Target Set Selection

Analisi dell'applicazione sulla rete di contatti della piattaforma Twitch

Studenti:

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

Docenti:

Adele A. Rescigno
Luisa Gargano

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

Indice

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

1. Introduzione

Il lavoro svolto ha l'obiettivo di analizzare il modo in cui una nuova idea o un prodotto riesca a diffondersi completamente all'interno di una rete, partendo da un insieme di nodi S inizialmente attivi.

Nello specifico, la problematica analizzata è conosciuta come Target Set Selection (TSS) e consiste nel trovare il più piccolo insieme di nodi S che, inizialmente attivi, riescano a diffondere il prodotto nella rete facendolo adottare da tutti gli altri nodi.

Per l'analisi è stato scelto come dataset la rete delle relazioni di amicizia tra gli utenti di nazionalità britannica di Twitch, una famosa piattaforma di game streaming.
Tale rete è costituita da circa settemila nodi e più di trentacinquemila edges.

Successivamente, è stato implementato un algoritmo risolutivo in Python che utilizza la libreria Snap.py per l'elaborazione e l'analisi di reti di grandi dimensioni, e che restituisce in output l'insieme S e la sua cardinalità.

L'esecuzione di tale algoritmo sul grafo iniziale utilizza diversi approcci, ciascuno dei quali inizializza in maniera specifica i threshold dei nodi della rete.
Inoltre, alcuni di questi approcci utilizzano la decisione differita per ridurre la dimensione del grafo iniziale.
Tali approcci saranno accuratamente descritti successivamente nel capitolo dedicato *(Capitolo 4)*.

Infine, sono stati analizzati i diversi risultati ottenuti applicando i vari approcci.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

2. Target Set Selection

Il Target Set Selection (TSS) è un problema NP-hard ed è piuttosto difficile da approssimare.

Sia G=(V, E) un grafo di n vertici, in cui d_v indica il grado del vertice *v*, e t_v indica il threshold associato al vertice *v*, ovvero il numero minimo di nodi adiacenti al nodo *v* che devono essere attivi per consentire a *v* di attivarsi allo step successivo.

Consideriamo un processo in cui, al tempo t_0, un insieme S di nodi siano attivi e, ad ogni intervallo di tempo successivo $t_i > t_0$, ogni nodo *v* che abbia almeno t_v nodi adiacenti attivi, si attivi a sua volta.

L'insieme S sarà valido se alla fine del processo tutti i nodi del grafo diventano attivi.

La problematica del Target Set Selection si pone l'obiettivo di trovare il più piccolo insieme di nodi S che, inizialmente attivi, riescano ad attivare alla fine del processo tutti gli altri nodi della rete.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

3. Dataset scelto

Il dataset selezionato per lo studio è stato prelevato dalla repository di dataset pubblici messi a disposizione da https://snap.stanford.edu/data/index.html.

In particolare è stato utilizzato un dataset relativo alle relazioni di "amicizia" sulla piattaforma di game streaming Twitch, reperibile presso il seguente link: https://snap.stanford.edu/data/twitch-social-networks.html. Tale dataset è stato costruito a partire dalle informazioni sulla rete di amicizie raccolte a Maggio 2018.

Twitch è una piattaforma nata per lo streaming di videogiochi, in cui gli utenti possono assumere due ruoli:

1. Streamer: condivide in diretta prestazioni videoludiche che possono essere accompagnate da un commentario live, interazione con la chat di spettatori e spesso anche da una facecam;
2. Spettatore: visualizza i contenuti condivisi in diretta da uno streamer potendo comunicare con gli altri utenti tramite una chat in cui interagiscono tutti gli altri spettatori dello streamer, nonché con lo streamer in questione.

Negli ultimi anni si ha avuto un grande incremento di attività su tale piattaforma, portando alla nascita anche di nuove tipologie di streaming quali "just chatting" (in cui un utente crea in live una discussione con la propria utenza, la quale può comunicare tramite una chat pubblica) e "food and drinks" (nel quale un utente condivide in live un'esperienza culinaria). Ciò ha portato ad un ampliamento nel numero di utenti che approcciano la piattaforma da streamer, ma anche nel numero di utenti che partecipano attivamente come spettatori seguendo tali stream in diretta.

Le motivazioni dietro la volontà di studiare e comprendere le relazioni tra utenti di tale piattaforma deriva principalmente dall'aumento di utenza e attività su Twitch che si è avuto negli ultimi anni.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

Il dataset presenta le seguenti caratteristiche:

	DE	EN	ES	FR	PT	RU
Nodes	9,498	7,126	4,648	6,549	1,912	4,385
Edges	153,138	35,324	59,382	112,666	31,299	37,304

Tabella 1 - Descrizione del dataset di Twitch in base al numero di nodi e di archi, e in base alla nazionalità degli utenti.

Sono raccolte nel dataset informazioni relative a 6 reti di amicizia su Twitch, in particolare gli utenti vengono raggruppati a seconda della lingua utilizzata sulla piattaforma (DE: tedesco; EN: inglese(GB); ES: spagnolo; FR: francese; PT: portoghese; RU: russo).

Nel dataset scelto i vertici rappresentano gli utenti, mentre gli archi rappresentano una mutua relazione di amicizia sulla piattaforma.

Studiare tali reti può portare ad una migliore comprensione delle relazioni tra gli utenti, migliorando anche l'efficacia di pubblicità mirate nonché la diffusione di informazioni.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

4. Studio effettuato

Lo studio effettuato ha avuto lo scopo di comprendere quanto efficacemente è possibile diffondere influenza nella rete.

In particolare è stata implementata la versione del **Target Set Selection** (TSS) illustrata in uno studio portato avanti da alcuni docenti e ricercatori dell'Università degli Studi di Salerno: "*Discovering Small Target Sets in Social Networks: A Fast and Effective Algorithm*", di Gennaro Cordasco, Luisa Gargano, Marco Mecchia, Adele A. Rescigno e Ugo Vaccaro, pubblicato il 26 Ottobre 2017.

Tale implementazione è stata utilizzata per analizzare il comportamento del TSS sulla rete di amicizie di Twitch descritta nel capitolo 3.

Data la taglia del dataset utilizzato, lo studio è stato limitato all'analisi del grafo delle amicizie sull'utenza anglofona di Twitch (EN_GB).

Si è deciso di applicare l'algoritmo del TSS utilizzando differenti valori per il threshold di attivazione dei nodi.

Gli approcci di assegnazione dei threshold di attivazione ai nodi sono stati due:

1. *Threshold statico*: ad ogni nodo è stato assegnato un threshold fisso ed uguale per ogni nodo della rete;
2. *Threshold proporzionale al grado*: ad ogni nodo è stato assegnato un threshold che varia a seconda del grado del nodo considerato (un nodo si attiva se una certa frazione dei nodi adiacenti è attivo).

Nel primo caso, sono stati utilizzati threshold X che variano da 1 a 10 (es. un nodo si attiva se ha X vicini attivi).
Nel secondo caso invece sono stati utilizzati threshold proporzionali al grado dei nodi, facendo variare tale proporzione da 1/10 ad 1/2 (es. un nodo si attiva se ha 1/X dei propri vicini attivi).

Successivamente è stata estesa l'analisi considerando una distribuzione di probabilità sugli archi. Essa rappresenta la probabilità di attivazione degli archi e di conseguenza la loro probabilità di essere preso in considerazione come "nodo attivo adiacente" da valutare nei confronti del proprio threshold di attivazione.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

L'approccio probabilistico è stato analizzato sia con l'assegnazione di un threshold statico che con l'assegnazione di un threshold proporzionale al grado, confrontando i risultati con i corrispettivi senza la presenza della probabilità di attivazione.

Per poter lavorare con un processo dinamico di questo tipo è stato utilizzato l'approccio della "**Decisione differita**", riducendolo ad un processo statico più facilmente analizzabile.
In particolare è stata assegnata una probabilità di attivazione generata in maniera casuale per ogni arco, quest'ultima equivale alla probabilità di attivazione del suddetto arco per "diffondere" influenza.
Per applicare la decisione differita è stato generato un ulteriore valore casuale per ogni arco, se quest'ultimo risulta essere maggiore della probabilità di attivazione dell'arco, esso verrà preso in considerazione per l'esecuzione del TSS, altrimenti se quest'ultimo risulta essere minore della probabilità di attivazione dell'arco, esso verrà escluso dal grafo per l'esecuzione del TSS.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

5. Implementazione

Si procede in questa sezione ad analizzare le parti salienti dell'implementazione fornita per l'algoritmo del TSS. Implementazione che può essere reperita presso la repository pubblica "https://github.com/jackcaj97/RetiSociali".

I 4 casi di analisi presi in considerazione nello studio sono stati implementati in 4 funzioni distinte, così da mantenere un sufficiente livello di modularità nel codice.
In ciascuna di esse, l'esecuzione presenta la medesima struttura:

1. Caricamento del grafo (come grafo non direzionato: "*TUNGraph*") a partire dal dataset;
2. Si inizializza una hash table per associare un threshold di attivazione ad ogni nodo del grafo;
3. Viene costruita la hash table con i valori di threshold dei nodi;
4. Si forniscono il grafo e la hashtable in input all'implementazione del TSS;
5. Si ottiene il seed set di cui si analizza la dimensione.

Nelle versioni che considerano la presenza della probabilità di attivazione per ogni arco, vi sono alcune aggiunte rilevanti:

3.1 - Si costruisce una seconda hash table utile ad assegnare ad ogni arco una probabilità di attivazione (generata in maniera casuale tramite la funzione "random" del package "random");
3.2 - Si itera 10 volte la costruzione di un nuovo grafo tramite la decisione differita (dettagli specificati successivamente nella funzione "deferred_decision");
3.3 - Per ognuna della iterazioni si utilizza il grafo ottenuto come input per la funzione che implementa il TSS;
3.4 - Si effettua una media sui risultati delle 10 iterazioni per la dimensione del seed set selezionato.

In particolare per l'assegnazione dei threshold, per l'assegnazione delle probabilità di attivazione e per l'applicazione della decisione differita sono state implementate delle funzioni ad hoc:

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

set_static_threshold(graph, hash_table, value)

Descrizione:

Costruisce la hash table relativa ai threshold di ogni singolo nodo del grafo, assegnando a ciascuna key (id del nodo) un valore statico:

Corpo della funzione:

```
_____________________________________
for v in graph.Nodes():
    hash_table[v.GetId()] = value
return hash_table
_____________________________________
```

set_degree_based_threshold(graph, hash_table, proportion)

Descrizione:

Costruisce la hash table relativa ai threshold di ogni singolo nodo 'x', assegnando a ciascuna key (id nodo) un valore proporzionale al grado del nodo 'x':

Corpo della funzione:

```
_____________________________________
for v in graph.Nodes():
    hash_table[v.GetId()] = math.ceil(proportion * v.GetOutDeg())
return hash_table
_____________________________________
```

Da notare che è stato scelto di assegnare la parte superiore della frazione di nodi necessari per l'attivazione (evitando dunque l'assegnazione di 0 come threshold). Tale scelta è stata presa in seguito ad un'analisi della rete utilizzata. La maggior parte dei nodi della rete infatti presenta un grado molto basso (es. 1, 2 o 3), per ogni proporzione presa in considerazione si sarebbe ridotto il threshold a 0, semplificando fin troppo la selezione del seed set (questi nodi sarebbero stati considerati come attivi a prescindere, avendo threshold pari a 0).

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

set_edge_probability(graph, hash_table)

Descrizione:

Assegna una probabilità casuale ad ogni singolo arco del grafo.
L'ID di un edge è formato da un oggetto “Tuple” contenente (SourceNode, DestinationNode).
Per tale motivo, come key della hash table delle probabilità è stata utilizzata una stringa ottenuta dalla concatenazione di “SourceNode-DestinationNode”.

Corpo della funzione:

```
random.seed(123)
for edge in graph.Edges():
    tupla = edge.GetId()
    key = str(tupla[0]) + "-" + str(tupla[1])
    hash_table[key] = random.random()
return hash_table
```

deferred_decision(graph, probability_hash_table)

Descrizione:

Genera un grafo ridotto utilizzando la decisione differita.
Nello specifico, genera una probabilità pseudo-casuale per ogni singolo arco del grafo e la confronta con la probabilità precedentemente assegnatagli dalla funzione “set_edge_probability()", memorizzata nell'oggetto "probability_hash_table".
Se la probabilità generata per un arco è minore di quella memorizzata nella "probability_hash_table" relativa a quell’arco, allora lo rimuove dal grafo.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

Corpo della funzione:

```
_____________________________________
for edge in graph.Edges():
        number_generated = random.random()
        tupla = edge.GetId()
        key = str(tupla[0]) + "-" + str(tupla[1])
        if number_generated < probability_hash_table[key]:
                graph.DelEdge(tupla[0], tupla[1])
return graph
_____________________________________
```

Un'ulteriore scelta saliente dell'implementazione è stata la seguente:

- Necessità di ottenere una deepcopy della hash table con i thresholds per ognuna delle 10 iterazioni effettuate quando si ricorre alla decisione differita. Ciò è dovuto alla necessità di dover aggiornare i valori di threshold per ogni nodo, che andrebbe a falsare i risultati ottenuti per le iterazioni successive. Problematica ovviata tramite la costruzione di una deepcopy.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

6. Risultati

Come anticipato, l'algoritmo TSS è stato eseguito utilizzando quattro diversi approcci:

- assegnazione statica del valore dei threshold;
- assegnazione del valore dei threshold proporzionale al grado dei nodi;
- utilizzo della probabilità di attivazione in associazione all'assegnazione statica del valore dei threshold;
- utilizzo della probabilità di attivazione in associazione all'assegnazione del valore dei threshold proporzionale al grado dei nodi.

In particolare, l'assegnazione della probabilità di attivazione a ciascun arco del grafo, e la conseguente generazione di un numero casuale per ciascun arco, avviene utilizzando una funzione random con un seed fisso.
In questo modo, ogni esecuzione dell'algoritmo restituisce sempre lo stesso output, dunque i risultati saranno riproducibili.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

- **Threshold statico**

Threshold	Seed Set Size
1	1
2	1220
3	2195
4	2949
5	3549
6	4002
7	4380
8	4657
9	4928
10	5156

Tabella 2 - Risultati ottenuti dall'algoritmo TSS assegnando ai threshold un valore statico.

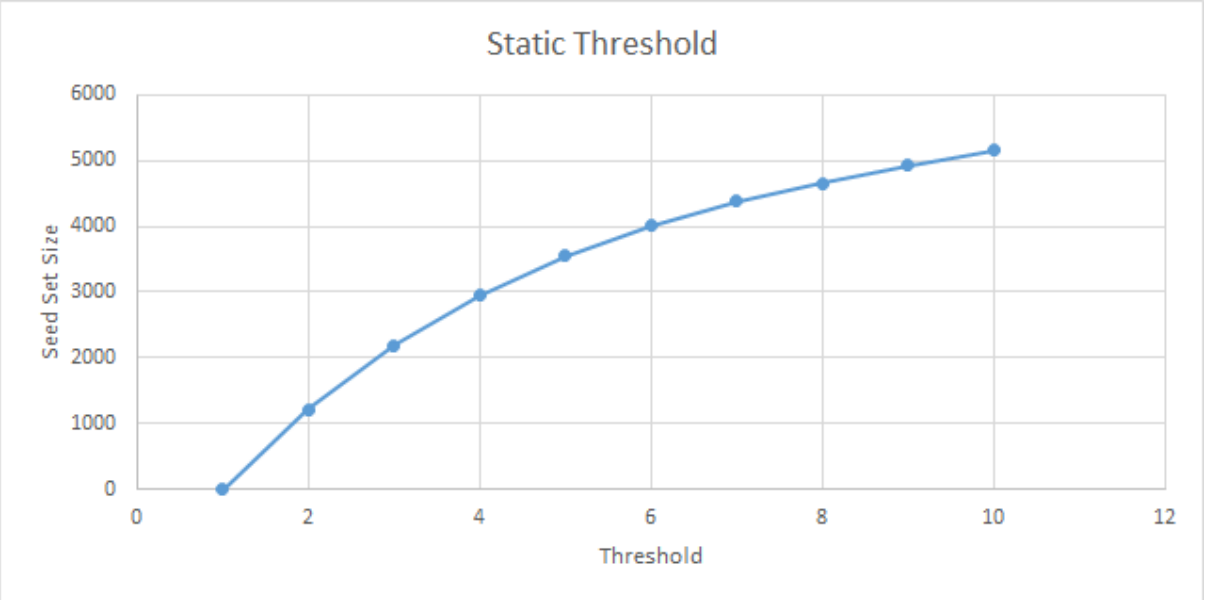

Figura 1 - Risultati ottenuti dall'algoritmo TSS assegnando ai threshold un valore statico.

Come atteso l'andamento della dimensione del seed set è crescente all'aumentare del valore del threshold statico.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

In particolare possiamo notare come con threshold pari ad 1 sia stato selezionato solo un nodo nel seed set (tendenzialmente causando un effetto a cascata per l'attivazione di tutti gli altri nodi.)
Con l'incremento del threshold la cardinalità del seed set cresce sempre più lentamente.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

- **Threshold proporzionale al grado**

Threshold Proportional to Degree	Seed Set Size
1/10 = 0.1	3
1/9 = 0.111	6
1/8 = 0.125	8
1/7 = 0.142	10
1/6 = 0.166	14
1/5 = 0.2	24
1/4 = 0.25	45
1/3 = 0.333	97
1/2 = 0.5	319

Tabella 3 - Risultati ottenuti dall'algoritmo TSS assegnando ai threshold un valore proporzionale al grado dei nodi.

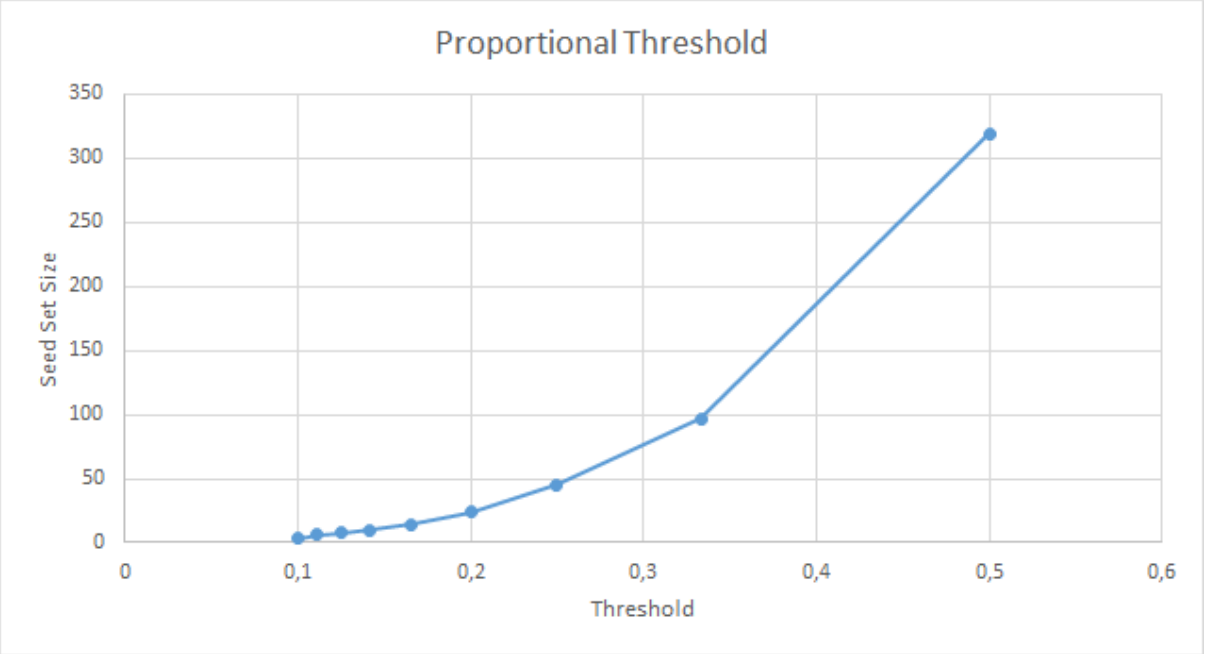

Figura 2 - Risultati ottenuti dall'algoritmo TSS assegnando ai threshold un valore proporzionale al grado dei nodi.

Considerando che nella rete la maggior parte dei nodi ha un basso numero di adiacenti, il comportamento con proporzioni basse per il threshold risulta essere particolarmente piatto e simile. Per la maggior parte dei nodi

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

della rete non vi sarà differenza nel valore del threshold di attivazione tra, ad esempio, 1/10 ed 1/8, poiché in entrambi i casi un nodo con grado 1 avrà ancora threshold pari ad 1.
Dunque una maggiore differenza nella dimensione del seed set selezionato la si può notare in corrispondenza delle proporzioni più vicine ad 1/2 (almeno 50% dei vicini attivi).

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

- **Threshold statico con probabilità di attivazione**

Threshold	Mean Seed Set Size
1	622.7
2	683.9
3	1162.6
4	1609.4
5	2019
6	2390
7	2728.1
8	3031.9
9	3302.7
10	3545.3

Tabella 4 - Risultati ottenuti dall'algoritmo TSS utilizzando la decisione differita basata sulla probabilità di attivazione, e assegnando ai threshold un valore statico.

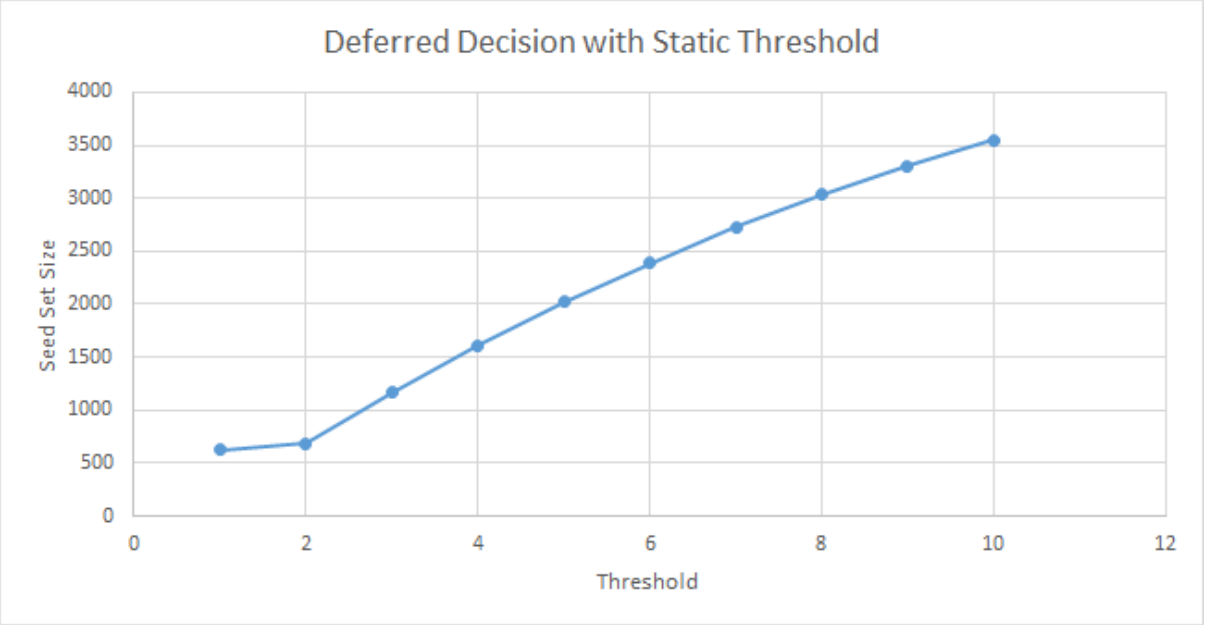

Figura 3 - Risultati ottenuti dall'algoritmo TSS utilizzando la decisione differita basata sulla probabilità di attivazione, e assegnando ai threshold un valore statico.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

In questo caso il comportamento ottenuto successivamente all'applicazione della decisione differita ha portato ad una serie di risultati generalmente migliore rispetto a quelli ottenuti in assenza di tale approccio (grafo considerato nella sua interezza).
Seppur possiamo notare che l'andamento della crescita della dimensione del seed set è coerente con quella riscontrata senza la decisione differita.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

- **Threshold proporzionale al grado con probabilità di attivazione**

Threshold	Mean Seed Set Size
1/10 = 0.1	629.1
1/9 = 0.111	631.9
1/8 = 0.125	638.2
1/7 = 0.142	647.0
1/6 = 0.166	661.9
1/5 = 0.2	699.4
1/4 = 0.25	790.5
1/3 = 0.333	1083.1
1/2 = 0.5	2270.1

Tabella 5 - Risultati ottenuti dall'algoritmo TSS utilizzando la decisione differita basata sulla probabilità di attivazione, e assegnando ai threshold un valore proporzionale al grado dei nodi.

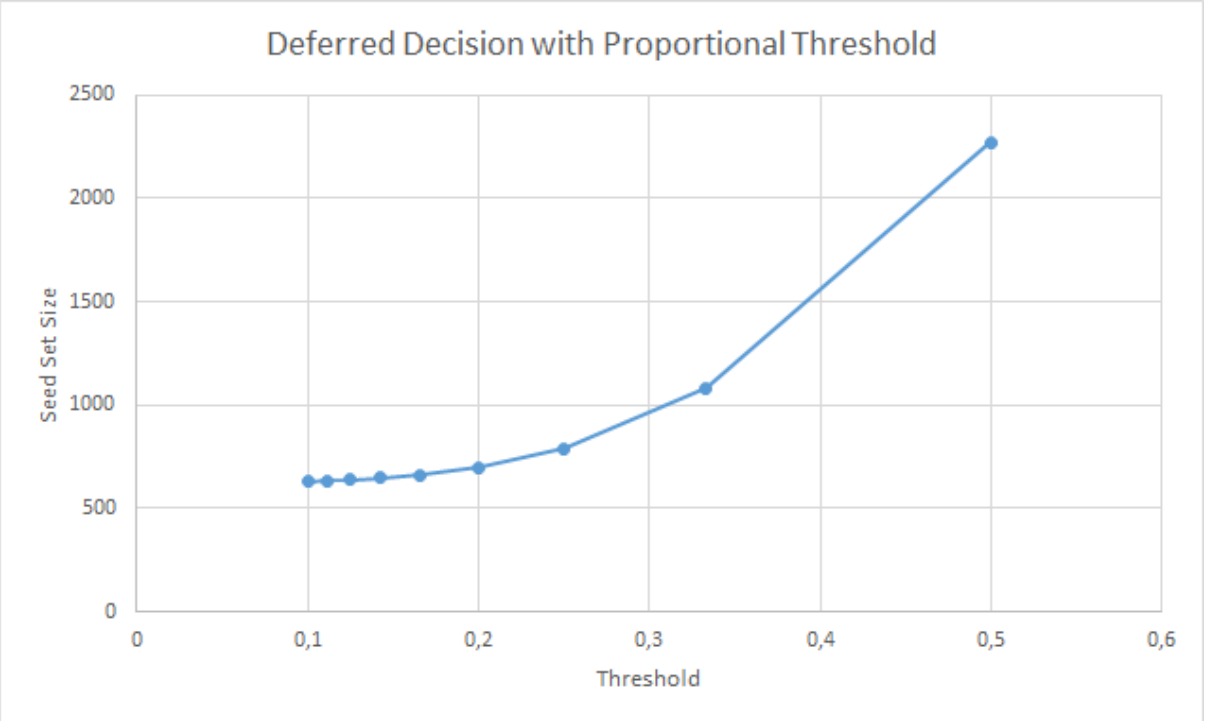

Figura 4 - Risultati ottenuti dall'algoritmo TSS utilizzando la decisione differita basata sulla probabilità di attivazione, e assegnando ai threshold un valore proporzionale al grado dei nodi.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

In quest'ultimo caso di analisi, i risultati ottenuti per la cardinalità del seed set sono generalmente peggiori rispetto a quelli ottenuti in assenza di decisione differita.
Ciò è consistente con le aspettative, poiché il threshold dei nodi è proporzionale al suo grado nel grafo originale. Nel momento in cui si va a considerare la possibilità di "non veicolazione" dell'influenza per alcuni archi (rimossi tramite la decisione differita), si ha un minore numero di vicini attivi che possono ridurre il valore del threshold necessario per attivarsi, portando di conseguenza ad un maggior numero di nodi selezionati nel seed set.
Di seguito analizziamo un semplice esempio per chiarire tale considerazione:

> Supponiamo di avere un nodo che ha grado 10 ed un threshold di attivazione proporzionale ad 1/2. Per attivarsi, tale nodo dovrebbe avere 5 dei suoi vicini attivi, oppure far parte del seed set. In seguito all'applicazione della decisione differita, solo alcuni dei suoi vicini potranno veicolare l'influenza sulla loro attivazione. Supponiamo che nella decisione differita vengano rimossi (considerati inattivi) più di 5 degli archi con i suoi vicini. Di conseguenza, nel nuovo grafo ottenuto, il suo grado sarà minore di 5, e rientrerà nel caso del TSS in cui necessariamente tale nodo deve far parte del seed set.

Nonostante ciò l'andamento di crescita della dimensione del seed set risulta essere analogo a quello riscontrato in assenza delle decisione differita.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

7. Conclusioni

In generale il lavoro svolto ha avuto l'obiettivo di implementare la soluzione per il problema del Target Set Selection come illustrata nel lavoro "*Discovering Small Target Sets in Social Networks: A Fast and Effective Algorithm*".

L'analisi effettuata sul dataset selezionato è stata limitata principalmente dalla complessità del problema e dal tempo necessario per replicare lo studio per via dell'elevata dimensione delle reti considerate.

Vi sono vari possibili sviluppi futuri per lo studio effettuato:

- Si potrebbe replicare tale analisi sui dataset di Twitch di differenti nazionalità per poi confrontare i risultati ottenuti con quelli analizzati in questo documento relativi al dataset degli utenti Twitch anglofoni.
- Si potrebbe replicare questo studio considerando altre piattaforme di game streaming diverse da Twitch (es. YouTube e Mixer) e confrontare i risultati con quelli ottenuti in questo documento.
- E' possibile effettuare un'analisi dell'implementazione effettuata ed apportare opportune modifiche per migliorare il tempo d'esecuzione generico dell'algoritmo nonché l'utilizzo di memoria.

Delli Paoli Michele - 0522500797
Porcelli Jacopo - 0522500809

Riferimenti:

1. "Target Set Selection for Conservative Populations", di Uriel Feige e Shimon Kogan - 8 Settembre 2019;
2. Libreria Snap.py per la manipolazione e analisi di reti di grandi dimensioni;
3. Dataset Twitch utilizzato - Maggio 2018;
4. Piattaforma Twitch;
5. "*Discovering Small Target Sets in Social Networks: A Fast and Effective Algorithm*", di Gennaro Cordasco, Luisa Gargano, Marco Mecchia, Adele A. Rescigno e Ugo Vaccaro - 26 Ottobre 2017;
6. Repository per l'implementazione utilizzata nello studio svolto;