

2023

Network Analisys

Indice

1 Analisi della Rete.....	2
Caratteristiche delle reti :.....	2
2 Rappresentazione grafica della Rete.....	3
3 Community Detection.....	5
3.1 Implementazione Simulated Annealing.....	7
4 Conclusioni.....	8

Liste delle figure

FIGURA 1 IA-REALITY MININGS NETWORKS.....	3
FIGURA 2 ANALISI DEI NODI ADIACENTI	4
FIGURA 4 ANALYZER NETWORK CYTOSCAPE	4
FIGURA 5 DIAMETRO MEDIO	4
FIGURA 3 COEFFICIENTE DI CLUSTERING.....	4
FIGURA 6 COMMUNITY DETECTION - METODO LOUVAIN - IGRAPH.....	5
FIGURA 7 COMMUNITY DETECTION - METODO LOUVAIN - RADATOOLS	5
FIGURA 8 COMMUNITY DETECTION - METODO INFOMAP - IGRAPH	6
FIGURA 9 MODULARITYPROC- SIMULATED ANNEALING	8

1 | Analisi della Rete

La rete oggetto dell'analisi definita "Reality Mining" è stata estratta da un progetto di studio più complesso presso il Massachusetts Institute of Technology (MIT), dove 100 utenti sono stati forniti di telefoni cellulari e sono stati monitorati i diversi sensori messi a disposizione dagli stessi telefoni. La rete nella fattispecie mostra le interazioni avvenute (chiamate o messaggi) tra i soggetti sia interni al gruppo facente parte dell'esperimento che all'esterno. Il dataset utilizzato è disponibile al seguente links

<https://networkrepository.com/ia-reality.php>

Caratteristiche delle rete :

Le caratteristiche principali della rete oggetto dell'analisi sono le seguenti :

- **Nodi:** 6.809

- **Archi:** 7.680

- **Densità:** 0,000331351. La densità di una rete misura quanto è "connessa". In questo caso, indica che la rete è relativamente scarsamente connessa, con poche connessioni rispetto al numero possibile di connessioni.

- **Grado massimo:** 261. Il grado di un nodo è il numero di archi che si connettono a quel nodo. Il grado massimo indica che c'è un nodo (il 60) con 261 connessioni, il che è significativamente più alto rispetto alla media.

- **Grado minimo:** 1. Il grado minimo indica che c'è almeno un nodo con una sola connessione, il che è il grado più basso possibile.

- **Grado medio:** 2 .Il grado medio indica la media dei gradi dei nodi nella rete, che in questo caso è 2, suggerendo una rete relativamente scarsamente connessa.

- **Assortatività:** -0,675344. L'assortatività misura la tendenza dei nodi a connettersi con altri nodi con gradi simili. Un valore negativo come questo suggerisce una tendenza verso l'assortatività negativa, cioè nodi con gradi diversi tendono a connettersi tra loro.

- **Numero di triangoli:** 1.200 . Un triangolo in una rete è una situazione in cui tre nodi sono connessi tra loro. Questo valore indica che ci sono 1.200 triangoli nella rete.

- **Numero medio di triangoli:** 0 . L'average number of triangles indica quanti triangoli, in media, sono collegati a ciascun nodo nella rete. In questo caso, sembra che i nodi abbiano pochi triangoli collegati in media.

- **Numero massimo di triangoli:** 52. Il numero massimo di triangoli collegati a un singolo nodo è 52, il che indica che alcuni nodi hanno più triangoli collegati rispetto ad altri.

- **Coefficiente di clustering medio:** 0,0177709 Il coefficiente di clustering misura la tendenza dei nodi a formare cluster o gruppi. Questo valore relativamente basso suggerisce una bassa tendenza alla formazione di cluster nella rete.

- **Frazione di triangoli chiusi:** 0,00240715. La frazione di triangoli chiusi rappresenta la proporzione di triangoli nella rete che sono "chiusi", cioè tutti e tre i nodi sono connessi tra loro. In questo caso, è una frazione molto bassa.

- **Massimo k-core:** 6 . Il massimo k-core rappresenta il k-core più grande nella rete, con k uguale a 6. Questo indica una struttura di rete con alcune porzioni altamente connesse.

- **Limite inferiore del massimo clic:** 5. Il limite inferiore del massimo clic rappresenta la dimensione minima di un clic massimo nella rete, che è 5. Un clic massimo è un sottoinsieme di nodi in cui ogni coppia di nodi è connessa tra loro.

2 | Rappresentazione grafica della Rete

Per la rappresentazione grafica della rete è stato utilizzato il software Cytoscape, evidenziando i nodi proporzionalmente al grado di connessione. Come ci si aspettava dall'analisi precedente la rete risulta alquanto eterogenea, con alcuni nodi più attivi nelle connessioni rispetto ad altri, inoltre il coefficiente di assortatività negativa già suggeriva la tendenza alla connessione di nodi con gradi diversi, pertanto otteniamo.

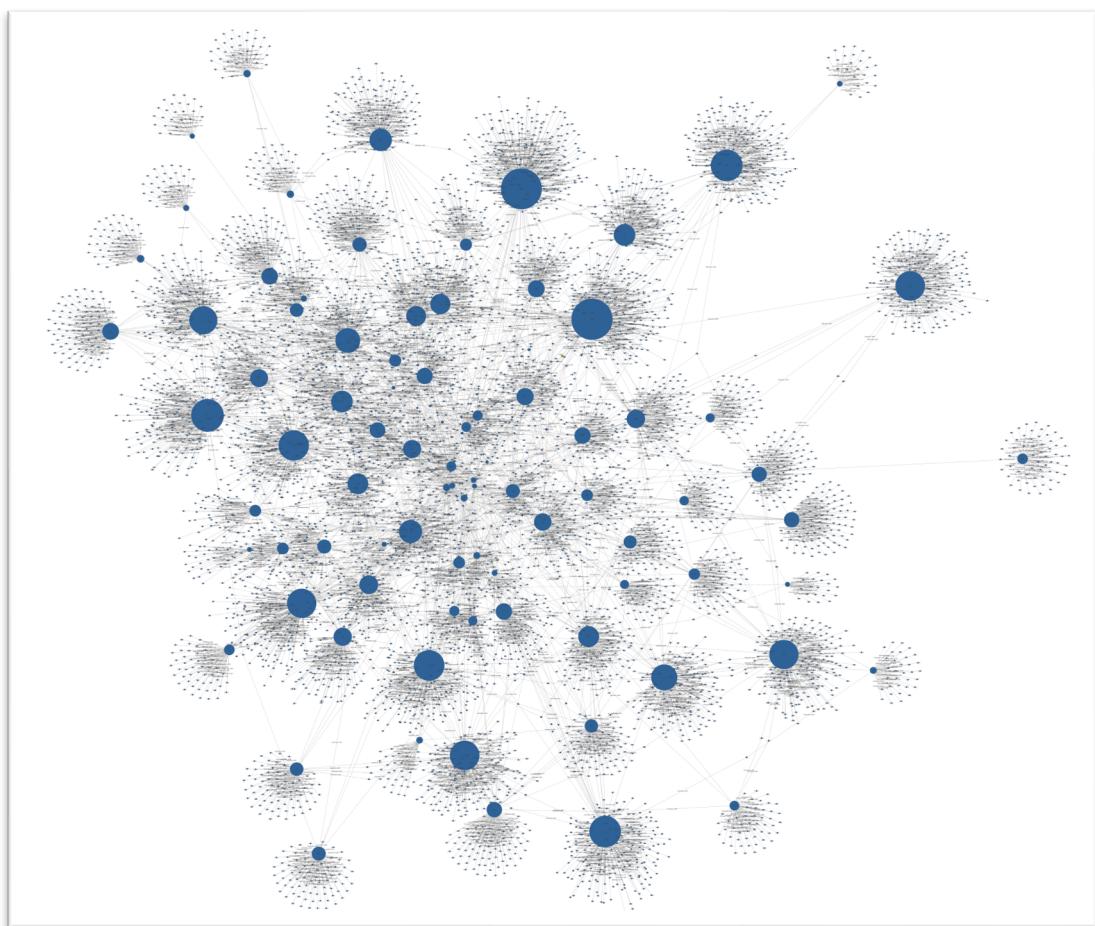


Figura 1 IA-Reality Minings Networks

Effettuando un ulteriore analisi dei nodi adiacenti (primi vicini), partendo dai nodi con il grado maggiore, si nota che alcuni nodi quali il (100) ed il (124) ed il (236) sembrano fungere da hub di connessione.

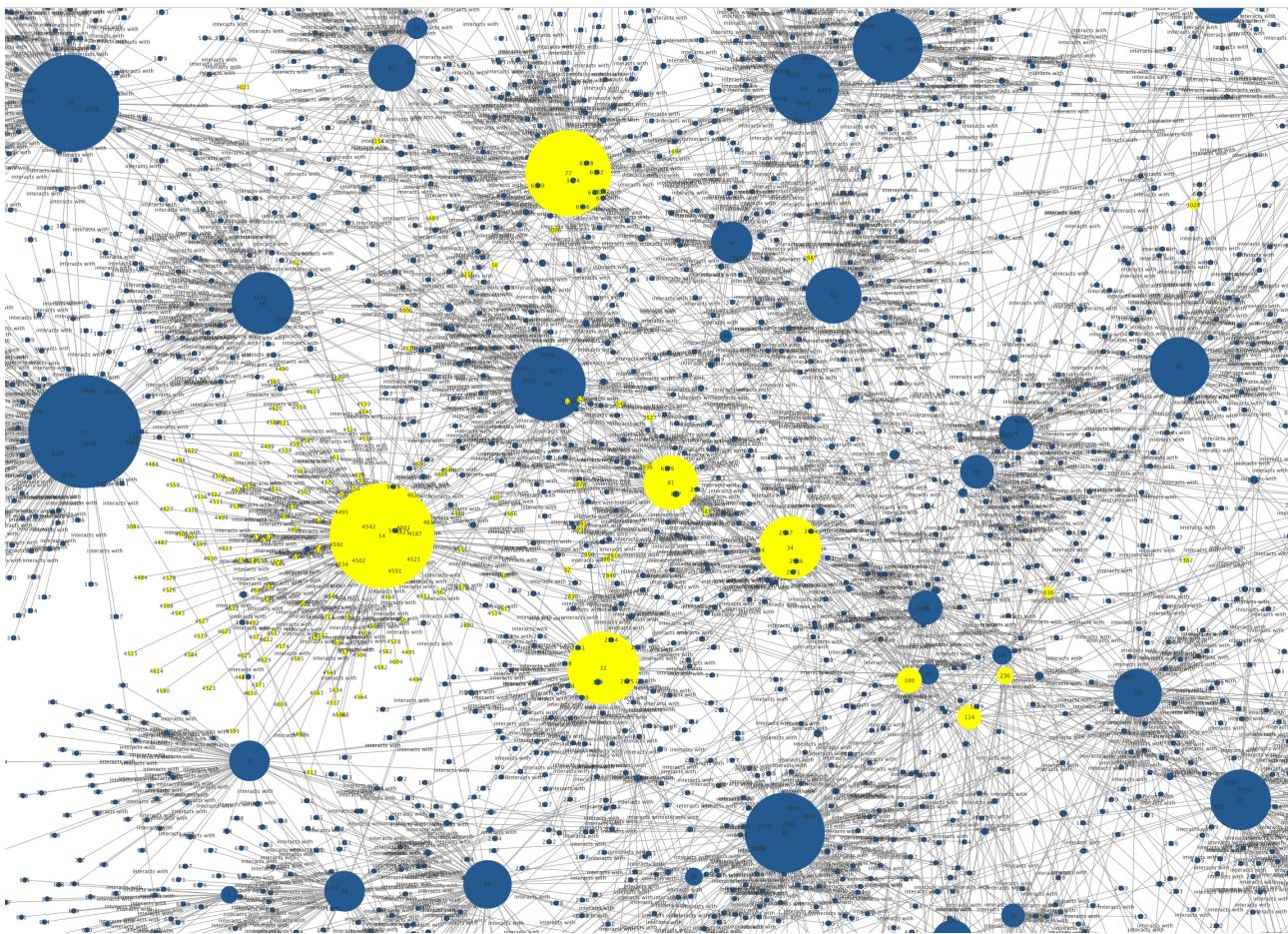


Figura 2 Analisi dei nodi Adiacenti

Si noti anche la tendenza al raggruppamento in comunità, dove i soggetti interni al campus interagiscono tendenzialmente verso l'esterno della rete; quindi, ci si aspetta un numero di cluster che sia prossimo al numero di soggetti interni alla rete.

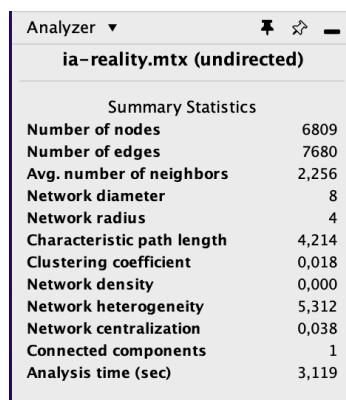


Figura 4 Analyzer Network Cytoscape

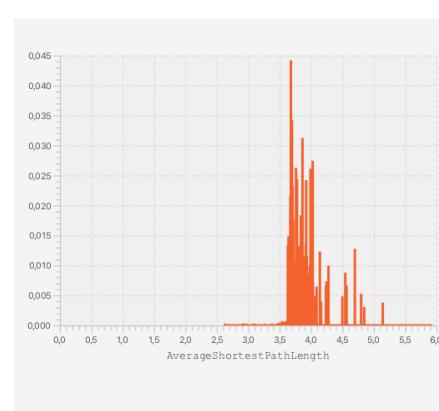


Figura 5 Diametro medio

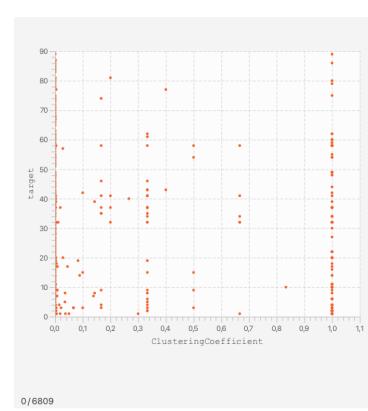


Figura 3 Coefficiente di clustering

3 | Community Detection

Dalla rappresentazione iniziale della rete e dalle indicazioni suggerite dal coefficiente di clustering ci si aspetta una tendenza a raggruppamenti in comunità esterne. Per l'analisi delle comunità risulta utile effettuare delle elaborazioni con diversi algoritmi, in quanto ciascuno esprime differenti peculiarità. Per l'elaborazione si utilizza dapprima il package *igraph* disponibile per il software RStudio. Il primo che si utilizza è l'algoritmo di Louvain, un metodo basato sulla massimizzazione della modularità, ha un approccio euristico di ottimizzazione e risulta essere computazionalmente più rapido.

```
{r echo=TRUE}
timing_result <- system.time({
  comunita_louv <- cluster_louvain(my_graph)
  num_comunita <- length(comunita_louv)
  print(num_comunita)
  lista_comunita_louv <- membership(comunita_louv)

  for (i in unique(lista_comunita_louv)) {
    nodi_comunita <- which(lista_comunita_louv == i)
    cat("Comunità", i, ":", nodi_comunita, "\n")
  }

  modularity_value_louv <- modularity(comunita_louv)
  print(modularity_value_louv)
})

print(timing_result)
```

Figura 6 Community detection - Metodo Louvain - *igraph*

Il metodo Louvain ha individuato circa 44 comunità, ed un valore di Modularità pari a 0,8730.

Valori analoghi si ottengono utilizzando la suite Radatools ed impostando una sola ripetizione del metodo.

```
/Users/agostinofontana/Desktop/radatools-5.2-mac/Communities_Detection/ia-reality.net :
Parameters: UN lf! 1
Q(l) = 0.860645 (1.140s)
Q(f) = 0.866010 (0.601s)
--- Ini_Best ---
Q_Best = 0.866010 (1.741s) communities: 44 parameters: UN lf! 1
(base) IMACDIAGOSTINO2:radatools-5.2-mac agostinofontana$ █
```

Figura 7 Community detection - Metodo Louvain - *Radatools*

Tuttavia, se aumentiamo il numero di ripetizioni ed variamo i metodi di esplorazione, otteniamo i seguenti risultati :

```

=====
Starting 100 Louvain
  Q = 0.860457  comms = 81
  Q = 0.860637  comms = 80
  Q = 0.860645  comms = 80
  Q = 0.860645  comms = 80
  resizing
  Q = 0.860645  comms = 80
  Q = 0.189739  comms = 26
  Q = 0.247171  comms = 15
  Q = 0.251518  comms = 14
  Q = 0.256934  comms = 13
  Q = 0.261771  comms = 11
  Q = 0.265511  comms = 12
  Q = 0.271290  comms = 10
  Q = 0.273257  comms = 9
  Q = 0.274727  comms = 8
  resizing
  Q = 0.757533  comms = 8
  Q = 0.051724  comms = 2
  resizing
  Q = 0.464524  comms = 2
  End rep
End of 100 Louvain
Q(l) = 0.860645 (0.651s)
=====
Starting 100 Fast
  Q = 0.864740  comms = 60
  Q = 0.866010  comms = 44
  End rep
End of 100 Fast
Q(f) = 0.866010 (0.336s)
=====
Starting 100 Exhaustive
```

Proviamo adesso ad utilizzare un secondo approccio, "Infomap" che sfrutta la teoria dell'informazione, cercando di minimizzare la funzione costo, ovvero la lunghezza della descrizione dei camminamenti casuali.

```

{r echo=TRUE}
timing_result <- system.time({
  comunita_louv <- cluster_louvain(my_graph)
  num_comunita <- length(comunita_louv)
  print(num_comunita)
  lista_comunita_louv <- membership(comunita_louv)

  for (i in unique(lista_comunita_louv)) {
    nodi_comunita <- which(lista_comunita_louv == i)
    cat("Comunità", i, ":", nodi_comunita, "\n")
  }

  modularity_value_louv <- modularity(comunita_louv)
  print(modularity_value_louv)
})

print(timing_result)
```

Figura 8 Community detection - Metodo Infomap - igraph

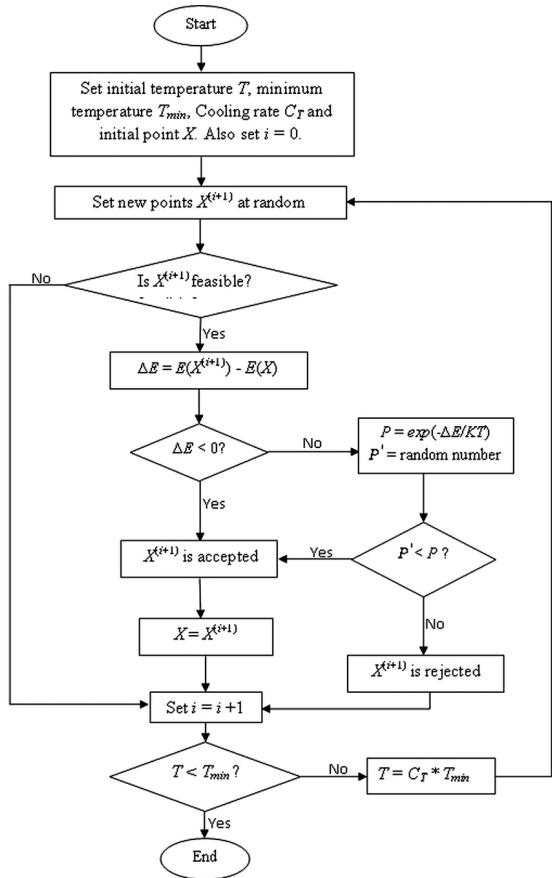
il metodo Infomap individua circa 90 comunità con un coefficiente di modularità pari a 0.86 circa, tuttavia i tempi di esecuzione sono di 10 volte superiori.

3.1 | Implementazione Simulated Annealing

Proviamo adesso ad applicare il metodo del *Simulated Annealing* effettuando una partizione in due comunità, e ricercando la soluzione che massimizza la modularità.

Per l'applicazione dell'algoritmo del *Simulated Annealing* i passi fondamentali sono i seguenti:

1. **Inizializzazione** : definizione della temperatura iniziale, temperatura finale, costante di decremento della temperatura.
2. **Calcolo della funzione obiettivo** : "modularità"
3. **Ciclo principale** :
 1. Generazione di una nuova soluzione effettuando una perturbazione dalla soluzione di partenza
 2. Calcolo della funzione obiettivo
 3. Decisione:
 1. Se la nuova configurazione ha una modularità maggiore, accettata;
 2. Se la modularità è inferiore, si accetta la nuova configurazione con una probabilità che dipende dalla differenza di modularità e dalla temperatura corrente. Questo permette di esplorare lo spazio di soluzioni evitando i minimi locali;
 3. Aggiornamento della temperatura;
4. **Reiterazione del processo**;
5. **Soluzione finale**.



Per l'implementazione del codice utilizzato si rimanda al file notebook allegato "simulated annealing in reality.rmd"

Il metodo del simulated annealing, risulta particolarmente efficace nell'esplorazione dello spazio delle soluzioni, in quanto il metodo accetta con una certa probabilità decrescente, delle soluzioni subottimali con valori peggiori dei precedenti. Nel caso della rete oggetto di studio la modularità definita dalla soluzione euristica di partenza è pari a -0,004514982 ovvero prossimo allo 0, ciò significa che la rete è essenzialmente casuale. Stabiliti i parametri fondamentali per l'applicazione dell'algoritmo, ovvero : funzione obiettivo - T0 temperatura iniziale, Tf temperatura finale, Nsim numero di simulazioni; è possibile iterare il processo.

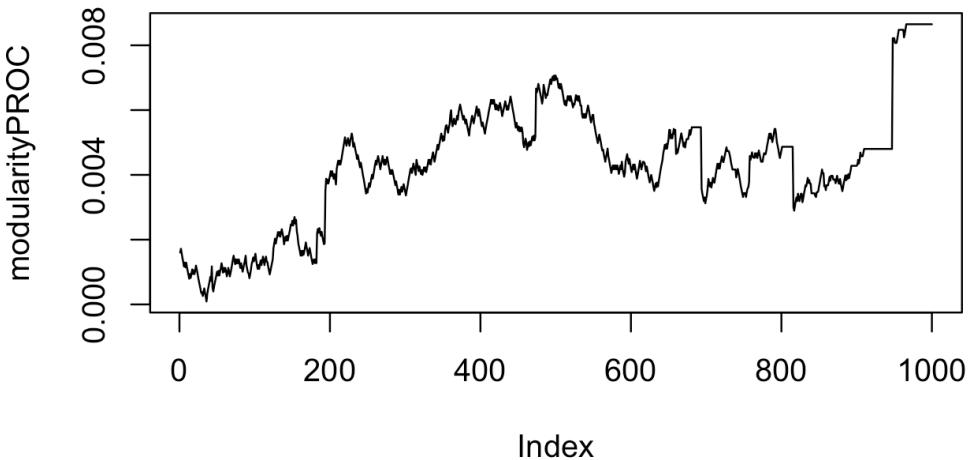


Figura 9 ModularityPROC- Simulated Annealing

La rappresentazione grafica mostra l'andamento della modularità al variare del numero di iterazioni, si può osservare un significativo miglioramento dopo circa 900 iterazioni, ciò significa che l'algoritmo sta convergendo verso una soluzione ottimale che massimizza la modularità; che in questo caso vale 0,008650682.

4| Conclusioni

In definitiva, analizzata la rete ed osservati i risultati ottenuti con i diversi algoritmi di ottimizzazione, è necessario effettuare delle osservazioni; I metodi utilizzati per l'individuazione delle comunità hanno espresso caratteristiche differenti, in quanto differenti sono gli approcci utilizzati nella ricerca della funzione obiettivo.

L'algoritmo del Simulated Annealing essendo un metodo di ottimizzazione della funzione costo, che in questo caso è pari alla modularità, ovvero, ottenuta la suddivisione in due comunità, ne ricerca la combinazione ottimale tra lo spazio delle soluzioni, risulta essere computazionalmente più complesso e con tempi di elaborazione decisamente più lunghi, legati all'esplorazione dello spazio delle soluzioni.

L'algoritmo di Louvain invece risulta essere molto più rapido nell'individuare il numero ottimale di cluster che massimizzano la modularità, essendo specializzato nel rilevamento delle comunità.

L'altro algoritmo utilizzato è stato il metodo Infomap, che si basa sul flusso delle informazioni, nella rete in esame ha individuato circa 90 comunità, un valore prossimo al numero di soggetti interni alla rete.

In generale quindi, potrebbe essere opportuno utilizzare l'algoritmo del Simulated Annealing partendo da una soluzione subottimale, ottenuta con gli altri due metodi analizzati.