ComPAS'2013 : RenPar'21/ SympA'15/ CFSE'9
Grenoble, France, du 16 au 18 janvier 2013

Comparaisons équitables des algorithmes de gossip sur les topologies aléatoires à grande-échelle

Ruijing Hu, Julien Sopena, Luciana Arantes, Pierre Sens, Isabelle Demeure

Université Pierre et Marie Curie (Paris 6)
4 place Jussieu
75252, Paris Cedex 5 France
prénom.nom@lip6.fr

Résumé
Cet article compare les performances des trois grandes familles de protocoles probabilistes de dissémination d'informations (gossip) exécutées sur trois graphes aléatoires. Les trois graphes représentent les topologies typiques des réseaux à grande-échelle : le graphe de Bernoulli (ou Erdős-Rényi), le graphe géométrique aléatoire et le graphe scale-free. Nous proposons un nouveau paramètre générique : le *fanout effectif*. Pour une topologie et un algorithme donnés, le fanout effectif caractérise la puissance moyenne de la dissémination des sites. Il permet l'analyse précise du comportement d'un algorithme sur une topologie. De plus, il simplifie la comparaison théorique des différents algorithmes sur une topologie. En s'appuyant sur les résultats obtenus par les expérimentations dans le simulateur OMNET++, qui exploitent le fanout effectif, nous étudions l'impact des topologies et les algorithmes sur les performances. Nous suggérons également une façon de les combiner afin d'obtenir le meilleur gain en termes de fiabilité.

Mots-clés : algorithmes de gossip, topologies aléatoires, évaluation de performance, fiabilité, complexité en messages

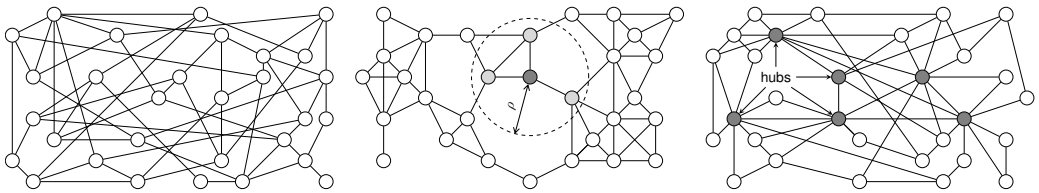
---

1. Introduction

La dissémination d'informations (broadcast), où un site tente de diffuser des messages à tous les autres sites dans le réseau, est essentielle pour beaucoup d'applications réparties.
Les *protocoles inondation* (*flooding*) sont une solution simple mais inefficace pour disséminer des informations. Lors de la première réception d'un nouveau message, tous les sites le retransmettent à tous leurs voisins [13]. Pour réduire le nombre de messages, les algorithmes probabilistes de gossip sont une solution efficace et fiable. Ils sont ainsi couramment utilisés dans les réseaux logiques (overlay) [6, 8, 14], les réseaux sans fil ad-hoc et les réseaux de capteurs [3, 10, 12, 22]. Il existe trois grandes classes d'algorithmes de gossip : le gossip avec fanout fixé (*GossipFF*) [6, 8, 14], le gossip avec choix d'arêtes probabiliste (*GossipPE*) [10, 22] et le gossip basé sur une diffusion probabiliste (*GossipPB*) [3, 12, 23].
Ces algorithmes peuvent s'exécuter sur des topologies de réseaux ayant des propriétés très différentes en termes de distribution des degrés, de la dépendance des arêtes,etc. qui ont un impact important sur les performances. Notre article considère les topologies suivantes : le graphe de Bernoulli (ou Erdős-Rényi) ($\mathcal{B}(N, p_N)$), le graphe géométrique aléatoire ($\mathcal{G}(N, \rho)$) et le graphe

scale-free ($\mathcal{S}(N, m)$). Ces trois familles des graphes modélisent respectivement un système pair à pair [14], un réseau de capteurs [12] et Internet [16].
Cet article compare les performances des trois algorithmes de gossip sur ces trois topologies aléatoires. Pour que la comparaison soit équitable, nous avons introduit un nouveau paramètre : le *fanout effectif* qui caractérise la puissance moyenne de dissémination des sites infectés dans le système, c'est-à-dire ceux qui reçoivent le message au moins une fois. Pour une topologie donnée, le fanout effectif peut être calculé en fonction du paramètre d'entrée de l'algorithme correspondant. Ce nouveau paramètre simplifie l'étude théorique des différents algorithmes sur une topologie.
En exploitant le fanout effectif, nous présentons dans cet article les résultats d'évaluation de performance sur le simulateur OMNET++ [1], qui comparent *GossipFF*, *GossipPE* et *GossipPB* sur les trois topologies. À notre connaissance, une telle comparaison comparaisons n'a jamais été faite.
Les sections 2, 3 et 4 décrivent respectivement les familles de graphe, les trois algorithmes de gossip et les métriques de performance. La section 5 présente un état de l'art. Le fanout effectif est décrit dans la section 6. La section 7 présente la comparaison des trois algorithmes.

2. Topologies du système



(a) Graphe de Bernoulli (b) Graphe géométrique aléatoire (c) Graphe scale-free

FIGURE 1 – Exemples des topologies aléatoires avec 30 sites et de degré moyen=4

Dans la suite, $|\, l \,|$ dénote le cardinal de l'ensemble l.
On considère que le système de dissémination Π comprend N sites $\{s_1, s_2, \cdots, s_N\}$. L'ensemble des voisins de s_i est noté Λ_i et $V_i = |\, \Lambda_i \,|$ indique le degré de s_i. $P(k)$ représente la distribution des degrés d'un site ayant k voisins (c'est-à-dire, la fraction de sites avec un degré k) dans le graphe et $\bar{V}$ est le degré moyen du graphe. Donc, $\bar{V} = \sum_{k=0}^{N-1} P(k) \cdot k$. Les sites ne bougent pas, ni tombent en panne dans nos études actuelles. En plus, aucune perte de message n'est prise en compte. Tous ces points seront adressés au travail du futur.
Nous étudions trois topologies aléatoires : le graphe de Bernoulli (ou Erdős-Rényi) $\mathcal{B}(N, p_N)$ [5], le graphe géométrique aléatoire $\mathcal{G}(N, \rho)$ [18] et le graphe scale-free $\mathcal{S}(N, m)$ [2] (voir respectivement les figures 1(a), 1(b) et 1(c)).
Le graphe de Bernoulli (ou Erdős-Rényi) $\mathcal{B}(N, p_N)$ est un graphe aléatoire bidirectionnel, construit en créant indépendamment une arête entre deux sites du système avec une probabilité p_N. On admet que $p_N > \frac{(1+\varepsilon)\cdot \ln(N)}{N}$ avec une constante positive ε. Ceci permet la création d'une *composante géant* de N sites dont la distribution des degrés suit la loi de Poisson $P(k) = \exp\left(-\bar{V}\right) \frac{\bar{V}^k}{k!}$, où $\bar{V} = p_N \cdot N$.

Le graphe géométrique aléatoire $\mathcal{G}(N, \rho)$ est un graphe aléatoire bidirectionnel dans une région bornée. Dans nos études c'est une région rectangulaire de longueur a et de largeur b. $\mathcal{G}(N, \rho)$ est généré en plaçant les sites uniformément, aléatoirement et indépendamment dans la région. En outre, deux sites sont connectés, si et seulement si la distance entre eux est inférieure à ρ. Basé sur [19], on définit $\rho > \sqrt{\frac{(1+\varepsilon)\cdot\ln(N)\cdot a\cdot b}{N\cdot\pi}}$ avec une constante positive ε afin d'assurer que le graphe soit connecté avec une distribution des degrés suivant la loi de Poisson $P(k) = \exp\left(-\bar{V}\right)\frac{\bar{V}^k}{k!}$ où $\bar{V} = \frac{N\cdot\pi\cdot\rho^2}{a\cdot b} - 1$, si l'on ignore l'effet de la borne à la région dans $\mathcal{G}(N, \rho)$.
Le graphe scale-free est un graphe aléatoire bidirectionnel dont la distribution des degrés suit une loi de puissance. Un graphe scale-free $\mathcal{S}(N, m)$ peut être généré par le modèle Barabási-Albert [2]. La génération du réseau commence à partir d'une clique de m_0 sites ($m_0 \ll N$). Puis chaque nouveau site crée $m(\leqslant m_0)$ arêtes connectées à m différents sites déjà présents dans le graphe. La probabilité p qu'un nouveau site soit connecté à un site existant est proportionnelle au degré de ce dernier. Ceci est appelé *l'attachement préférentiel*. Ce processus assure que le graphe est connecté avec une distribution des degrés suivant la loi de puissance, qui est approximée par $P(k) = \frac{2m(m+1)}{k(k+1)(k+2)}$, où $k = m, m+1, \cdots, N-1$ et $\bar{V} = 2m$ [20]. Dans ce graphe, il y a des **hubs** et **sites périphériques** dont les degrés respectifs sont supérieurs à $2m$ et entre m et $2m$. On peut déduire $|\Pi_p| > 3\,|\Pi_h|$, où Π_h est l'ensemble de hubs et Π_p est l'ensemble de sites périphériques.
La dépendance des arêtes (ou le coefficient de clustering) d'un graphe aléatoire donné, pour trois différents sites s_i,s_j,s_k, est définie par la probabilité conditionnelle sachant l'existence des arêtes $s_i \sim s_k$ et $s_j \sim s_k$, qu'une arête $s_i \sim s_j$ existe (c'est-à-dire, $P(s_i \sim s_j | s_i \sim s_k, s_j \sim s_k)$).

3. Algorithmes de gossip

La dissémination d'informations dans un réseau à grande échelle s'appuie sur l'algorithme de gossip générique illustré par l'algorithme 1. Initialement, la source envoie son message à *tous* ses voisins (lignes 2 et 3). Un site retransmet le message reçu en appelant la procédure Gossip() (ligne 8) à condition qu'il ne l'ait jamais reçu. Sinon, le message est abandonné. Les sites qui reçoivent le message au moins une fois sont appelés **sites infectés**.

Algorithme 1 : Algorithme de gossip générique

```
1 Broadcast (⟨msg⟩)
2     foreach s_j ∈ Λ_i do
3         Envoyer (⟨msg⟩, s_j) ;
4 Recevoir (⟨msg⟩)
5     if msg ∉ msgHistory then
6         Livrer(⟨msg⟩) ;
7         msgHistory ← msgHistory ∪ {⟨msg⟩} ;
8         Gossip(⟨msg⟩,paramètres) ;
```

Il y a trois grandes classes d'algorithmes de gossip pour implémenter la procédure Gossip() : (1) le gossip avec fanout fixé (*GossipFF*), (2) le gossip avec choix d'arêtes probabiliste (*GossipPE*) et (3) le gossip basé sur une diffusion probabiliste (*GossipPB*). En plus du message reçu, chacun de ces algorithmes reçoit d'autres paramètres dont la valeur est identique pour tous les sites.

```
9  /* fanout : nombre de voisins sélectionnés */
10 GossipFF (⟨msg⟩,fanout)
11   if fanout ⩾ V_i then
12     toSend ← Λ_i
13   else
14     toSend ← ∅
15     for f = 1 to fanout do
16       aléatoirement selectionner s_j ∈ Λ_i/toSend
17       toSend ← toSend ⋃ s_j
18   foreach s_j ∈ toSend do
19     Envoyer (⟨msg⟩, s_j)
```

Algorithme 2 : Gossip avec fanout fixé (dans s_i)

```
20 /* p_e : probabilité d'utiliser l'arête */
21 GossipPE (⟨msg⟩,p_e)
22   foreach s_j ∈ Λ_i do
23     if Random() ⩽ p_e then
24       Envoyer (⟨msg⟩, s_j)
```

Algorithme 3 : Gossip avec choix d'arêtes probabiliste (dans s_i)

```
25 /* p_ν : probabilité de broadcast */
26 GossipPB (⟨msg⟩,p_ν)
27   if Random() ⩽ p_ν then
28     foreach s_j ∈ Λ_i do
29       Envoyer (⟨msg⟩, s_j)
```

Algorithme 4 : Gossip basé sur une diffusion probabiliste (dans s_i)

Dans *GossipFF* (Algorithme 2), le site s_i envoie le message msg à un nombre fixé de sites, noté $fanout$, qui sont aléatoirement sélectionnés dans Λ_i (lignes 15-17). Notons que si $fanout \geqslant V_i$, s_i transmet msg à tous ses voisins (lignes 11 et 12). Particulièrement, si $fanout \geqslant \max\{V_1, V_2, \cdots V_N\}$, l'algorithme 2 devient un algorithme d'inondation.
Dans les algorithmes suivants, `Random()` génère un nombre aléatoire dans l'intervalle $[0, 1]$.
Dans *GossipPE* (Algorithme 3), chaque site choisit avec une probabilité p_e les arêtes sur lesquelles msg est retransmis (voir ligne 23). Notons que si $p_e = 1$ pour tous les sites, nous obtenons l'algorithme d'inondation.
Contrairement à *GossipPE*, dans *GossipPB* (Algorithme 4), chaque site, excepté la source, diffuse msg à tous ses voisins avec une probabilité p_ν (lignes 27-29). Si $p_\nu = 1$, ce protocole revient à un algorithme d'inondation.

4. Métriques de performance

Les métriques suivantes sont couramment utilisées dans les littératures [12, 14, 15] pour l'évaluation des performances des algorithmes de dissémination.
La complexité en messages, notée M : mesure le nombre de messages envoyés (ou reçus, car il n'y pas de perte de message) par chaque site, $M = \frac{\Omega}{N-1}$ où Ω est le nombre total de messages échangés pendant la dissémination.
La fraction de sites infectés, notée α : définit le pourcentage de sites dans le système qui ont reçu le message généré par la source à la fin de la dissémination.
La fiabilité, notée R : définit le pourcentage de messages générés par la source, qui ont été reçus par tous les sites. La fiabilité égale à 100% indique que l'algorithme réussit à diffuser tout message généré par la source à tous les autres sites dans le système, ce qui assure *l'atomicité* [14].
La latence, notée L : mesure le nombre de sauts nécessaire pour diffuser un message à tous les destinataires, c'est-à-dire, la longueur du chemin le plus long parmi les chemins les plus courts de la source à tous les sites qui ont reçu le message.
Un algorithme de gossip efficace vise à assurer une grande fraction de sites infectés et une forte fiabilité, en minimisant à la fois la complexité en messages et la latence.

5. État de l'art

De nombreuses études ont analysé la performance des trois algorithmes sur les trois topologies en termes de fiabilité, complexité en messages et latence.
Dans [14], les auteurs étudient la fiabilité de la dissémination d'informations en appliquant l'algorithme *GossipFF* dans $\mathcal{B}(N, p_N)$. En supposant que le *fanout* de tous les sites dans le système

est inférieur au nombre de ses voisins, cet article conclut que pour qu'un système avec N sites atteigne la fiabilité R, il faut fixer $fanout = -\ln\left(\frac{-\ln(R)}{N}\right)$. Des études reposant sur des simulations [6, 8], aboutissent à des résultats similaires. Cependant, ils n'ont considéré que *GossipFF* dans $\mathcal{B}(N, p_N)$.
La performance de *GossipPB* dans $\mathcal{G}(N, \rho)$ est discutée et implémentée dans [3, 12], alors que [4] a théoriquement analysé *GossipPB* dans $\mathcal{B}(N, p_N)$. La performance de *GossipPB* dans $\mathcal{G}(N, \rho)$ est également étudiée de façon théorique dans [23], dans le but de choisir p_ν afin d'atteindre une forte fiabilité. Á partir de la discussion dans [17] sur la fiabilité, en s'appuyant sur la propriété de percolation dans $\mathcal{B}(N, p_N)$, les auteurs proposent une expression asymptotique du nombre moyen de messages et de la latence moyenne nécessaire pour atteindre l'atomicité. Néanmoins, par rapport à nos recherches, leurs efforts se sont focalisés sur la performance d'un algorithme de gossip dans une topologie aléatoire précise. Sur les deux topologies aléatoires $\mathcal{S}(N, m)$ et $\mathcal{B}(N, p_N)$, la latence d'une version modifiée de *GossipPB*, qui permet tout site de renvoyer le message plusieurs fois à un de ses voisins avec certaine probabilité, est analysée par le modèle SIS (Susceptible-Infection-Susceptible) dans [9].
Dans [10], les auteurs montrent que la performance des trois algorithmes de gossip dans $\mathcal{S}(N, m)$ est meilleure que dans $\mathcal{B}(N, p_N)$. *GossipPE* présente de meilleures performances que *GossipPB* dans $\mathcal{G}(N, \rho)$, en fonction de la taille du système (ou le degré du site) dans [22]. En plus, *GossipFF*, *GossipPE* et *GossipPB* dans $\mathcal{S}(N, m)$ sont comparés de la même manière dans [7]. Dans [21], le choix entre *GossipPE* et *GossipPB* dépend des contraintes des différentes applications dans $\mathcal{G}(N, \rho)$.
Néanmoins, pour une fiabilité donnée, il n'existe pas de méthode générale pour obtenir les gains quantitatifs pour tous les algorithmes dans les différents graphes en termes de complexité en messages. Nous introduisons donc un nouveau paramètre générique : le *fanout effectif*, qui exprime la puissance moyenne de dissémination des sites infectés et permet les comparaisons équitables des algorithmes de gossip.

6. Fanout effectif

Le nombre de messages retransmis des trois algorithmes dépend des paramètres d'entrée (p_ν, p_e ou $fanout$) qui sont très différents. Dans le but de faire une comparaison équitable entre ces algorithmes de gossip sur les topologies décrites dans la section 2, nous introduisons un nouveau paramètre : le ***fanout effectif*** noté F_{eff}. Ce paramètre permet des analyses précises du comportement d'un algorithme de gossip sur une topologie. Il simplifie la comparaison des différents algorithmes sur une topologie. Ainsi, pour un algorithme et une topologie donnés, F_{eff} caractérise la puissance moyenne de dissémination des sites infectés. Pour une topologie fixée, il est possible de déduire les paramètres d'entrée à partir du fanout effectif.
Pour les algorithmes *GossipPE*, *GossipPB* et *GossipFF*, nous définissons respectivement que :

$$F_{eff}^{GossipPE} = p_e \cdot \bar{V} \tag{1}$$

$$F_{eff}^{GossipPB} = p_\nu \cdot \bar{V} \tag{2}$$

$$F_{eff}^{GossipFF} = \sum_{k=1}^{fanout-1} P(k) \cdot k + \sum_{k=fanout}^{N-1} P(k) \cdot fanout \tag{3}$$

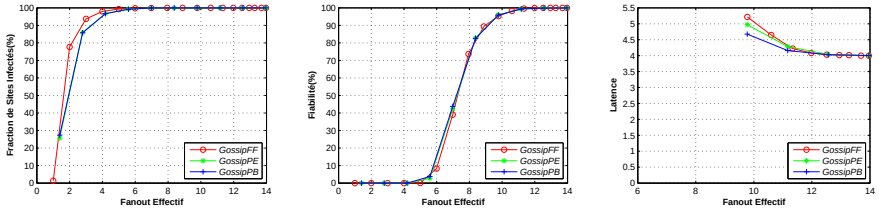
7. Évaluation de performance

Dans cette section, nous présentons et discutons des résultats obtenus par les expérimentations dans le simulateur OMNET++ [1]. Nous évaluons les métriques présentées dans la section 4 pour les trois algorithmes sur les trois topologies de graphes aléatoires.
Pour que nos comparaisons soient équitables, nous avons calculé le fanout effectif de chaque algorithmique en faisant varier le paramètre d'entrée (voir la section 6). Les résultats expérimentaux sont présentés en fonction du fanout effectif. Pour un fanout effectif fixé, nous comparons la fraction de sites infectés, la fiabilité et la latence des algorithmes sur les topologies. Nous considérons que des fiabilités supérieures à 90%.
Le réseau compte $N = 1000$ sites et, afin d'assurer la connectivité, $\varepsilon = 1$. Pour que toutes les topologies possèdent le même degré moyen ($\bar{V} \approx 14.0$), les paramètres des topologies suivants ont été choisis dans le tableau de la figure 2 :

TOPOLOGIES	PARAMETRES
$\mathcal{B}(N, p_N)$	$p_N = 0.014$
$\mathcal{G}(N, \rho)$	$a = 7500$, $b = 3000$, $\rho = 330$
$\mathcal{S}(N, m)$	$m_0 = 9$ (m_0-clique), $m = 7$

FIGURE 2 – Paramètres des topologies

Pour chaque algorithme de gossip, 200 différents messages sont générés par 200 différentes sources aléatoirement choisies parmi 1000 sites sur 20 different graphes liés à chacune des topologies aléatoires. Les résultats pour chaque fanout effectif sont moyennés par les $200 \times 20 = 4000$ disséminations de message.



(a) Sites infectés dans $\mathcal{B}(N, p_N)$ (b) Fiabilité dans $\mathcal{B}(N, p_N)$ (c) Latence dans $\mathcal{B}(N, p_N)$

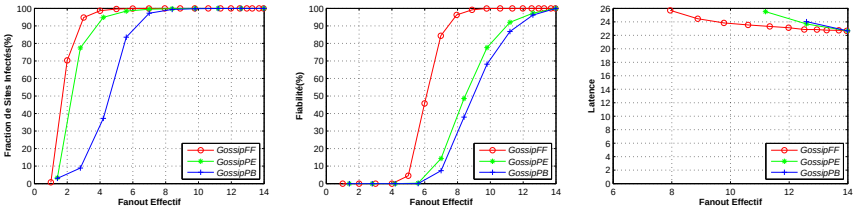
FIGURE 3 – Comparison des performances des algorithms dans $\mathcal{B}(N, p_N)$.

7.1. Équivalence des algorithmes dans $\mathcal{B}(N, p_N)$

Nous évaluons les performances des algorithmes de gossip sur $\mathcal{B}(N, p_N)$. D'une manière générale les résultats des figures 3(a) et 3(b) montrent que la fiabilité (R) et la fraction de sites infecté (α) sont nulles jusqu'à un certain seuil puis augmente rapidement à 100%. D'autre part, on voit que les performances des trois algorithmes sont les mêmes pour un même F_{eff}. Cependant, les valeurs des seuils pour les métriques R et α sont différentes : α percole plus tôt que R. En effet, lorsque α augmente très rapidement, presque tous les sites reçoivent chaque message

($\alpha \approx 100\%$) mais aucun message n'est reçu par tout le monde ($R \approx 0$). Ce n'est qu'après un seuil plus grand que presque tous les messages seront reçus par tout le monde ($R \approx 100\%$).
Comme les performances sont les mêmes pour tous les algorithmes, le F_{eff} permet d'utiliser le resultat théorique de *GossipFF* dans [14] : $fanout = -\ln\left(\frac{-\ln(R)}{N}\right)$, pour déterminer les paramètres d'entrée correspondants aux seuils de *GossipPE* and *GossipPB*. Par exemple, pour $R = 99,4\%$, $fanout = -\ln\left(\frac{-\ln(,994)}{1000}\right) \approx 12$. Selon l'équation (3) dans la section 6, on obtient $F_{eff} = 11,3$. D'où, $p_e = p_\nu = F_{eff}/\bar{V} = 11,3/14 = 0,81$. Il devient donc possible de dimensionner les probabilités pour obtenir la fiabilité R désirée.
Dans la figure 3(c), après une certaine valeur du fanout effectif, la latence ne décroît plus, mais reste constante convergeant vers l'algorithme de flooding, c'est-à-dire, le plus grand chemin parmi le plus courts chemins entre la source et les autres sites, et alors vers la latence minimum.

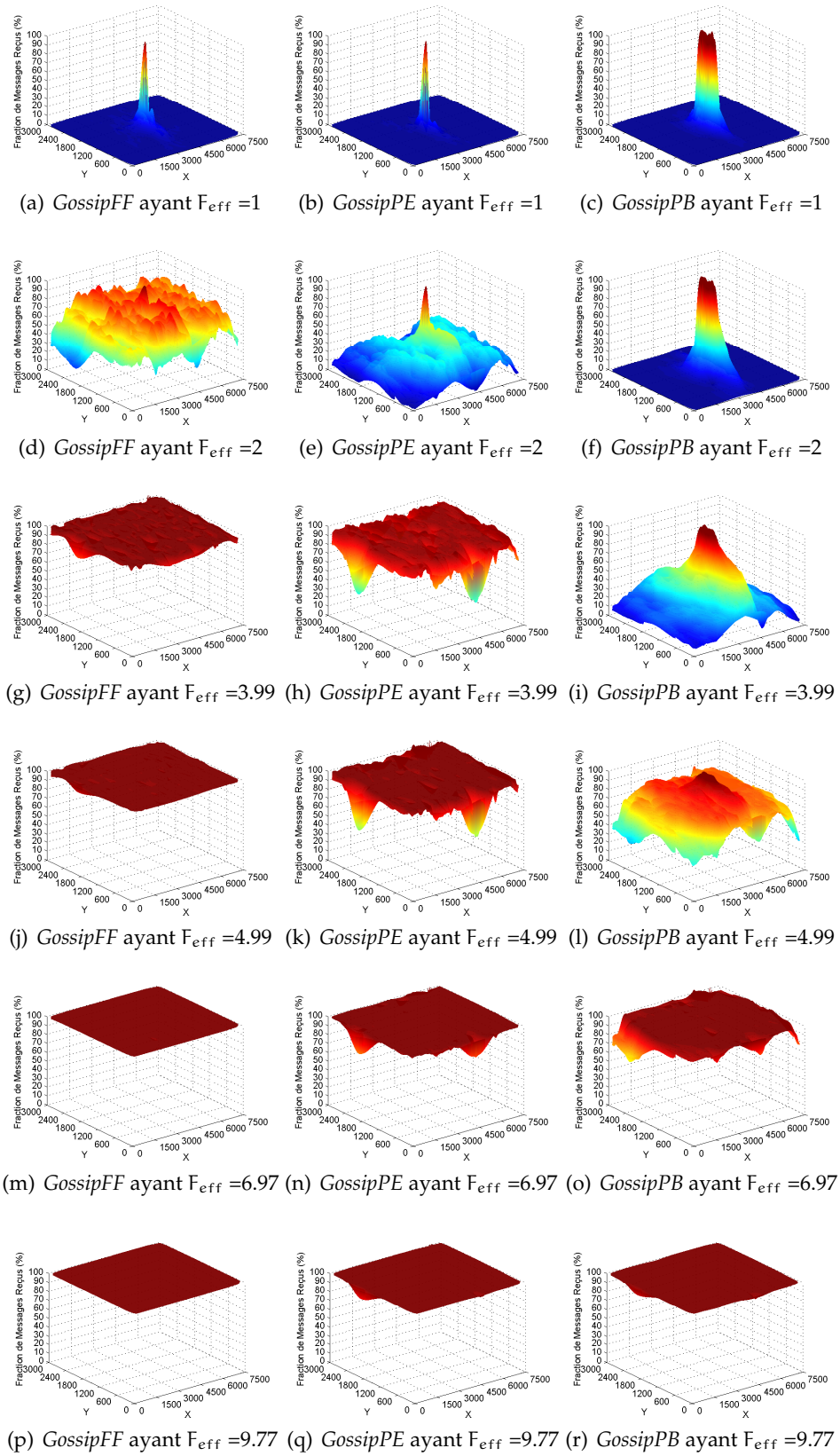
7.2. Différence des algorithmes dans $\mathcal{G}(N, \rho)$



(a) Sites infectés dans $\mathcal{G}(N, \rho)$ (b) Fiabilité dans $\mathcal{G}(N, \rho)$ (c) Latence dans $\mathcal{G}(N, \rho)$

FIGURE 4 – Comparison de performance des algorithmes dans $\mathcal{G}(N, \rho)$.

Si les performances des algorithmes était identiques dans $\mathcal{B}(N, p_N)$, ce n'est pas toujours le cas. Ainsi, si l'on considère maintenant la topologie $\mathcal{G}(N, \rho)$, la figure 4 montre de grandes différences entre les différents algorithmes. Ainsi, si l'on considère la fiabilité dans la figure 4(b), on s'aperçoit que *GossipFF* est plus performant (99% pour F_{eff} de 8,5) que les deux algorithme (99% pour F_{eff} de 14). De plus, si c'est les derniers qui gardent un effet de seuil, celui-ci est beaucoup plus atténué (de 5,5 à 14), *GossipPE* restant un tout petit peu meilleur que *GossipPB*. Si l'on regarde maintenant les résultats en termes de sites infectés, le résultat est tout autre. *GossipFF* reste le meilleur mais *GossipPE* présente des performance assez similaire. *GossipPB* présente lui de bien moins bonne performance : il faut un fanout effectif d'environ 8 pour pouvoir contaminer presque tout le monde.
Le comportement de la latence dans la figure 4(c) pour $\mathcal{G}(N, \rho)$ est similaire à celui pour $\mathcal{B}(N, p_N)$, sauf que la latence minimum soit environ 23 hops, parce que le diamètre du premier est plus grand que le dernier.
Pour analyser ces résultats nous avons réalisé une séries d'expériences spécifiques en plaçant la source au centre d'un rectangle 3000×7500 contenant 1000 sites. Le degré moyen correspondant à un rayon de $\rho = 330$ est de 14. Les résultats sont présentés sur la figure 5 sous forme de courbes en 3 dimensions correspondant à plusieurs fanout effectifs, où à chaque coordonné (x, y) d'un site on fait correspondre le pourcentage de messages reçus. Plus le point est haut plus le site a reçu de messages. Un point de hauteur zéro correspond donc à un site qui n'a reçu aucun message.
En regardant les performances de *GossipFF* sur la première colonne de la figure 5, on peut vérifier qu'il est le premier ayant contaminé tous les sites ($F_{eff} = 6,97$) tandis que les deux autres



(a) *GossipFF* ayant F_{eff} =1 (b) *GossipPE* ayant F_{eff} =1 (c) *GossipPB* ayant F_{eff} =1

(d) *GossipFF* ayant F_{eff} =2 (e) *GossipPE* ayant F_{eff} =2 (f) *GossipPB* ayant F_{eff} =2

(g) *GossipFF* ayant F_{eff} =3.99 (h) *GossipPE* ayant F_{eff} =3.99 (i) *GossipPB* ayant F_{eff} =3.99

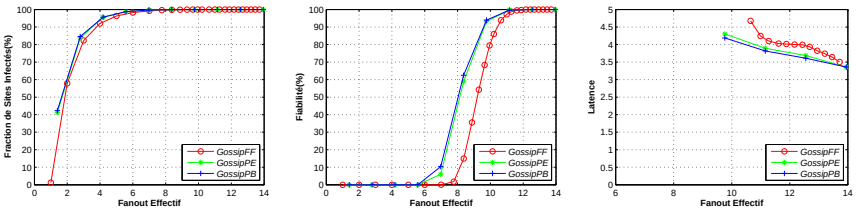
(j) *GossipFF* ayant F_{eff} =4.99 (k) *GossipPE* ayant F_{eff} =4.99 (l) *GossipPB* ayant F_{eff} =4.99

(m) *GossipFF* ayant F_{eff} =6.97 (n) *GossipPE* ayant F_{eff} =6.97 (o) *GossipPB* ayant F_{eff} =6.97

(p) *GossipFF* ayant F_{eff} =9.77 (q) *GossipPE* ayant F_{eff} =9.77 (r) *GossipPB* ayant F_{eff} =9.77

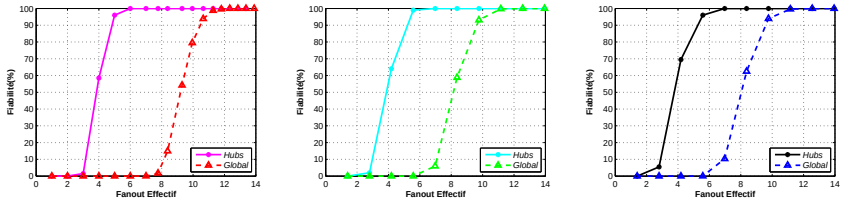
FIGURE 5 – Reception de messages de tous les sites dans $\mathcal{G}(N, \rho)$.

(colonnes 2 et 3) n'ont pas encore contaminé tout le monde pour ($F_{eff} = 9,77$). Même si les performances de ces deux algorithmes finissent par être identiques, l'évolution des infections est très différente.
Sur la dernière colonne on remarque que *GossipPB* conserve longtemps une forme de pic sur les graphiques. Cette forme implique que les sites infectés sont regroupés autour de la source : le message s'est arrêté rapidement. Ceci s'explique par un effet de clustering induit par la probabilité d'émission. En effet, dans cet algorithme les sites ne renvoient pas du tout le message avec une probabilité $1 - p_b$. Si cette probabilité est trop grande il se forme une clôture de site n'émettant pas autour de la source (la bordure du pic). Ce n'est que lorsque la probabilité augmente que l'effet de clustering s'arrête. Le message est alors reçu par tout le monde.
A l'inverse pour *GossipPE* (voir la deuxième colonne) en accroissant un tout petit peu le fanout effectif, très vite presque tous les sites reçoivent tous les messages. Mais contrairement à *GossipFF* il reste toujours certains sites problématiques : sur les bords mais aussi dans les zones où la densité des sites est faible. En effet, l'algorithme impose un tirage au sort pour chaque arrête quel que soit le degré. Sur l'ensemble des sites ayant peu de voisins il en est toujours ceux qui ne reçoivent pas un message. Ceci explique la mauvaise fiabilité (voir la figure 4(b)), alors que presque tous les sites sont infectés (voir la figure 4(a)).
Cette étude montre donc pourquoi *GossipFF* est particulièrement performant : en forçant chaque site à renvoyer certains messages il évite l'effet de clustering qui arrête l'algorithme de gossip, et en obligeant les sites ayant peu de voisins (c'est-à-dire que leurs degrés sont inférieurs à `fanout`) à retransmettre à tous leurs voisins il traite efficacement les zones à risque.



(a) Sites infectés dans $\mathcal{S}(N, m)$ (b) Fiabilité dans $\mathcal{S}(N, m)$ (c) Latence dans $\mathcal{S}(N, m)$

FIGURE 6 – Comparaison de performance des algorithmes dans $\mathcal{S}(N, m)$.



(a) Fiabilité par *GossipFF* (b) Fiabilité par *GossipPE* (c) Fiabilité par *GossipPB*

FIGURE 7 – Fiabilité de tous les hubs et de tous les sites par les algorithmes de gossip dans $\mathcal{S}(N, m)$

7.3. Différence des algorithmes sur $\mathcal{S}(N, m)$

Si l'on regarde maintenant les performances des différents algorithmes de gossip dans le cas d'une topologie $\mathcal{S}(N, m)$ (voir la figure 6), on trouve des résultats complètement différents des deux autres topologies. Comme pour le graphe géométrique aléatoire les trois algorithmes présentent les mêmes performances en termes de fraction de sites infectés (voir la figure 6(a)). Mais la fiabilité du *GossipFF* est ici pire que celle des deux autres approches.

Ce phénomène s'explique par la distribution des degrés dans ces graphes qui conduit à la présence de hub (voir la section 2). Pour comprendre l'importance de ces hub nous avons mesurer pour différents F_{eff} la fiabilité dans le sous-graphe des hubs, c'est-à-dire la proportion des messages qui sont reçus par tous les hubs du graphe. Les résultats sont présentés dans la figure 7, en les comparant pour chacun des algorithmes de gossip à la fiabilité du système global. L'étude de ces graphes montre que les hubs sont infectés en priorité quelque soit l'algorithme. Ainsi, avec un F_{eff} de 6 la fiabilité des hubs est de 100%, alors que presque aucun message n'est reçu par tout le monde et la fiabilité du système global est nulle.

Les mauvaises performances du *GossipFF* s'explique par la mauvaise exploitation de ces hubs. En effet, alors que le degré de ces derniers est énorme, l'algorithme limite leur pouvoir d'émission à $fanout$ voisins. Or, il faut comprendre qu'un envoi de 10 messages par un site est plus puissant que 1 envoi par 10 sites. En effet, dans le premier cas tous les destinataire sont différents, ce qui assure une bonne dissémination du message en diminuant la redondance de message.

Cette mauvaise utilisation des hubs se retrouve aussi dans la latence (voir la figure 6(c)). Si *GossipPB* et *GossipPE* présentent la même latence, il n'en est pas de même pour *GossipFF* qui conserve une latence supérieure alors que sa fiabilité est proche de 100%). En effet, dans $\mathcal{S}(N, m)$ les hubs représentent le cœur du réseau : presque tous les sites périphériques sont connectés à un hub. En limitant le puissance d'émission des hubs *GossipFF* se prive de nombreux raccourcis.

7.4. Impact des topologies sur les algorithmes

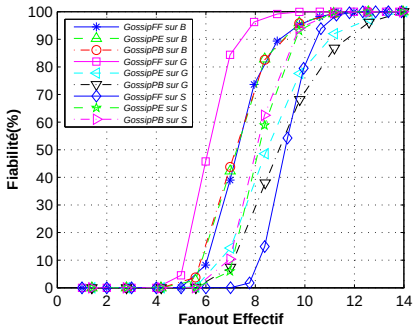


FIGURE 8 – Impact des topologies sur les algorithmes

	Faible variance des degrés	**Forte variance des degrés**
Faible dépendance des arêtes	$\mathcal{B}(N, p_N)$: ***GossipFF,GossipPE, GossipPB***	$\mathcal{S}(N, m)$: ***GossipPE, GossipPB***
Forte dépendance des arêtes	$\mathcal{G}(N, \rho)$: ***GossipFF***	——

FIGURE 9 – Choix des algorithmes

Comme tous les graphes sont du même degré moyen ($\bar{V} \approx 14.0$), nous pouvons comparer la fiabilité des algorithmes dans les différentes topologies (figure 8). Le tableau de la figure 9 résume les choix convenables des algorithmes de gossip. Quand le graphe possède une faible *dépendance des arêtes* (dans $\mathcal{B}(N, p_N)$), les trois algorithmes présentent le même comportement. Quand la *dépendance des arêtes* (respectivement, la *variance des degrés*) est introduite dans le graphe, mais

la *variance des degrés* (respectivement, la *dependence des arêtes*) n'est pas changée comme dans $\mathcal{G}(N, \rho)$ (respectivement, $\mathcal{S}(N, m)$), la performance de *GossipPB* (respectivement, *GossipFF*) se dégrade.
De tels résultats confirment que le meilleur choix des algorithmes pour la fiabilité avec la même complexité en messages dépend de la topologie sous-jacente. Il est intéressant de souligner que la performance de *GossipPE* n'est jamais pire que *GossipPB*, parce que dans la théorie de percolation [11], le premier peut être modélisé par la percolation de lien tandis que le dernier correspond à la percolation de site. Le seuil de percolation pour la percolation de site est toujours plus grand que celui de la percolation de lien dans toute topologie.
Nous aussi mesurons le gain relatif du fanout effectif des algorithmes de gossip dont leur fiabilité atteint 99% et 80%, et la fraction de sites infectés est très importante (c'est-à-dire, α est proche de 100%) par rapport à l'algorithme de flooding (c'est-à-dire, la complexité en messages maximale). Les résultats sur les trois topologies sont montrés dans les figures 10 et 11 respectives. Nous observons que sur $\mathcal{G}(N, \rho)$ en ayant $R = 99\%$ le gain de *GossipPB* et de *GossipPE* est zéro. Donc, ils ont besoin presque la même complexité en messages que l'algorithme de flooding. D'autre part, afin d'obtenir $R = 80\%$, $\mathcal{B}(N, p_N)$ s'expose le meilleur gain de *GossipPB* et de *GossipPE* parmi toutes les topologies aléatoires. De plus, *GossipFF* sur $\mathcal{G}(N, \rho)$ est la meilleure combinaison pour atteindre le gain le plus important.

	$\mathcal{B}(N, p_N)$	$\mathcal{S}(N, m)$	$\mathcal{G}(N, \rho)$
GossipFF	14%	14%	**43%**
GossipPB	14%	**21%**	0%
GossipPE	14%	**21%**	0%

FIGURE 10 – Gain en termes de fanout effectif en ayant $R = 99\%$

	$\mathcal{B}(N, p_N)$	$\mathcal{S}(N, m)$	$\mathcal{G}(N, \rho)$
GossipFF	40%	23%	**52%**
GossipPB	40%	**34%**	27%
GossipPE	40%	**34%**	31%

FIGURE 11 – Gain en termes de fanout effectif en ayant $R = 80\%$

En conclusion, afin de réduire la complexité en messages entraînée par l'algorithme de gossip par rapport à l'algorithme de flooding, il est nécessaire de considérer à la fois l'algorithme et la topologie.

8. Conclusion

Pour étudier les algorithmes de gossip, nous avons introduit un nouveau paramètre, le fanout effectif, qui prend en compte la distribution des degrés de la topologie. Ainsi, le fanout effectif est utile pour comparer des algorithmes de gossip s'exécutant sur des topologies aléatoires avec un distributions des degrés différentes. Il caractérise la puissance moyenne de dissémination des sites infectés dans le système.
Grâce à lui, nous avons pu équitablement évaluer les performances et étudier les compromis entre la fiabilité, la proportion de sites infectés, la complexité en messages et la latence des trois grandes familles d'algorithmes sur trois topologies aléatoires typiques. Nous avons ainsi montré qu'en termes de fiabilité, *GossipFF* est le meilleur choix dans $\mathcal{G}(N, \rho)$ mais le pire dans $\mathcal{S}(N, m)$, et que les trois algorithmes se présentent les performances équivalentes sur $\mathcal{B}(N, p_N)$.
Les résultats obtenus dans notre étude aident à choisir le meilleur algorithme de gossip en fonction de la topologie aléatoire sur laquelle il s'exécute. Le fanout effectif peut être également utilisé pour analyser théoriquement le comportement des algorithmes.

Bibliographie

1. Omnet++ : Discrete event simulation system *http ://www.omnetpp.org/*.
2. Albert (R.) et Barabási (A.-L.). – Statistical mechanics of complex networks. *Reviews of Modern Physics*, vol. 74, Jan. 2002, pp. 47–97.
3. Blywis (B.), Günes (M.), Juraschek (F.) et Hofmann (S.). – Gossip routing in wireless mesh networks. *ISPIMRC*, 2010, pp. 1572–1577.
4. Crisostomo (S.), Schilcher (U.), Bettstetter (C.) et Barros (J.). – Analysis of probabilistic flooding : How do we choose the right coin ? *ICC*, Jun. 2009, pp. 1–6.
5. Erdős (P.) et Rényi (A.). – On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, vol. 5, n17, 1960, pp. 17–60.
6. Eugster (P. T.), Guerraoui (R.), Handurukande (S. B.), Kermarrec (A.-M.) et Kouznetsov (P.). – Lightweight probabilistic broadcast. *ACM Transaction on Computer Systems*, vol. 21, 2003, pp. 341–374.
7. Ferretti (S.) et D'Angelo (G.). – Multiplayer online games over scale-free networks : a viable solution ? *ICST*, 2010.
8. Frey (D.), Guerraouia (R.), Kermarrec (A.-M.), Koldehofe (B.), Mogensen (M.), Monod (M.) et Quéma (V.). – Heterogeneous gossip. *Int. Conf. on Middleware*, 2009, pp. 42–61.
9. Ganesh (A.), Massoulié (L.) et Towsley (D.). – The effect of network topology on the spread of epidemics. *INFOCOM*, 2005, pp. 1455–1466.
10. Garbinato (B.), Rochat (D.) et Tomassini (M.). – Impact of scale-free topologies on gossiping in ad hoc networks. *NCA*, 2007, pp. 269–272.
11. Grimmett (G.). – *Percolation*. – Springer, 1989.
12. Haas (Z.), Halpern (J.) et Li (L.). – Gossip-based ad hoc routing. *INFOCOM*, vol. 3, 2002, pp. 1707–1716.
13. Jetcheva (J.), Hu (Y.), Maltz (D.) et Johnson (D.). – A simple protocol for multicast and broadcast in mobile ad hoc networks. *Internet Draft : draft-ietf-manet-simple-mbcast-01.txt*, 2001.
14. Kermarrec (A.-M.), Massoulié (L.) et Ganesh (A.). – Probabilistic reliable dissemination in large-scale systems. *IEEE TPDS*, vol. 3, Mar. 2003, pp. 248–258.
15. Leitao (J.), Pereira (J.) et Rodrigues (L.). – Epidemic broadcast trees. *Int. SRDS*, 2007, pp. 301–310.
16. Newman (M. E. J.). – The structure and function of complex networks. *SIAM REVIEW*, vol. 45, 2003, pp. 167–256.
17. Oikonomou (K.), Kogias (D.) et Stavrakakis (I.). – Probabilistic flooding for efficient information dissemination in random graph topologies. *Computer Networks*, vol. 54, 2010, pp. 1615–1629.
18. Penrose (M.). – *Random Geometric Graphs*. – Oxford University Press, 2003, oxford studies in probability éditionvolume 5.
19. Philips (T.), Panwar (S.) et Tantawi (A.). – Connectivity properties of a packet radio network model. *IEEE Transactions on Information Theory*, 1989, pp. 1044–1047.
20. Polynikis (A.). – *Random Walks and Scale-Free Networks*. – Thèse, University of York, UK, 2006.
21. Raman (V.) et Gupta (I.). – Performance tradeoffs among percolation-based broadcast protocols in wireless sensor networks. *W. on Wireless Ad hoc and Sensor Networking*, 2009, pp. 158–165.
22. Shen (C.-C.), Huang (Z.) et Jaikaeo (C.). – Directional broadcast for mobile ad hoc networks with percolation theory. *IEEE Transactions on Mobile Computing*, vol. 5, n4, Apr. 2006, pp. 317–332.
23. Vakulya (G.). – Energy efficient percolation-driven flood routing for large-scale sensor network. *IMCSIT*, vol. 3, 2008, pp. 877–883.