Repairable Fountain Codes with Unequal Repairing Locality in D2D Storage System

Yue Li, Shushi Gu(✉), Ye Wang, Juan Li, and Qinyu Zhang

Communication Engineering Research Center, Harbin Institute of Technology, Shenzhen, Guangdong, China
{liyue,lijuan2016}@stu.hit.edu.cn, {gushushi,wangye83,zqy}@hit.edu.cn

Abstract. In this paper, we propose a novel repairable fountain codes (RFC) used in D2D data storage systems for failure data recovery. This RFC has the priority of unequal repairing locality (URL), which can provide unequal data protection for different nodes' bandwidth and power in different areas. The lower locality of URL-RFC can reduce the repair bandwidth in D2D storage system, and tradeoff different nodes' capabilities of transmitting. We firstly give the heterogeneous D2D storage network model, and analysis the communication cost for data download and node repair. Then, the construction method of URL-RFC is given based on generated matrix. Simulation results show that, URL-RFC significant outperforms conventional distributed codes on communication cost in heterogeneous D2D storage system.

Keywords: D2D data storage system · Repairable fountain codes
Unequal repairing locality · Repair and download bandwidth

1 Introduction

The rapid growth of mobile data traffic has caused tremendous pressure on distributed storage systems (DSS) and cellular base stations (BSs). A rising technique called device-to-device (D2D) distributed storage, is proposed to relieve the pressure of BS, which also takes advantage of the increasingly powerful storage capacity of mobile devices [1,2]. Presently, the redundancy has been brought into DSS, such as replication scheme, MDS codes and regenerating code, which have derived some crucial research achievements in the application DSS in D2D networks [3–5]. The focus of these research is on how to reduce the repair cost and download cost of the system.

Repairable fountain code (RFC) is a new family of fountain codes that can be applied to DSS, that is a rateless and systematic code and has low locality [6]. Therefore, this new family of RFC has great application potential in

J. Zheng et al. (Eds.): ADHOCNETS 2018, LNICST 258, pp. 272–281, 2019.
https://doi.org/10.1007/978-3-030-05888-3_25

the D2D distributed storage system with large network heterogeneity (nodes' bandwidth, communication distance, or device reception or transmitting signal power). However, the problem of further reducing the communication cost for the heterogeneous network of D2D distributed storage is still difficult to solve. The research of fountain code in the field of unequal error protection (UEP) in broadcast transmission has been extensive [7,8], and the local reconstruction code have also been studied preliminarily in terms of unequal failure protection [9] in file storage system.

This paper proposes a novel RFC for D2D distributed storage systems, which is mainly used for the repair of failure data. Our new RFC has different priorities for repairing locality (URL) and can provide different data protection for nodes with different bandwidth and power in different areas. The lower locality of URL-RFC can reduce the repair bandwidth cost in D2D storage systems and tradeoff transmission capacity of different nodes. We first give a heterogeneous D2D storage network model and analyze the data download and repair communication cost. Then, based on the generator matrix, the URL-RFC construction method is given. Simulation results show that, in heterogeneous D2D storage systems, the communication cost of URL-RFC is better than that of traditional distributed systems.

The structure of this paper: Sect. 2 gives the system model of heterogeneous D2D distributed storage. In Sect. 3, the constructions method of RFC and the URL-RFC are introduced. In Sect. 4, we analyze the communication cost of the URL-RFC scheme. In Sect. 5, simulation results show that URL-RFC can reduce communication cost. Finally, the Sect. 6 is the summary of the main research in this paper.

2 System Model

We consider the D2D distributed storage system shown in Fig. 1. A base station (BS) can cover two areas at the same time. The mobile devices in the red area have a stronger D2D communication capability than the mobile devices in the green area, because the antenna size and power amplifier capability of the devices in the two areas are different. The yellow cell phone represents the node that requested the file (requester), the blue cell phone represents the new storage node (empty), and the red and green cell phones represent nodes that store the data (helper).

The Working Process of the System is as Follows:
As the nodes enter and exit (Poisson process) the coverage area of the BS randomly, the number of storage nodes in the system may be insufficient, that is, the file data may be lost. The requester will request to download all data with a certain probability, so the lost node (data) must be repaired. When the D2D communication capability is insufficient or does not have the enough number of nodes to repair data, it is necessary to rely on the BS for data download and repair. Due to the differences in device capabilities between heterogeneous areas, there will be differences in the energy costs of downloading data and repairing data between the two areas.

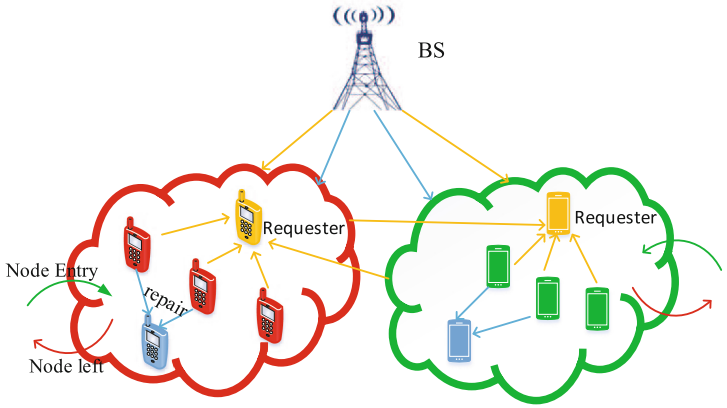

Fig. 1. Heterogeneous D2D distributed storage system model. (Color figure online)

The research work in this article is based on the following basic assumptions:

1. Assume that the data storage space of each node is infinite, and we consider that there is a single file, of size M bits, stored at the BS.
2. We denote by ρ_{BS} the cost of transmitting one bit from the BS, ρ_r the cost of transmitting one bit between nodes in the red area and ρ_g the cost of transmitting one bit between nodes in the green area. We assume that $\rho_r > \rho_g$. The cost for one bit inter-area transmission is set to be the same, which is $\rho_{rg} = \rho_{gr}$.
3. According to pass loss of wireless signals, i.e. lager distance is required, more power is consumed. Therefore, the distance between the nodes of one area is smaller than the distance between the nodes of different areas, and the both are less than the distance from BS to nodes, i.e. $\rho_{BS} > \rho_{rg} = \rho_{gr} \geq \rho_r > \rho_g$.
4. The expected numbers of nodes in red and green areas are N_r and N_g, respectively. The incoming rates of nodes in the two areas are $N_r\lambda$ and $N_g\lambda$, respectively. The instantaneous number of nodes can be described by the $M/M/\infty$ Markov model. The probabilities that the number of nodes in the two areas are i satisfy Eqs. (1) and (2), respectively [10].

$$\pi_r(i) = \frac{{N_r}^i}{i!} e^{-N_r} \tag{1}$$

$$\pi_g(i) = \frac{{N_g}^i}{i!} e^{-N_g} \tag{2}$$

Data Storage: The file is divided into k packets, and k is composed of two parts, k_r and k_g. Then, the two parts are encoded by RFCs with parameters $(n_r, k_r, d(k_r))$ and $(n_g, k_g, d(k_g))$, respectively. Moreover, n is the number of encoding symbols, k is the number of input symbols and d is the repair locality. Finally, the encoded data generated by the two parts of packets are respectively stored in the nodes of red area and green area. For simplicity, we assume

$n_r \ll N_r$, $n_g \ll N_g$, that is, the number of storage nodes is much less than the number of expected nodes, then $\sum_{i=0}^{n_r-1} \pi_r(i) \ll 1$, $\sum_{i=0}^{n_g-1} \pi_g(i) \ll 1$. Therefore, the probability that the number of nodes in the red area is less than n_r can be negligible, while the probability that the number of nodes in the green area is less than n_g can also be negligible.

We consider a uniform allocation in our system model. Hence, each storage node stores α bits, then α is

$$\alpha = \frac{M}{k_r + k_g}. \tag{3}$$

File Download: Assume that the rate of each node requesting a file is ω, and the interval between the requests for the file in the red area and in the green area is $\frac{1}{N_r\omega}$, $\frac{1}{N_g\omega}$, respectively. When the number of storage nodes in the system is greater than or equal to $(1+\varepsilon)(k_r + k_g)$, the file download can be performed through D2D links, i.e. the download requires slightly more data than M bits. Otherwise, the entire file must be downloaded from the BS.

Data Repair: Assume that the departure rate of a node is λ. When the storage node leaves the system, the data stored in the node will also be lost. In order to reduce the power cost of repairing data, we consider to use the nodes in the same area to repair the lost data as much as possible. When the number of storage nodes in the same area is insufficient, the nodes in different areas can be used for repair. If the number of helper nodes is still insufficient, then we can use the BS to repair the data.

We consider the ideal conditions: When a storage node leaves, data repair is performed immediately and there is no repair delay. Because the probability that the number of nodes in the red area is smaller than n_r and the probability that the number of nodes in the green area is smaller than n_g are negligible. So, the storage nodes that can connect to when repairing in both of the two areas are sufficient, that is, the participation of nodes in another area and BS are not needed. In this case, the files can always be stored in the D2D distributed storage system and there is no need to download file from the BS. Therefore, the bandwidth costs of repairing one node's data in the red area and the green area are described as formula (4) and (5), respectively, where $\beta = \alpha$.

$$\gamma_r = d(k_r)\beta \tag{4}$$

$$\gamma_g = d(k_g)\beta \tag{5}$$

3 RFC and URL-RFC

3.1 Construction of RFC

Repairable Fountain Code (RFC) inherit the rateless property of classical fountain codes, so we do not need to pre-determine the number of coded symbols.

Unlike classical fountain codes, the RFC is a systematic code and its parity symbols also have logarithmic sparseness [6].

The repair fountain code divides the source file into k input symbols. The input symbols are encoded by a $k \times n$ generator matrix, resulting in n encoded symbols containing copies of k input symbols and n-k parities symbol. Each parity symbol is generated by a linear combination of $d(k) = clogk$ input symbols selected uniformly at random. The coefficient ω of the linear combination is selected from the finite field F_q, and c is a constant. Therefore, the generator matrix can be represented as $\mathbf{G} = [\mathbf{I_k}|\mathbf{P}]$, as shown in Fig. 2. The identity part of $\mathbf{G}$ corresponds to the systematic symbols, and the matrix $\mathbf{P}$ corresponds to the parity symbols.

$$\mathbf{G[I_k \mid P]} = \left[\begin{array}{cccc|ccc} 1 & \cdots & \cdots & 0 & \omega_{1,k+1} & \cdots & \omega_{1,n} \\ \vdots & \ddots & & \vdots & \vdots & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & \cdots & 1 & \omega_{k,k+1} & \cdots & \omega_{k,n} \end{array}\right]$$

k input symbol columns n-k parity symbol columns

Fig. 2. Generator matrix of RFC.

A parity symbol along with the systematic symbols covered by it form a local group. Any symbol in the local group can be reconstructed by the linear combination of other symbols in the local group, and the local goup size is $d(k) + 1$. The RFC trades its low locality with its MDS property, but it still possesses near-MDS property. When downloading the entire file, a very small decoding overhead $\varepsilon > 0$ is required, so that any subset of $k' = (1 + \varepsilon)k$ symbols can reconstruction the file. The maximum likelihood decoding method can be used for decoding, which is equivalent to solving the solutions of k' linear equations.

3.2 Unequal Repairing Locality Based on RFC

In D2D distributed storage systems, the repair cost is an important component of communication cost. If we do not consider the difference in the D2D communication capabilities of the nodes, the repair cost of the nodes in the red area of the Sect. 2 will be particularly large. Therefore, based on the RFC, this paper proposes an unequal repair locality code based on RFC (URL-RFC). This URL-RFC can reduce the repair cost of red area and reduce the overall communication cost. The specific plan is designed as follows:

The k input symbols are divided into k_r and k_g groups in a certain proportion. Let $u_1 \sim u_{k_r}$ represent the first set of input symbols, and $u'_1 \sim u'_{k_g}$ represent the second set of input symbols. Two sets of encoded symbols are generated by the two sets of input symbols and stored in the nodes of the red area and the green area, respectively, as shown in Fig. 3. The first set of n_r encoded symbols contains systematic symbols $v_1 \sim v_{k_r}$ and parity symbols with degree $d(k_r)$. The second set of n_g encoded symbols are similar to the first set. Based on the fourth assumption in Sect. 2, it is not necessary to generate the global parity symbols. The generator matrix $\mathbf{G[I_k|P_r|P_g]}$ in Fig. 4 is composed of three parts, namely a identity matrix corresponding to systematic symbols, $\mathbf{P_r}$ corresponding to the first set of parity symbols, and $\mathbf{P_g}$ corresponding to the second set of parity symbols. The encoding process can be expressed as

$$v = u\mathbf{G[I_k|P_r|P_g]}. \tag{6}$$

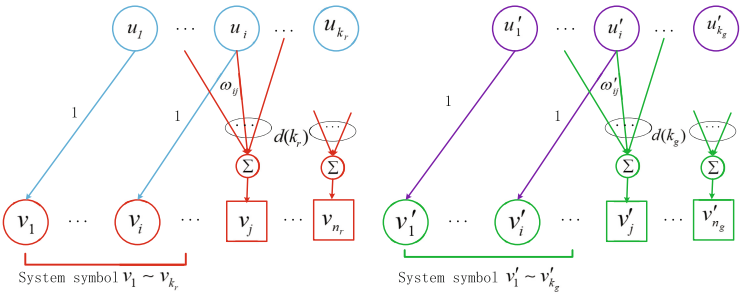

Fig. 3. URL-RFC encoding process. (Color figure online)

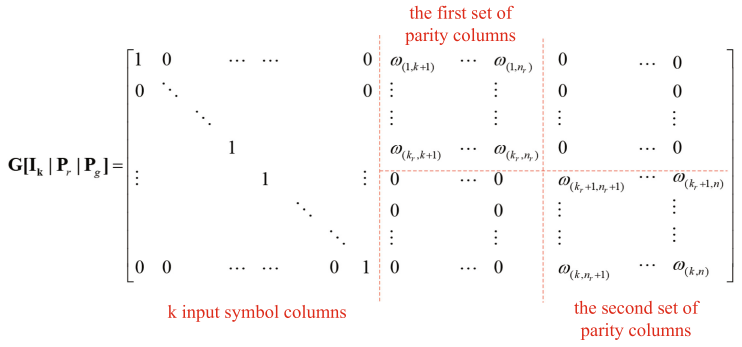

Fig. 4. URL-RFC generation matrix.

When decoding, the new generator matrix $\mathbf{G_S}$ is composed of the columns of the symbols in the available helper nodes, it is a sub-matrix of $\mathbf{G[I_k|P_r|P_g]}$.

If $\mathbf{G_S}$ is full rank, then we can use $u = v{\mathbf{G_S}}^{-1}$ to decode the input symbol. The condition for $\mathbf{G_S}$ full rank is that the number of helper nodes both in the red area and the green area are equal to or greater than $(1+\varepsilon)k_r$ and $(1+\varepsilon)k_g$ respectively. The encoded symbol v_j can be written as

$$v_j = u\mathbf{G}(j) = \sum \omega_{ij} u_i, \tag{7}$$

where $\mathbf{G}(j)$ represents the jth column of the generator matrix $\mathbf{G}[\mathbf{I_k}|\mathbf{P_r}|\mathbf{P_g}]$. When repairing an encoded symbol, we can connect the other symbols in the local group to reconstruct the encoded symbol by formula (7).

4 Repair and Download Cost

In this section we derive the analytical expressions for repair cost (C_{r-URL}), download cost (C_{d-URL}), total communication cost (C_{URL}) of URL-RFC and the repair cost (C_{r-ERL}), download cost (C_{d-ERL}), total communication cost (C_{ERL}) of Equal Repair Locality(ERL-RFC). Consider the system model in Sect. 2, which has the following parameters: N_r, N_g, n_r, n_g, k_r, k_g, M, λ, ω, ε, c, ρ_r, ρ_g, ρ_{rg}, ρ_{gr},α, γ_r, γ_g. We let $n = n_r + n_g$, $k = k_r + k_g$ and $N = N_r + N_g$. Without loss of generality, let the file size $M = 1$. The cost is defined in cost units per bit and time unit.

Average Download Cost: For ERL, the number of coded symbols in the red and green regions is randomly assigned in proportion to N_r and N_g.

The average download cost of URL-RFC is:

$$\begin{aligned} C_{d-URL} &= \frac{N_r\omega(1+\varepsilon)\alpha(\rho_r k_r+\rho_{gr}k_g)+N_g\omega(1+\varepsilon)\alpha(\rho_{rg}k_r+\rho_g k_g)}{N\omega k\alpha} \\ &= \frac{(1+\varepsilon)[N_r(\rho_r k_r+\rho_{gr}k_g)+N_g(\rho_{rg}k_r+\rho_g k_g)]}{Nk}. \end{aligned} \tag{8}$$

The average download cost of ERL-RFC is:

$$\begin{aligned} C_{d-ERL} &= \frac{N_r\omega(1+\varepsilon)k\alpha(\rho_r \frac{N_r}{N}+\rho_{gr}\frac{N_r}{N})+N_g\omega(1+\varepsilon)k\alpha(\rho_{rg}\frac{N_r}{N}+\rho_g\frac{N_r}{N})}{N\omega k\alpha} \\ &= \frac{(1+\varepsilon)[N_r(\rho_r N_r+\rho_{gr}N_g)+N_g(\rho_{rg}N_r+\rho_g N_g)]}{N^2}. \end{aligned} \tag{9}$$

Proof: The total request rate of the red area and the green area is $N_r\omega$, $N_g\omega$ respectively. The amount of data to be transmitted using the D2D download file is $(1+\varepsilon)k\alpha$ bits. For URL, the transmitted data has $(1+\varepsilon)k_r$ bits from the storage node in the red area and $(1+\varepsilon)k_g$ bit data from the storage node in the green area.For ERL, the transmitted data has $\frac{N_r}{N}k$ bits from the storage node in the red area and $\frac{N_g}{N}k$ bit data from the storage node in the green area. Normalize the total download data $N\omega k\alpha$, we get the formula (8) and (9).

Average Repair Cost: Assuming only one lost data is repaired at a time, the average repair cost of URL-RFC is:

$$C_{r-URL} = \frac{n_r\lambda\gamma_r\rho_r + n_g\lambda\gamma_g\rho_g}{n\lambda\alpha} = \frac{k_r c\log(k_r)\rho_r + k_g c\log(k_g)\rho_g}{k}. \tag{10}$$

Proof: The number of storage nodes leaving the red and green areas in unit time is $n_r\lambda$, $n_g\lambda$ respectively, and the bandwidth cost of repairing a lost data is $\gamma_r\rho_r$, $\gamma_g\rho_g$ respectively. Normalize the total repair data $n\lambda\alpha$ to get the formula (10).

The average repair cost for ERL-RFC is:

$$\begin{aligned} C_{r-ERL} &= \frac{n_r\lambda\gamma(\frac{n_r-1}{n-1}\rho_r+\frac{n_g}{n-1}\rho_{gr})+n_g\lambda\gamma(\frac{n_g}{n-1}\rho_{rg}+\frac{n_g-1}{n-1}\rho_g)}{n\lambda\alpha} \\ &= \frac{d(k)[\rho_r N_r(N_r k-RN)+2\rho_{gr}N_r N_g k+\rho_g N_g(N_g k-RN)]}{N^2(k-R)}. \end{aligned} \tag{11}$$

where, $\gamma = d(k)\beta = d(k)\alpha = c\log(k)\cdot\frac{M}{k}$, $R = \frac{k}{n}$.

Proof: The amounts of data to be transmitted using D2D repair both in the two area is γ bits. When the storage node in the red area leaves the system, it needs $\frac{n_r-1}{n-1}$ parts of data from the red area with a cost of ρ_r and $\frac{n_g}{n-1}$ parts of data from the green area, with a cost of ρ_{gr}. When the storage node in the green area leaves the system, it is similar to the red. Normalizetotal repair data $n\lambda\alpha$ to get the formula (11).

Total Average Cost: The total average cost is defined as the sum of the average download cost and the average repair cost.

The total cost of URL-RFC is:

$$C_{URL} = C_{d-URL} + C_{r-URL}, \tag{12}$$

$$C_{ERL} = C_{d-ERL} + C_{r-ERL}. \tag{13}$$

5 Simulation and Results

This section compares the total cost of URL-RFC and ERL-RFC under different conditions. In the simulation, the common parameter values are given as follows. Code rate is $R = \frac{1}{2}$, and input symbols number is $k = 100$. The constant $c = 4$ in the encoding symbol degree of RFC, and the decoding overhead $\varepsilon = 0.1$.

In Fig. 5, we select three different communication power cost ratios $\rho_{gr} : \rho_r : \rho_g$, which are 3:2:1, 5:3:1 and 10:5:1. We can see that, the communication cost of URL-RFC is less than ERL-RFC at three different ratios. As the ratio increases, the gain difference of the URL-RFC compared to the ERL-RFC also increases.

In Fig. 6, we change the ratio of the number of input symbols allocated in the two areas under the total input symbols fixed, and consider the influence of the number of nodes in the two areas on the communication cost. When the number of encoded symbols in the green area increases and the coded symbols in the red area decrease correspondingly, it can be seen that the overall communication cost is continuously decreasing. This is because k_g increases the k_r reduction, which reduces the locality of the encoded symbols in the red area, so that the repair cost of the red area nodes is reduced. Since we calculate the average cost of transmitting unit bits, the change in the number of nodes in the two areas

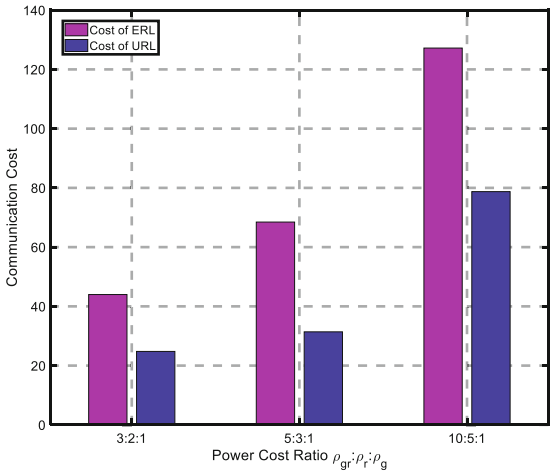

Fig. 5. Comparison of communication cost between URL-RFC and ERL-RFC under different power cost.

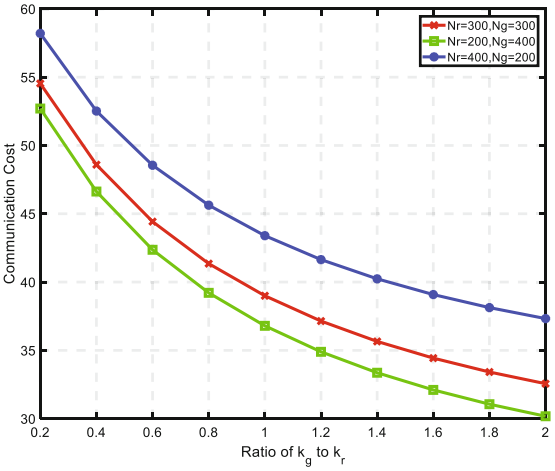

Fig. 6. Comparison of communication cost of URL-RFC under different ratio of $k_g : k_r$. (Color figure online)

has little effect on the result. However, it can be seen that when the number of nodes in the red area is large, the communication cost is larger than the other two cases.

In summary, URL-RFC can reduce the overall communication cost of the heterogeneous D2D distributed storage network. When using URL-RFC, in order to reduce the communication cost of the network, we should allocate input symbols as many as possible to the green area if the number of nodes in the green area is sufficient.

6 Conclusions

In this paper, we introduce an unequal repairing locality code based on repairable fountain codes (URL-RFC) used in heterogeneous D2D data storage systems. It can provide unequal data protection for different nodes transmission capacity in different areas. The lower locality of URL-RFC can reduce the repair cost in D2D storage system, and tradeoff different node capabilities of transmitting. We firstly give the heterogeneous D2D storage network model, and analysis the communication cost for data download and repair. Then, the construction method of URL-RFC is given based on generated matrix. Simulation results show that, URL-RFC significant outperforms conventional distributed codes of communication cost in heterogeneous D2D storage system. And we show that assigning symbols to green areas as much as possible will have lower communication cost.

Acknowledgments. This work was supported in part by the National Natural Sciences Foundation of China under Grant 61701136, Grant 61501140 and Grant 61525103, and China Postdoctoral Science Foundation Grant 2018M630357, and Shenzhen Basic Research Program under Grant JCY2017081114233370 and Grant ZDSYS20170728090330586.

References

1. Wang, L., Wu, H., Han, Z.: Wireless distributed storage in socially enabled D2D communications. IEEE Access **4**, 1971–1984 (2017)
2. Pääkkönen, J., Hollanti, C., Tirkkonen, O.: Device-to-device data storage for mobile cellular systems, pp. 671–676 (2013)
3. Pääkkönen, J., Hollanti, C., Tirkkonen, O.: Device-to-device data storage with regenerating codes. In: Jonsson, M., Vinel, A., Bellalta, B., Tirkkonen, O. (eds.) MACOM 2015. LNCS, vol. 9305, pp. 57–69. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23440-3_5
4. Pedersen, J., Amat, A.G.I., Andriyanova, I., Brännström, F.: Distributed storage in mobile wireless networks with device-to-device communication. IEEE Trans. Commun. **64**(11), 4862–4878 (2016)
5. Pedersen, J., Amat, A.G.I., Andriyanova, I., Brännström, F.: Optimizing MDS coded caching in wireless networks with device-to-device communication (2018)
6. Asteris, M., Dimakis, A.G.: Repairable fountain codes. IEEE J. Sel. Areas Commun. **32**(5), 1037–1047 (2014)
7. Luo, Z., Song, L., Zheng, S., Ling, N.: Raptor codes based unequal protection for compressed video according to packet priority. IEEE Trans. Multimedia **15**(8), 2208–2213 (2013)
8. Chen, Z., Xu, M., Yin, L., Lu, L.: Unequal error protected jpeg 2000 broadcast scheme with progressive fountain codes, pp. 1–5 (2012)
9. Hu, Y., et al.: Unequal failure protection coding technique for distributed cloud storage systems. IEEE Trans. Cloud Comput. **PP**(99), 1 (2017)
10. Miller, S., Childers, D.: Probability and Random Processes, 2nd edn (2012)