web of science.).

[ZSY+12] Mohammed Javeed Zaki, Arno Siebes, Jeffrey Xu Yu, Bart Goethals, Geoffrey I. Webb, and Xindong Wu, editors. *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*. IEEE Computer Society, 2012.