|  |
| --- |
| [Digitare il nome della società] |
| Dynamic Fanout & Advertising Limit |
| Come vengono calcolati |

|  |
| --- |
| Lorenzo Pagliari  25/05/2015 |

Sommario

[1. Introduzione 3](#_Toc420918621)

[2. Dynamic Fanout 4](#_Toc420918622)

[2.1. Versione 1 5](#_Toc420918623)

[2.1.1. Battery Factor 5](#_Toc420918624)

[2.1.2. Number of Nodes Factor 6](#_Toc420918625)

[2.2. Trends Graph 7](#_Toc420918626)

[2.3. Versione 2 9](#_Toc420918627)

[2.3.1. Battory Factor 9](#_Toc420918628)

[2.3.2. Number of Nodes Factor 10](#_Toc420918629)

[2.3.3. Trends Graph 12](#_Toc420918630)

[3. Advertising Limit 14](#_Toc420918631)

[3.1. Battery Factor 14](#_Toc420918632)

[3.2. Number of Nodes Factor 14](#_Toc420918633)

[3.3. Trends Graph 15](#_Toc420918634)

# Introduzione

In questo documento cercherò di illustrare quali sono i principali due parametri usati in questo progetto e in che modo essi vengono calcolati.

L’ambiente a cui si farà riferimento sarà sempre un ambente con un numero di nodi maggiore di zero, con la possibilità di mobilità dei nodi. Infatti i parametri verranno calcolati dinamicamente ad intervalli regolari in modo da adeguarsi al cambiamento dell’ambiente circostante e alle sue variazioni.

I due principali parametri che discuterò sono:

* Dynamic Fanout
* Advertising Limit

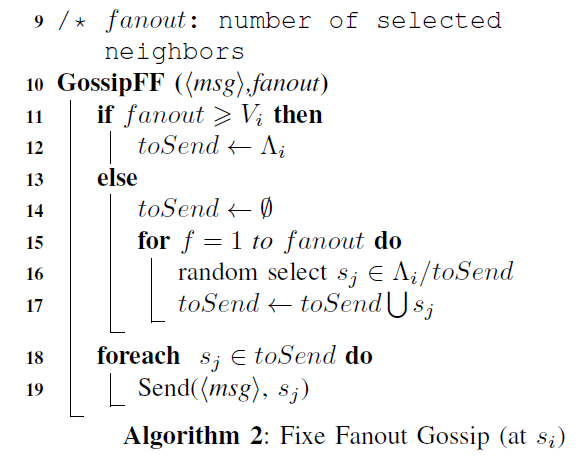
Lo scopo del progetto è di creare un sistema di comunicazione alternativo, su piattaforma mobile, da utilizzare in situazioni di emergenza quando le principali linee di comunicazione non sono disponibili. Quindi il principale obiettivo è rendere possibile e disponibile la comunicazione tenendo conto del consumo energetico del dispositivo mobile che si sta utilizzando, in modo da poter garantire per il maggior tempo possibile la possibilità di ricevere comunicazioni. Dato che il progetto utilizza la tecnologia Bluetooth per diffondere informazioni, l’altro punto chiave che potrebbe influire sul consumo energetico è la quantità di “nodi” presenti in una determinata area e quindi il numero di nodi che ogni nodo è in grado di “sentire”. Un protocollo di broadcasting puro tenderebbe a sovraccaricare ogni nodo di lavoro inutile facendogli sprecare batteria e riducendone così la durata. Da questo è nata l’esigenza di avere dei parametri che possano rappresentare una sorta di limite alla trasmissione, quindi al consumo di energia, in base alle condizioni dell’ambiente circostante ed essere in grado di variare dinamicamente in modo da seguire i cambiamenti dell’ambiente. Inoltre i parametri devono tenere conto anche dell’energia del dispositivo stesso, meno energia esso ha, meno carico di lavoro bisogna dargli in modo da permettergli una maggiore operatività globale.

Quindi i due fattori che servono per regolare i parametri sono:

* Livello della batteria del dispositivo
* Numero di nodi che il dispositivo riesce a percepire nel suo raggio d’azione.

# Dynamic Fanout

L’idea base che sta ci sta sotto è quella dell’algoritmo di Gossip del **Fixed Fanout**, descritto dall’algoritmo seguente (fig.2.1):



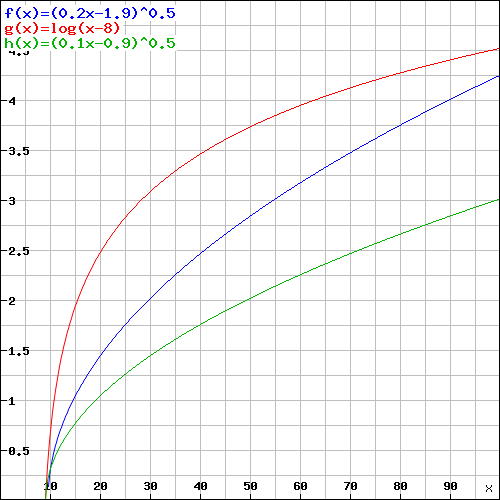
*Figura 2‑1: Fixed Fanout Algorithm*

Ho tenuto la stessa idea di Fanout come metodo di diffusione del messaggio, inserendo sistemi di “tuning” automatici in modo da introdurre una dipendenza dal livello della batteria e dal numero di nodi raggiungibili. Ulteriori piccole modifiche sono state fatte per far aderire questo algoritmo al sistema di trasmissione utilizzato.

## Versione 1

### Battery Factor

Questo fattore del Dynamic Fanout tiene conto del livello di batteria del dispositivo. Si è cercato di studiare quale fosse stato un comportamento corretto al variare della percentuale di carica e quindi si è pensato di adattare il comportamento di funzioni positive monotone crescenti, nel primo quadrante, nell’intervallo 0-100. Di seguito in fig. 2.2, è riportato un grafico con le tre funzioni sono sembrate adatte.



*Figura 2‑2: Battery factor graph*

La funzione scelta è stata la seguente:

L’ipotesi è che il dispositivo mobile fornisca al sistema il valore della batteria come un numero intero compreso tra 1 e 100; proprio come si usa vedere nei cellulari, la percentuale di batteria visualizzata a schermo varia tra 1 e 100. La radice viene divisa per 10 per ottenere il corretto valore percentuale compreso tra 0 e 1.

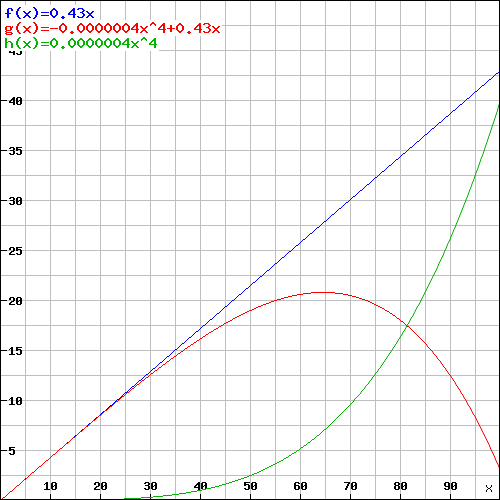
Quello che risulta da questo fattore, è una percentuale e indica la percentuale di contanti che verrà scelto come Dynamic Fanout, prima della correzione.

### Number of Nodes Factor

Questo fattore del Dynamic Fanout tiene conto del numero di altri dispositivi mobile che il nodo riesce a sentire nel suo raggio d’azione. L’idea è di avere un andamento linearmente crescente proporzionale al numero di utenti. Per evitare di avere un enorme fanout per numero di utenti molto elevato, il che potrebbe comunque rivelarsi inutile e irraggiungibile, viene inserito un fattore di smorzamento per i valori di utenti molto elevati.

***“…idealmente ci dovrebbe essere un comportamento asintotico per il numero di nodi che tende a ∞, ma per ora è stato progettato il comportamento fino ad un max di 100 utenti che è stata valutata una situazione limite e difficilmente raggiungibile…”***

In figura 2.3 è rappresentato un esempio dell’andamento di questo fattore per il 100% di batteria con l’aggiunta del fattore di correzione.



*Figura 2‑3: Number of Nodes Factor*

In blu l’andamento dato dal livello di batteria, in verde il fattore di correzione e in rosso l’andamento totale risultante.

Fattore correttivo:

Dynamic Fanout risultante:

## Trends Graph

Nella figura seguente vengono rappresentati su grafico, gli andamenti del Battery Factor ogni 10% di batteria. Viene inoltre rappresentato la curva del fattore di smorzamento.

Mentre nel grafico seguente vi è il risultato finale dell’andamento del Dynamic Fanout, risulante dalla differenza delle curve di cui al grafico precedente. E’ stata inserita forzatamente l’asintoticità per quantità di nodi superiori a 60 e il cui DF risulti minore del 5% dei nodi percepiti. E’ stata solo una convezione per garantire un DF positivo su tutto il grafico.

Il motivo principale dell’aggiunta del fattore asintotico è quella di non avere mai un DF negativo e stabilire un minimo % per ogni curva di batteria che il sistema può garantire, così da poter ovviare a DF negativi dati dalla formula per valori molto alti di nodi.

L’asintoto viene calcolato così:

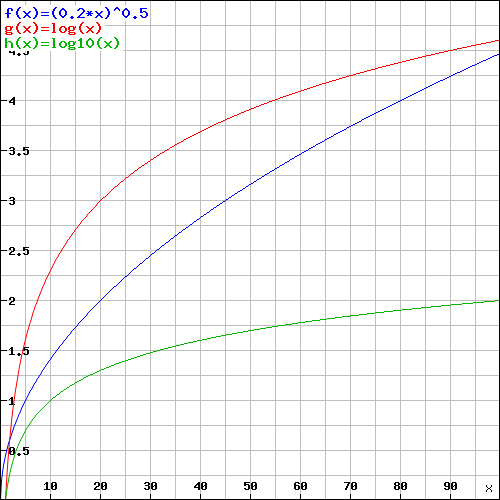
Asintoto:

## Versione 2

Versione 2 del calcolo del Dynamic Fanout. La versione 1 è molto più conservativa, mentre questa versione è più abbondante per valori piccoli di numero nodi, ma ha anche un fattore di correzione più efficace anch’esso a partire da valori piccoli di numero nodi. Questo risulta in una riduzione delle curve e dei rispettivi massimi.

### Battory Factor

E’ stata mantenuta la stessa funzione, ma è stato tolta la traslazione sull’asse delle x.

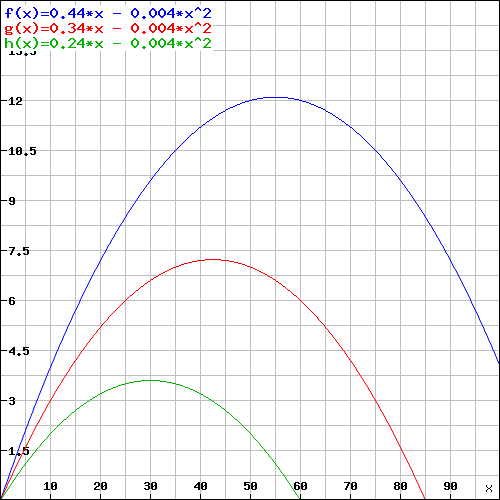


*Figura 2‑4: Battery graph*

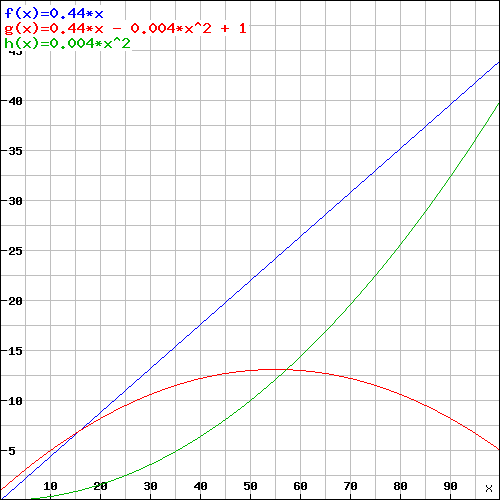
La funzione scelta è stata la seguente:

### Number of Nodes Factor

In figura 2-5, vengono rappresentate 3 curve del DF al variare del numero di nodi. In questo grafico viene solo rappresentato il comportamento, col fattore di correzione incluso, ma senza la componente asintotica. Le curve rappresentate sono quelle per il 100%, 60% e 30% di batteria.



*Figura 2‑5: Dynamic Fanout - number of nodes*



*Figura 2‑6: single funzioni e fattore correttiva*

In blu viene rappresentato il fattore per batteria 100%, in verde il fattore correttiva e in rosso la funzione risultante. Successivamente, nel paragrafo *Trends Graph* vengono rappresentati tutti gli andamenti per tutte le curve di batteria.

Fattore correttivo:

Dynamic Fanout risultante:

### Trends Graph

Nella figura seguente vengono rappresentati su grafico, gli andamenti del Battery Factor ogni 10% di batteria. Viene inoltre rappresentato la curva del fattore di correzione.

Mentre nel grafico successivo vi è il risultato finale dell’andamento del Dynamic Fanout, risulante dalla differenza delle curve di cui al grafico precedente. E’ stata inserita forzatamente l’asintoticità per quantità di nodi superiori a 60 e il cui DF risulti minore del 5% dei nodi percepiti. E’ stata solo una convezione per garantire un DF positivo su tutto il grafico.

Il motivo principale dell’aggiunta del fattore asintotico è quella di non avere mai un DF negativo e stabilire un minimo % per ogni curva di batteria che il sistema può garantire, così da poter ovviare a DF negativi dati dalla formula per valori molto alti di nodi.

Il fattore asintotico viene inserito per valori alti di numero nodi ed è fissato a:

Asintoto:

Inserendo il Battery Factor nel calcolo dell’asintoto, fa si che quasi tutte le curve abbiamo il loro asintoto proporzionale al livello di batteria.

# Advertising Limit

L’advertising Limit è un vincolo di stop spamming in fase di advertising atto a ridurre comunque gli sprechi inutili di energia dovuti a trasmissioni a loro volta inutili.

Un dispositivo con un messaggio da diffondere inizia a fare advertising, e dopo ogni trasmissione conclusa con successo ricomincia a fare advertising. Il problema è che questa azione di per sé non ha un limite e il dispositivo continuerà a “spammare” all’infinito sprecando energia, finché altri fattori gli impediranno di farlo. Per questo motivo ho cercato di dare una logica sulla scelta di ricominciare a diffondere un messaggio oppure no. La logica che vi sta sotto, come per il DF, si basa sia sul livello di batteria sia sul numero di nodi percepiti. Questo perché se il dispositivo ha poca batteria dovrà anche cercare di preservarne il più possibile, mentre se ne ha abbastanza può permettersi di fare qualche tentativo in più. Allo stesso modo, più nodi un nodo percepisce, più sarà utile alzare questo limite per agevolare la diffusione dell’informazione. Questo fattore viene usato come limite al numero di tentativi di advertising consecutivi andati a vuoto, cioè che non hanno avuto risposta. Questo rappresenta il fatto che se tutti i nodi che sento, e che quindi ricevono il mio tentativo di advertising, non mi rispondono, vuol dire che sono disinteressati all’informazione oppure che l’hanno già ricevuta, e quindi non ha più senso che il dispositivo continui nell’attività di advertising sprecando energia.

## Versione 1

Nella versione 1 dell’Advertising Limit è presente una componente dipendente dal livello di batteria e un fattore dipendente dal numero di nodi. Questa versione risulta essere molto conservativa a livello energetico, ma potrebbe causare grosse perdite in termini di efficienza per quanto riguarda lo spreading di informazione.

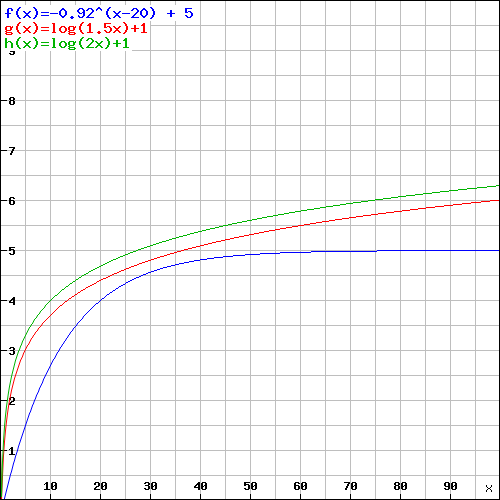
### Battery Factor

Per l’Advertising Limit il Battery Factor è semplicemente il livello di batteria, messo in percentuale e servirà come riduzione sulla curva relativa al numero di utenti.

Battery Factor:

### Number of Nodes Factor

Per il fattore che tiene in considerazione in numero di dispositivi percepiti, ho utilizzato una funzione monotona crescente, che ha una rapida crescita all’inizio e molto più lenta al crescere del numero di dispositivi. Nella figura seguente si vedono i tre andamenti delle tre funzioni più adatte che ho trovato.



*Figura 3‑1: Numbre of Nodes functions*

La funzione che ho scelto è stata la seguente.

Users Factor:

Quindi alla fine la formula per l’AL risulta essere.

Advertising Limit:

### Trends Graph

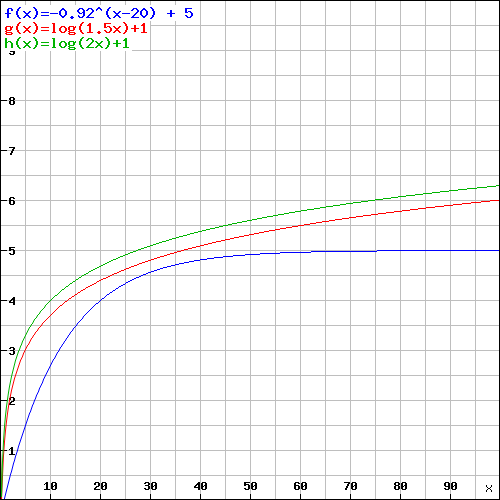
Nel grafico seguente vengono riportate le 10 curve per i 10 stadi di batteria. Le curve per il 100% e il 90% risultano essere sovrapposte e ciò non crea problemi in quanto per quei livelli di batteria ha senso non fare differenze sui limiti. Le ottimizzazioni devono farsi sentire più si va verso livelli più bassi di energia.

## Versione 2

Nella versione 2 dell’Advertising Limit non è presente una componente dipendente dal livello di batteria ma solo un fattore dipendente dal numero di nodi. Questa versione risulta essere meno conservativa a livello energetico, ma alza notevolmente il limite di advertising, permettendo e a volte garantendo, ottimi miglioramenti in termini di spreading dell’informazione.

### Number of Nodes Factor

Per il fattore che tiene in considerazione in numero di dispositivi percepiti, ho utilizzato una funzione monotona crescente, che ha una rapida crescita all’inizio e molto più lenta al crescere del numero di dispositivi. Nella figura seguente si vedono i tre andamenti delle tre funzioni più adatte che ho trovato.



*Figura 3‑2: Numbre of Nodes functions*

La funzione che ho scelto è stata la seguente.

Users Factor:

Quindi alla fine la formula per l’AL risulta essere.

Advertising Limit:

## Versione 3

Nella versione tre si cerca di trovare un compromesso tra le prime due versioni. Ovvero si cerca di ovviare a problemi della troppa costrizione della prima versione e alla contrapposta, troppo permissiva della versione due. Nella versione tre non avremo comunque un battery factor, ma cercheremo di modellizzare la funzione che ne descrive il comportamento, forzando una crescita molto lenta per quantità elevate.

Come nella versione due, la funzione base è:

NodeFactor:

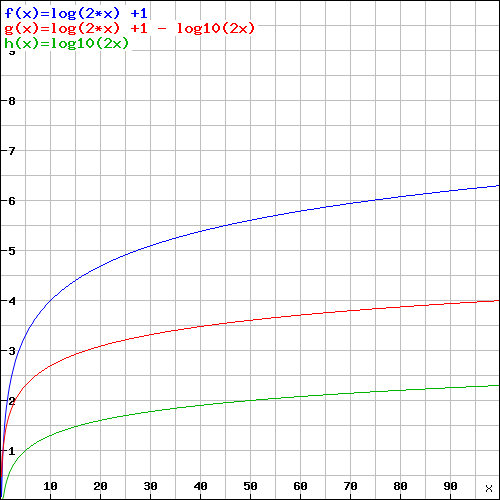
Ora applichiamo ad essa un fattore limitante

Fattore limitante:

L’Advertising Limit risulta quindi essere

Advertising Limit:

Nella figura seguente, vengono riportate le tre funzioni. In blu il Node Factor, in verde il fattore limitante e in rosso l’Advertising Limit.



*Figura 3‑3: Advertising Limit -v3*