Fundamental Journal of Mathematics and Applications, 5 (4) (2022) 240-244
Research Article

Fundamental Journal of Mathematics and Applications

Journal Homepage: www.dergipark.org.tr/en/pub/fujma
ISSN: 2645-8845
doi: https://dx.doi.org/10.33401/fujma.1091736

Error Elimination From Bloom Filters in Computer Networks Represented by Graphs

Gökçe Çaylak Kayaturan

Department of Mathematics, Faculty of Science and Letters, Ordu University, Ordu, Turkey

Article Info

Keywords: *Computer network, Edge labelling, Shortest path, Undirected graph*
2010 AMS: *68R10, 97K20, 97P20*
Received: *22 March 2022*
Accepted: *14 October 2022*
Available online: *27 October 2022*

Abstract

An undirected mathematical graph, $G = (V,E)$ where V is a set of vertices and $E = V \times V$ is the set of edges, can model a computer network. By this consideration we search for solutions to real computer network problems with a theoretical approach. This approach is based on labelling each edge by a subset of a universal set, and then encoding a path as the union of the labels of its edges. We label each vertex $v \in V$ by using a subset of universal set U, then we present a way to encode shortest paths in the graph G by using a way optimizing the data. By mathematical approach, it is provable that the labelling method we introduced eliminates the errors from the shortest paths in the graph. We aim to obtain the results in a more efficient use of network resources and to reduce network traffic. This shows how our theoretical approach works in real world network systems.

1. Introduction

We consider an undirected and unweighted regular graph $G = (V,E)$ where V is the set of vertices and $E = V \times V$ is the set of edges in the graph. This graph may represent a computer network. Therefore, it may be a reasonable approach that a real-time routing scenario in a computer network can be modeled in a mathematical graph. As a realistic model for a computer network we choose a graph denoted by a king's graph. The king's graph is a graph $G = (V,E)$ with a set of vertices V and a set of edges E where $V = \{(i, j)|i \in [0,M], j \in [0,N], M,N \in \mathbb{Z}\}$. The vertices (i, j) and (p,q) are connected in a king's graph by an edge if and only if $i = p$ and $j = q \pm 1$ or $i = p \pm 1$ and $j = q$ or $i = p \pm 1$ and $j = q \pm 1$ (see Figure 1.1).
In literature, some applications of king's graph such as tracking vehicles [1] has been studied by [2, 3]. In order to produce solutions to some network problems encoding the verticies [4] or edges [5] have been suggested in literature.
One of the labelling idea denoted by Bloom filter has been studied by [6]. A Bloom filter is a way to compress the data. Bloom filter has been widely preferred to seek for solutions to network problems [7]-[9]. This is because it saves time and space when querying the element whether in the set or not. Another application of the Bloom filter is to save space in big sized graphs [10]. Using small spaces in some models is an advantage for saving memory or reducing the network traffic [11]. A wide range of research of network applications of Bloom filters has been presented in [12]. Bloom filter is a random data structure, therefore it may produce errors denoted by false positives. These errors can be tolerable in the set, if the probability of false positives is highly small. Therefore, some applications of the Bloom filter aim to reduce the probability of false positives.
Considering routing scenarios in networks the users may face some delivery problems. For network deliveries using shortest paths [13] is an advantage to save time and network resources. However, this approach can cause additional network traffic in practice. In this case, the users may be forced to use any path between two distinct nodes rather than shortest paths. We have introduced encoding methods for shortest paths without false positives in some graphs [14]-[16]. An encoding method for the shortest path in king's graph were considered in [5]. In this paper, we consider the routing scenarios using any path for delivery in a king's graph. We build Bloom filters do not produce false positives and uses less space than the Bloom filter obtain in [5].

Email address and ORCID number: gokcekayaturan@odu.edu.tr, 0000-0001-6823-7686

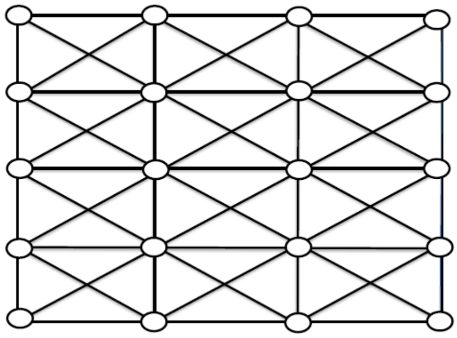

Figure 1.1: A king's graph with computers on each vertex [5].

In this research, we consider to encode any path by using the main idea behind the Bloom filters. This encoding method do not produce false positives. Note that a computer tends to send the message throughout all connections to it, hence the false positives in this model are the adjacent edges to path chosen in advance. The Bloom filter in this model of routing has a role of packet header that is sent with the message between computers. A sender can send messages through any path between two distinct nodes. We assume that the sender chooses a particular delivery path, encodes it as a Bloom filter, and this Bloom filter is sent along together with the message. We introduce a certain encoding method for the paths in this paper for this model to function.

2. Bloom filters for routing models

Bloom filter is a way to represent a subset S with n elements of a universe U [6]. We denote the Bloom filters by β. Each element in the set U is represented by a binary array of length m. The number of the bits 1 in this binary string is k.

The Bloom filter can be obtained by applying the binary operations to the binary strings of the elements in the set S. This structure of the Bloom filter is determined by design of the applications. We use bit-wise OR operation in this research like some other applications of the Bloom filter [17]. Binary OR operation takes the bit 0 and 1, then it produces the bit 1, otherwise it produces the bit 0.

An element x from the set U can be queried whether in the subset S or not by comparing the Bloom filter of the subset S with the binary array of the element x in all bits positions. This property of the Bloom filter provides the users to access the set very quickly.

We may denote the representative binary array of an element by $\beta(x)$ and the Bloom filter of the set S by $\beta(S)$. If $\beta(x) \not\leq \beta(S)$, then it can be concluded that x is definitely not in the set S. However, if $\beta(x) \leq \beta(S)$, then we cannot be certain about the existence of the element x in the set S. Since, the standard implementation of a Bloom filter, the bits 1s are placed in the array of each element randomly and the Bloom filter of the set S is obtained by adding these binary arrays together.

Because of this randomness, one can obtain that $\beta(x) \leq \beta(S)$ for some elements in the universe. Some of these elements may seem like an element of the set S, but they may not belong to the set S. These elements are called false positives. The probability of false positives is obtained after a simple calculation as $(1 - e^{\frac{-kn}{m}})^k$ where m is the length of the Bloom filter, n is the number of elements in the set S and k is the number of bit 1 in the Bloom filter of an element [6].

The probability of false positives can be reduced depending on the number of the bits 1 in the Bloom filter of the set S.Therefore, the optimum number of the bits 1s denoted by k in the Bloom filter is computed by the formula $\lceil ln2 \times \frac{m}{n} \rceil$ which is obtained by taking the derivation of the false positives probability formula [12].

3. A way of edge labelling

Suppose $G = (V,E)$ is a king's graph with a set of vertices V and a set of edges E, and U is a universal set of labels, obviously $V < E$ in the king's graph. Consider a labelling such that $U = V$, and for each $e \in E$, $\beta(e) = \{u,v\}$, where u and v are the endpoints of the edge e and $\beta(e)$ is the label of the edge e. That means each vertex in the label of a path is represented by one bit. Therefore, we may denote this labelling method by a bit-per-vertex labelling.

There are $(M+1)(N+1) = MN+M+N+1$ vertices in total in a king's graph of size of $M \times N$. Therefore, the length of the labels in the graph is $MN+M+N+1$. A path is a sequence of the consecutive distinct edges and an edge is denoted by $e = \{v_i, v_j\}$ where v_i and v_j are the end vertices of the edge e. A path $P = v_0, v_1, ..., v_n$ where v_i is a vertex and $i \in \{0,1,2,...,n\}$ is represented by a label that is obtained by applying bitwise OR operation to the labels of the vertices belonging to the path P. Binary OR operation takes the bits 0 and 1 as inputs and produces a bit 1, if at least one of the input is 1. Otherwise, it produces a bit 0, when all inputs are 0. The number of the bit 1 in the label of a vertex, an edge and a path are $1,2$ and n, which is the number of veritices in the path, respectively.

4. Properties of the shortest paths

The edges in a king's graph has four orientation of compass directions that are north (or south), east (or west), north-east (or south-west) and north-west (or south-east. For instance, vertical and horizontal edges have the orientation of the way of north or equally south and east or equally west, respectively. The orientations of the diagonal edges are north-east (south-west) and north-west (south-east) in a king's graph.

Lemma 4.1. *If a shortest path between the vertices $u = (x_i, y_j)$ and $v = (x_{(i+m)}, y_{(j+n)})$ consists of horizontal and diagonal edges, with at least one horizontal edge, then the first components of the vertices have a sequence of $x_i, x_{(i+1)}, x_{(i+2)}, ..., x_{(i+n)}$. If a shortest path between the vertices $u = (x_i, y_j)$ and $v = (x_{(i+m)}, y_{(j+n)})$ consists of vertical and diagonal edges, with at least one vertical edge, then the second components of the vertices have a sequence of $y_i, y_{(i+1)} y_{(i+2)}, ..., y_{(i+n)}$.*

Proof. Suppose a path P between vertices $u = (x_i, y_j)$ and $v = (x_{(i+m)}, y_{(j+n)})$. The endpoints of the vertical, horizontal and diagonal edges have a form of $\{(x_i, y_j), (x_i, y_{(j+1)})\}$, $\{(x_i, y_j), (x_{(i+1)}, y_j)\}$, and $\{(x_i, y_j), (x_{(i+1)}, y_{(j+1)})\}$, respectively.
There can be found two paths between the vertices (x_i, y_j) and $v = (x_{(i+1), y_(j+1)})$. One path P_1 is the diagonal edge where the sequence of the vertices in the path is $\{(x_i, y_j), (x_{(i+1)}, y_{(j+1)})\}$, and other path P_2 consists of one vertical and one horizontal edge with the sequence of vertices $\{(x_i, y_j), (x_{(i+1)}, y_j), (x_{(i+1)}, y_{(j+1)})\}$ or $\{(x_i, y_j), (x_i, y_{(j+1)}), (x_{(i+1)}, y_{(j+1)})\}$. Therefore, $|P_1| < |P_2|$. This concludes that vertical and horizontal edges do not appear in a shortest path. Besides, if $m > 0$ and $n > 0$ where the path lies between the vertices $u = (x_i, y_j)$ and $v = (x_{(i+m)}, y_{(j+n)})$, then this shortest path contains diagonal edges as many as possible.

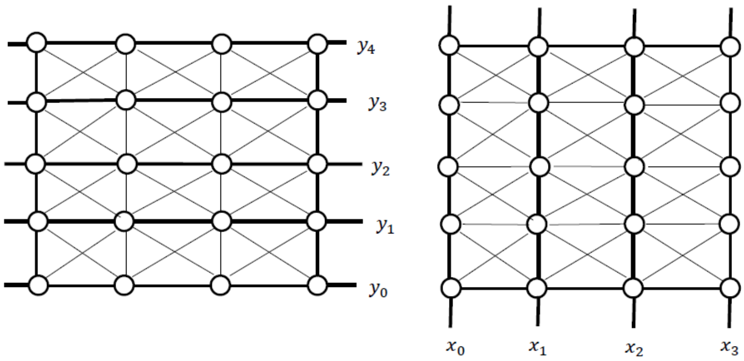

Figure 4.1: Vertical and horizontal lines in the king's graphs

The adjacent vertical edges which have the same first component constitute a vertical line in the graph (see Figure 4.1). The vertical lines in the graph are parallel to the y-coordinate and one can assign each of them with a number of a point from x-coordinate such as $\{x_1, x_2, ..., x_n\}$. Similarly, the adjacent horizontal edges which have the same second component constitute a horizontal line in the graph. The horizontal lines are parallel to the x-coordinate, then the number of the lines are $\{y_1, y_2, ..., y_n\}$. Therefore, each edge in the shortest path consisting of horizontal and diagonal edges takes place between the ordered pair of two consecutive vertical lines and the edges in the shortest path consisting of vertical and diagonal edges takes place between the ordered pair of two consecutive horizontal lines.
Suppose there are two edges from the shortest path consisting of horizontal and diagonal edges between two consecutive vertical lines numbered x_i and x_{i+1} where $i \in \{1, 2, ..., n\}$ and there is one edge from the shortest path between all other consecutive vertical lines. The previous edge of the edges lying between the vertical lines x_i and x_{i+1} has the endpoints between the lines x_{i-1} and x_i . The sequence of the vertical lines containing the endpoints of the these three edges is $x_{i-1}, x_i, x_{i+1}, x_i$. Theoretically, this fragment of the path P contains the edges between the lines x_{i-1} and x_i . Obviously, there can be found a another path between these two lines which is shorter than the path P which also contain another edges between the lines x_i and x_{i+1}. Similarly, we may suppose the shortest path consisting of vertical and diagonal edges contains two edges between two horizontal lines y_i and y_{i+1} and there is one edge from the shortest path between all other consecutive horizontal lines. In a fragment of the path, the endpoints of the adjacent edges belong to the horizontal lines y_{i-1}, y_i, y_{i+1}. There is one edge between the lines y_{i-1} and y_i and two edges between the lines y_i and y_{i+1}. However, there is another path containing the edges between the lines y_{i-1} and y_i and this path is shorter than the path containing two edges between the lines y_i and y_{i+1}.
□

5. Zero false positives zone

We suppose a king's graph represents a network and each node in the graph represents a computer. The shortest paths between distinct nodes are used for the message delivery. The sender computer chooses a shortest path to the receiver and the messages follow this route from the sender to the receiver. The messages are not sent back. Yet, a computer tends to send the message through the computers connected to it. Therefore, the false positives in this model are the adjacent edges to the chosen shortest path.

Theorem 5.1. *If a path P is one of the shortest between the vertices u and v in a king's graph, the label of the path does not produce a false positive.*

Proof. Suppose the edges in a king's graph are labelled by one-bit-vertex labelling. Consider a shortest path P with the sequence of vertices $v_0, v_1, ..., v_n$. The edges in the path P are represented by $e_k = \{v_k, v_{(k+1)}\}$ where $k \in \{0, 1, ..., n\}$. Suppose that there is a false positive $f = \{v_i, v_j\} \in E$.
Since f is adjacent to the shortest path, either $v_i \in \{v_0, v_1, ..., v_n\}$ or $v_j \in \{v_0, v_1, ..., v_n\}$. Suppose $v_i \in \{v_0, v_1, ..., v_n\}$ and $v_i = \{(p,q)\}$ where $p \in [0,M]$ and $q \in [0,N]$ in a $M \times N$ sized king's graph. Therefore, v_j belongs to $V' = \{(p+1,q),(p-1,q),(p,q+1),(p,q-1),(p+1,q+1),(p-1,q-1),(p+1,q-1),(p-1,q+1)\}$. A vertex in a shortest path is connected with two adjacent vertices which belong to the shortest path. Therefore, two adjacent vertices v_{i-1} and v_{i+1} to the vertex v_i also belong to V'. If $v_j = v_{i-1}$ or $v_j = v_{i+1}$, then we can conclude that f is an edge in the shortest path. According to bit-per-vertex labelling each node is represented by a bit in the Bloom filter of the shortest path. Therefore, it is obtained that $\beta(f) \leq \beta(P)$ and f is not a false positive.
Suppose $v_j \in V' - \{v_{i-1}, v_{i+1}\}$. The vertices v_{i-1}, v_i, v_{i+1} can belong to consecutive vertical lines x_{i-1}, x_i, x_{i+1}. Yet, the other vertices in $V' - \{v_{i-1}, v_{i+1}\}$ also belongs to the lines x_{i-1}, x_i, x_{i+1}.
However, if the shortest path consists of horizontal and diagonal edges, then by the Lemma 4.1 the endpoints of the edges in the shortest path belong to the vertical lines and there is one vertex from the shortest path on each vertical line. Since each vertex is represented by one bit in the shortest path, βP recognizes that the other vertices whether on the path or not. Therefore, when $v_j \in V' - \{v_{i-1}, v_{i+1}\}$, then $\beta(f) \nleq \beta(P)$ and f is not a false positive.
Similarly, if the shortest path consists of vertical and diagonal edges, then by the Lemma 4.1 the endpoints of the edges in the shortest path belong to the horizontal lines and there is one vertex from the shortest path on each horizontal line. Each horizontal line contains one vertex from the shortest path by Lemma 4.1. Therefore, when $v_j \in V' - \{v_{i-1}, v_{i+1}\}$, then $\beta(f) \nleq \beta(P)$ and f is not a false positive.
In conclusion, bit-per-vertex labelling method in king's graph do not produce any false positives for the shortest paths. □

6. Practical performance of bit-per-vertex encoding method

The edges in a graph G can be encoded by one bit. Therefore, in a Bloom filter of a shortest path each edge is represented by one bit 1 and each bit has a particular bit position in the Bloom filter. Therefore, this encoding method also do not produce a false positive.
Obviously, the length of the Bloom filter is $|E|$ and the number of the bits 1s k in the Bloom filter is the number of edges in the path P. The number of edges in a king's graph $|G|$, where the size of the graph is $M \times N$, is $4MN + M + N$. By using bit-per-vertex encoding method, the length of the Bloom filter is obtained as $|V| = MN + M + N + 1$. Obviously, in a king's graph $|V| < |E|$. Hence, the space is saved with the parameters used for coding edges in our method for the king's graphs.
Besides, if the standard Bloom filter is used for encoding the edges with the parameters that we have obtained with bit-per-vertex encoding, then we probably obtain false positives. The probability of false positives is [6] by the formula $(1 - e^{\frac{-kn}{m}})^k$ where m is length of Bloom filter, n is the number of edges in a shortest path between two distinct nodes and k is the number of the bits 1s in the Bloom filters of the edges.
In our model, $m = |V|$ and $k = 2$. If we take n as its maximum value. This is $max(M,N)$ where the size of the king's graph is $M \times N$, when the shortest path is one of the the number shortest path lying between one corner to opposite corner of a king's graph. In order to obtain less probability of false positives, we recalculate k by using the formula $\lceil ln2 \times \frac{m}{n} \rceil$, [11]. By this formula, optimum k is obtained with the maximum number of edges in a shortest path and the length of Bloom filters of the edges.

Size of a king's graph	$m = (M+1)^2$	Optimum k	$n = max(M,M)$	Probability of false positives
2×2	9	≈ 3	2	$\approx 0,115$
10×10	121	≈ 8	10	$\approx 0,002$
18×18	361	≈ 14	18	$\approx 0,000065$
25×25	676	≈ 18	25	$\approx 0,0000022$
30×30	961	≈ 22	30	$\approx 2,07090265e-7$

Table 1: The probabilities of false positives are obtained by using parameters from different sizes of graphs

Therefore k is $\lceil ln2 \times \frac{M^2+2M+1}{M} \rceil$ with the parameters of our model where the size of the graph is $M \times M$, the optimum k depends on the size of the king's graph. This results that the more the size of the graph increases, the more the value of the k rises. For example; $k \approx 2$, when the king's graph has a size of 1×1. Obviously, $k > 2$ in the other sizes of the king's graphs. However, real-world network models have bigger sizes than 1×1 sized king's graph. In the encoding method we introduce k is 2 in the Bloom filters of edges in any size of the king's graph. This is another advantage of bit-per-vertex encoding.

We can conclude that if the edges in a king's graph would have been encoded by standard Bloom filter with optimum parameters of our model, then the probability of the false positives would be calculated with $(1-e^{\frac{-kn}{m}})^k$. We list some examples of probabilities of false positives obtained from different sizes of graphs (see Table 1).
Another encoding method introduced in [5] do not produce false positives for shortest paths in king's graphs. The length of the Bloom filter in that study is $12\times(M\times N)$ where the size of the king's graph is $M\times N$. It can be seen that $12\times(M\times N)\leq MN+M+N+1$ when $M=N=22$. Therefore, one may think the method introduced in [5] which offers $12\times(M\times N)$ length Bloom filter saves more space than bit-per-vertex encoding method in the big size of king's graph where $M>22$ and $N\geq 22$. However, in literature the practicable size of a Bloom filter m has been chosen as 256 [13]. Hence, bit-per-vertex coding works for bigger sizes of king's graph models with more advantages than the other encoding methods, if m is chosen from 256 up to 506.

7. Conclusion

In this research we have chosen a graph with $V<E$ that is a realistic network model and we have built labels for the edges in this graph. The labels have reasonably small length. This property of the labels is an advantages, if the network users have a small space to store the data. Additionally, the bit-per-vertex labelling do not produce false positives for shortest paths. This is another advantage for some routing scenarios taking the mathematical graphs as a network model. We show that if a Bloom filter is built with some assumptions, then it is possible to obtain a model using less space without false positives. The encoding method and routing model we have introduced in this paper can work, when the graph is regular and has an overall shape of a king's graph. For the future work, we aim to generalize this encoding idea to arbitrary graphs. The shape of graph in this paper has been chosen specifically, it is regular and undirected. This graph has its specific geometric structure. Therefore, the coding structure can be changed for other types of graphs. Also, we may extend the work in neutrosophic environment for future studies.

Funding

There is no funding for this work.

Availability of data and materials

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author's contributions

All authors contributed equally to the writing of this paper. All authors read and approved the final manuscript.

References

[1] M. X. Punithan, S. Seo, *King's graph-based neighbor-vehicle mapping framework*, IEEE trans Intell Transp Syst, **14** (2013), 1313–1330.
[2] O. Favaron, G. H. Fricke, D. Pritikin, J. Puech, *Irredundance and domination in kings graphs*, Discrete Math. Elsevier, **262** (2003), 131–147 .
[3] E. J. Ionascu, D. Pritikin, S. E. Wright, *k-Dependence and domination in kings graphs*, Amer. Math. Monthly, **115** (2008), 820–836.
[4] R. Dantas, F. Havet, R. M. Sampaio, *Minimum density of identifying codes of king grids*, Discrete Math., **341** (2018), 2708–2719.
[5] G. C. Kayaturan, *Representing Shortest Paths in Graphs Using Bloom Filters without False Positives and Applications to Routing in Computer Networks*, Ph.D thesis, Univeristy of Essex, UK, 2018.
[6] B. H. Bloom, *Space/time trade-offs in hash coding with allowable errors*, Commun. ACM, **13** (1970), 422–426.
[7] Y. Lu, B. Prabhakar, F. Bonomi, *Perfect hashing for network applications*, IEEE Int. Symp. Inf. Theory - Proc., (2006), 2774–2778.
[8] C. E. Rothenberg, C. Macapuna, B. Alberto, M. F. Magalhães, F. L. Verdi, A. Wiesmaier, *In-packet Bloom filters: Design and networking applications*, Comput. Netw., Elsevier, **55** (2011), 1364–1378.
[9] X. Yang, A. Vernitski, L. Carrea, *An approximate dynamic programming approach for improving accuracy of lossy data compression by Bloom filters*, Eur. J. Oper. Res. **252** (2016), 985–994.
[10] A. Saha, N. Sengupta, M. Ramanath, *Reachability in large graphs using bloom filters*, Proceedings - 2019 IEEE Trans Knowl Data Eng, ICDEW 2019, (2019), 217–224.
[11] M. Mitzenmacher, *Compressed bloom filters*, IEEE ACM Trans Netw., **10** (2002), 604–612.
[12] A. Broder, M. Mitzenmacher, *Network applications of bloom filters: A survey*, Internet Math., Taylor & Francis , **1** (2004), 485–509.
[13] L. Carrea, A. Vernitski, M. Reed, *Optimized hash for network path encoding with minimized false positives*, Comput. Netw., **58** (2014), 180–191.
[14] G. C. Kayaturan, A. Vernitski, *A Way of eliminating errors when using Bloom filters for routing in computer networks*, Comput. Electr. Eng. (CEEC), 2016 8th, (2016), 95–100.
[15] G. C. Kayaturan, A. Vernitski, *Routing in hexagonal computer networks: How to present paths by Bloom filters without false positives*, Netw., ICN 2016. , (2016), 52–57.
[16] G. C. Kayaturan, A. Vernitski, *Encoding shortest paths in triangular grids for delivery without errors*, Proceedings - ICFNDS, (2017), 7.
[17] S. Tarkoma, C. E. Rothenberg, E. Lagerspetz, *Theory and practice of bloom filters for distributed systems*, IEEE Commun. Surv. Tutor., **14**, (2012), 131–155.