Optimal Subgraph Detection to Identify Opportunities for the Application of Deterministic Network Coding

M.J. Grobler *, A.S.J Helberg* and S.E. Terblanche†
*School of Electrical, Electronic and Computer Engineering
North-West University (Potchefstroom)
Email: leenta.grobler@nwu.ac.za, albert.helberg@nwu.ac.za
†Centre for Business Mathematics and Informatics, North-West University (Potchefstroom)
Email: fanie.terblanche@nwu.ac.za

***Abstract*— We present a new Subgraph Detection Model (SDM), based on a well known combinatorial optimization problem. The SDM method is used for the identification of nodes in a network, where network coding can be implemented. This method improves on work done in [1], by identifying a broader class of topology patterns. Our proposed method provides higher scalability when applied to networks with high link densities. The improvement using SDM allows opportunistic, deterministic network coding to be implemented on larger networks with the accompanying benefits of higher throughput and reduced network load.**

I. Introduction

Techniques to better utilize network resources are constantly being developed. Recent advances in Information Theory aiming to do exactly that, have lead to a technique called Network Coding [2]. This technique enables a network node to encode two or more packets by forming linear combinations of the packets it receives by means of an XOR-operation [2]. The size of the combined packet is therefore the same as the largest of the two original packets. The receiver node can then decode the original messages once it receives enough linear independent packets. By forwarding these combined packets fewer transmissions are needed to transfer the same amount of information, therefore, reducing the load on the network. Network coding offers a number of advantages over traditional routing, including [3]:

- Adaptability
- Limited inherent security
- Robustness
- Throughput

There are two main types of Network Coding: Deterministic Network Coding, originally presented by Ahlswede *et al* [2], and Random Network Coding presented by Ho *et al* [4]. Many researchers have tried to make Network Coding more practical [5], but most of their efforts went into Random Network Coding. Deterministic Network Coding however, holds some advantages over Random Network coding, including *less decoding delay*: Because a node will not have the uncertainty associated with waiting until sufficient independent linear messages were received, and *less overhead* : If decoding nodes have prior knowledge on how a packet was encoded, no coding vector has to be sent along with the message.

The problem with a practical implementation of Deterministic Network Coding is that it can only be done if we know exactly what the network looks like, which is often not the case in wireless mesh networks (WMNs). Grobler in [1] proposed a method to identify opportunities for the implementation of Deterministic Network Coding using the connection matrix of the network. The aim of this method is to search for topologies within a larger network, known to be suitable for Network Coding - three such topologies have been identified. Since we know how network coding should be implemented in these known topologies, we can opportunistically implement encoding and decoding at the applicable nodes within a network as these topologies present themselves within larger networks. The method relies on the concatenation of paths and performs a brute force search for a specific pattern. The method is briefly described in Section II.

In order to be more efficient in identifying opportunities for the implementation of Deterministic Network Coding an optimization based approach is explored in this paper. That is, instead of generating all instances of a specific topology pattern, the most desirable instance of a prescribed topology is identified based on some objective criterion. For instance, it might be that within a large wireless mesh network several suitable topologies exist, but only one topology satisfies a specific distance or bandwidth requirement. The optimization based approach has the added advantage of also being able to identify a larger class of topology patterns that will allow the implementation of Deterministic Network Coding.

In Section II, an overview of Network Coding is provided as well as a description of the method suggested in [1]. In addition, an overview of the combinatorial optimization problem, the Traveling Salesman Problem, will be given, which serves as a reference point for our new approach to be presented in Section III, the Subgraph Detection Model. A discussion on Computational Complexity and Optimality with

respect to the newly presented model is given in Section IV, followed by computational results in Section V for the purpose of evaluating the new approach. We conclude the paper with a summary in Section VI.

II. Background

In this section, we will provide some background on the problem and the initial model used to address it as described in [1] by Grobler. Subsequently, an overview of a well known optimization problem will be presented for the purpose of laying the groundwork needed in presenting our new approach.

A. Indentifying Opportunities for the Implementation of Network Coding

Grobler *et al* in [6] found that there was no known method to determine where in a WMN, or exactly how at an identified location, Deterministic Network Coding could be implemented. They found that opportunities for the implementation of Network Coding can be found by looking for known Network Coding topologies within larger networks.The unique characteristics of WMNs create multiple opportunities for opportunistic implementation of deterministic Network Coding.

In [1] Grobler examines the connection matrix of a network, searches for sub-matrices that represent known network coding topologies and identifies opportunities to implement Network Coding. The sub-matrices indicate which nodes should change in their functionality and which nodes should have information on the network topology. They reported a number of benefits which could arise from the implementation of such a search, including:

- An improvement in the total throughput (better utilization of the network's capacity).
- Lower occupation of the total network.
- Improved Quality of Service (QoS), because of a lower delay in the network.

They proposed five steps to implement Determanistic Network Coding in WMNs:

1) Select a Network Coding topology of which the gain and capacity is known.
2) Derive the connection matrix of the larger network using a suitable distance vector routing algorithm;
3) Search the larger network adjacency matrix for the known topology structure.
4) Implement Network Coding at the appropriate nodes.
5) Re-iterate steps (3) and (4) after each routing update.

They identified the Butterfly, Bowtie and Extended Butterfly topology, as can be seen in Figure 1. The Butterfly topology used by Ahlswede *et al* was used to illustrate how the concept of Network Coding could enable a network to reach its maximum calculated capacity [2]. The Bowtie and Extended Butterfly topologies were suggested in [1] as a simplification and a generalisation of the Butterfly topology.

To describe how the nodes in a network are connected to each other, we can construct the adjacency matrix $A = (a_{ij})$ an (n x n) matrix, where n is the number of nodes in the network, with $a_{ij} \in 1, 0$. Both the rows and columns represent the nodes in the network and the n'th row and the n'th column represent the same node ($a_{1n} = a_{n1} = a_n$) . A connection of one node to another is indicated by an 1 in the matrix and to indicate that a node is not connected (or within transmission range of another node) a 0 is placed in the matrix.

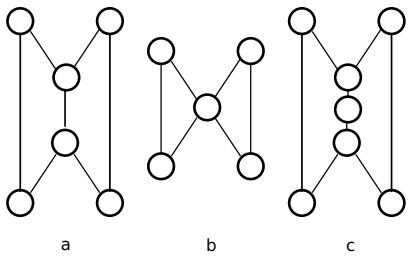

Fig. 1. Topologies: a) Butterfly, b) Bowtie and c) Extended Butterfly

In a network, and therefore in matrix A, each node is at least connected to itself and, therefore, the main diagonal of the matrix will contain only 1's . If we combine these characteristics we can say:

$$A_{n\times n} = (a_{ij}) \text{ where } a_{ij} \in 1, 0 \tag{1}$$

If we were to subtract the identity matrix I from (1) we would be left with $B_{n\times n}$ - the true representation of how the nodes are connected to each other:

$$B_{n\times n} = (b_{ij}) \text{ where } a_{ij} \in 1, 0 \tag{2}$$

We then search in every row of the $B_{n\times n}$ matrix, for all the 1s present. Each time a 1 is found, a 1-hop route exists, and is stored in an ($r \times 2$) matrix $O_{r\times 2}$, where r is the number of 1-hop routes that exist, such that:

$$O_{r\times 2} = [o_{r1} o_{r2}] \tag{3}$$

$$\text{where } \begin{matrix} o_{r1} = i \,\forall\, b_{ij} = 1 \\ o_{r2} = j \,\forall\, b_{ij} = 1 \end{matrix} \text{ , when } 1 \leq r \leq (n^2 - n^2) \text{and} i < j$$

Once we have this matrix of *1-hop* paths, we use each row in the matrix to concatenate *1-hop* paths that share common hops. These concatenated paths can then be used to search for a known Network Coding topology.

To search the larger matrix for the presence of these known topologies they considered using iterative looping or cross-correlation, as presented in [6], but settled on the concatenation model as described in the following section.

B. Concatenation Model (CM)

In this method, the Butterfly can be found using the *1-hop* and *3-hop* paths and the Bowtie can be found using the *1-hop* and *2-hop* paths, while the Extended Butterfly is found using the *1-hop* and *4-hop* paths.

Similar to $O_{r\times 2}$ containing the *1-hop* paths, there are also vectors $T_{s\times 3}$, $D_{t\times 4}$ and $W_{y\times 5}$ containing the *2-hop 3-hop* and *4-hop* paths.

Once all the needed paths are known, we can start searching for the known topology. To find a Butterfly, we examine $O_{r\times 2}$ and $D_{t\times 4}$ together and store all the *3-hop* paths that share start- and end-nodes with *1-hop* paths in a vector:

$$H_{c\times 4} = \begin{bmatrix} d_{u1} & d_{u2} & d_{u3} & d_{u4} \end{bmatrix} \tag{4}$$

$$\text{if } (o_{l1} = d_{u1}), (o_{l2} = d_{u4}) \text{ and } u < l \leq |O_{r\times 2}|.$$

We then search within (4) (the potential Butterfly's wings) to see whether there are rows in it that share two common middle entries so that:

$$(d_{u2}d_{u3}) = (d_{p2}d_{p3}) \text{ and } u \leq p \leq |H_{c\times 4}| \tag{5}$$

This will confirm a shared link between the wings, and therefore verify the presence of a Butterfly topology. Similar equations exist to find the common node shared by the wings of the Bowtie or the common two links shared by the wings of the Extended Butterfly.

A further check was implemented to ensure that there were no shared links between the discovered topologies in order to allow disjoint topologies to co-exist. The original work by Grobler [1], identified **all** the possible disjoint known topologies in a network which could be utilized to implement network coding. This method produced satisfactory results to show that the principle of looking for known topologies within larger networks is viable, and also that the these known topologies occurred often enough in random networks to warrant further investigation.

To be able to compare the above method to the newly proposed model, outlined in Section III, the above method was adapted by adding weights (representing link state, bandwidth or distance) to the adjacency matrix and deriving its sparse matrix. This enabled us to identify a single optimum subgraph from all available subnets where network coding could be implemented. For the purpose of this paper we assume that the weights represent the distance between nodes, and therefore, the known topology with the lowest link values are given priority as a suitable opportunity for the implementation of Network Coding, over those with higher link weights.

In the following section a well known optimization problem is discussed that provides a reference point for the optimization model we propose that will allow us to identify an optimal topology from a given larger network.

C. Travelling Salesman Problem (TSP)

The TSP is a well known combinatorial optimization problem used by many practitioners in theoretical computer science as a benchmark for testing computational efficiency of their algorithms (see e.g. [7]). The basic problem at hand is to find the shortest route among a list of cities such that each city is visited only once.

It suffices at this point to provide a short overview of formulating the TSP problem. Let $G = (V, E)$ be a complete graph with undirected edges. The set $\delta(v) = \{I \subseteq E\}$ is used to represent the incident edges $e \in I$ of the node $v \in V$. Let $W \subset V$ be a subset of V with each edge e within the set of edges $E(W)$ having both end-nodes in W. The graph $G = (V, E)$ denotes all potential routes that might be traversed by the salesman. In order to select the best route, we define $c \in \mathbb{R}^{|E|}$ as the cost vector with each component c_e associated with an edge $e \in E$. The cost vector is generic and may represent some optimisation criteria such as distance, or traveling time. In order to describe an optimal solution, we need to define the variables $y_e \in \{0, 1\}$, for all $e \in E$, that will indicate whether an edge $e \in E$ is part of the optimal route or not. Additionally, we introduce the variables $x_v \in \{0, 1\}$ that will indicate whether a node (city) $v \in V$ will be traversed as part of the optimal route or not. It should be noted that the variables x_v, $v \in V$ are redundant for the TSP, but are included for the purpose of clarifying concepts in the discussion of subsequent models.

The problem formulation for the TSP will be presented next, after which a clarification will be provided on the meaning of the different constraints in the model.

$$\min \sum_{e\in E} c_e y_e$$

$$\sum_{e\in\delta(v)} y_e - 2x_v = 0 \quad \forall\ v \in V \tag{6}$$

$$\sum_{e\in E(W)} y_e \leq |W| - 1 \quad \forall\ W \subset V \tag{7}$$

The constraints (6) will ensure that for any node there are exactly two incident edges that will be part of the solution. That is, for a node v there should exist two edges $e1 \in E$ and $e2 \in E$, such that $y_{e1} = 1$ and $y_{e2} = 1$. Evidently, for this to be true, the variables x_v should also take on a value of 1 in the solution. The constraints (7) will prevent the solution from including sub-tours. That is, for any subset $W \subset V$, with $3 \leq |W| \leq |V| - 1$, the solution shall have no more than $|W| - 1$ number of edges connecting nodes that reside in W.

III. Subgraph Detection Model (SDM)

The objective of the SDM is to identify an optimal subgraph from a given network based on some optimization criterion. By close inspection, the problem reduces to solving interlinked subproblems that have a familiar structure. For instance, in the case of the Butterfly subgraph, the problem is analogue to solving two Traveling Salesman Problems (TSP) that are interlinked. The significance of being able to decompose our problem into familiar subproblems, is that we might be in a situation where we could leverage on work already done on improving algorithms for solving the TSP.

It should be noted, however, that for the purpose of this paper, only an initial investigation is undertaken to establish how we could model subgraph detection without looking at algorithmic improvements. In order to obtain solutions we do make use of a commercially available solver (see Section V).

Recall that for the TSP problem we established the notation y_e for all $e \in E$ and x_v for all $v \in V$ to indicate

whether an edge e and a node v will be in the final solution respectively. Now, in order to complete the formulation of the SDM, we need to extend the notation and introduce some auxiliary variables. As stated previously, the SDM problem is analogue to solving two TSP problems that are interlinked, which support the introduction of an index set $R = \{1, 2\}$ representing the two routes of the two TSP problems. That is, we define $y_{er} \in \{0, 1\}$ to indicate whether an edge e will be in the final solution as part of route $r \in R$. Similarly we define $x_{vr} \in \{0, 1\}$ to indicate whether a node v will be in the final solution as part of route $r \in R$. In order to connect the two TSP routes, we define the variables $g_e \in \{0, 1\}$, for all $e \in E$, and $f_v \in \{0, 1\}$ for all $v \in V$. The variable g_e will take on a value of one if the edge e is common to the two routes $R = \{1, 2\}$, and the variable f_v will take on the value of one if the node v is common to the two routes $R = \{1, 2\}$.

A parameterized model is provided for the optimal detection of either the Butterfly, the Bowtie or Extended-Butterfly subgraphs, depending on the relevant parameters. That is, for the model below, a Butterfly subgraph is obtained by setting the parameters $\alpha = 4$, $\beta = 1$, and $\gamma = 2$, and a Bowtie subgraph is obtained by setting $\alpha = 3$, $\beta = 0$, and $\gamma = 1$. In order to obtain an Extended-Butterfly, the parameters $\alpha = 5$, $\beta = 2$, and $\gamma = 3$ are used. The meaning of the said parameters will become clear in the description of the constraints that will follow the presentation of the model. The optimal subgraph is detected from the given graph $G = (V, E)$ by solving the problem SDM(α, β, γ):

$$\min \sum_{e \in E} \sum_{r \in R} c_e y_{er}$$

$$\sum_{e \in \delta(v)} y_{er} - 2x_{vr} = 0 \quad \forall\ v \in V,\ \forall r \in R \tag{8}$$

$$\sum_{e \in E(W)} y_{er} \leq |W| - 1 \quad \forall\ W \subset V,\ \forall r \in R \tag{9}$$

$$\sum_{e \in E} y_{er} = \alpha \quad \forall\ r \in R \tag{10}$$

$$\sum_{r \in R} y_{er} - g_e \leq 1 \quad \forall\ e \in E \tag{11}$$

$$-\sum_{r \in R} y_{er} - 2g_e \leq 0 \quad \forall\ e \in E \tag{12}$$

$$\sum_{e \in E} g_e = \beta \quad \forall\ e \in E \tag{13}$$

$$\sum_{r \in R} x_{vr} - f_v \leq 1 \quad \forall\ v \in V \tag{14}$$

$$-\sum_{r \in R} x_{vr} - 2f_v \leq 0 \quad \forall\ v \in V \tag{15}$$

$$\sum_{v \in V} f_v = \gamma \quad \forall\ v \in V \tag{16}$$

The first two constraint sets, (8) and (9), relate to the connectivity and subtour elimination constraints of the TSP, respectively, as presented in the preceding section. The only difference is that now we have an additional index $r \in R$ allowing the model to solve as a subproblem two independent TSP problems. The constraints (10) impose a cardinality restriction on the number of edges within a TSP route, as provided by the parameter α. The implication of these cardinality constraints is that we only have to consider subsets W for the subtour elimination constraints for which $3 \leq |W| \leq \alpha - 1$. The constraint sets (11), (12) and (13) are responsible for facilitating the joining of the two TSP routes with respect to one or more common edges, based on the parameter β. Similarly, the constraint sets (14), (15) and (16) are responsible for facilitating the joining of the two TSP routes with respect to one or more common nodes, based on the parameter γ.

IV. Computational Complexity and Optimality

The proposed model SDM(α, β, γ) as presented in the preceding section is called a linear integer programming problem [8]. It is an optimization problem that comprises a linear objective function with linear constraints, but with the added complexity that the variables may only take on the binary values zero and one. For that reason then is this problem also referred to as a combinatorial optimization problem.

There are in principle two approaches in obtaining solutions for combinatorial optimization problems. Firstly, a heuristic approach could be followed, e.g. genetic algorithms, tabu-search techniques etc. Or secondly, an exact solution approach could be employed, e.g. branch-and-bound. The main difference between the two approaches has to do with the ability to validate optimality. In many cases heuristic approaches are fast in generating feasible solutions, but are unable to indicate whether a given solution is optimal, or how far the solution is from optimal. On the other hand, exact approaches may be computationally expensive, but they have the advantage of being able to quantify optimality. That is, for a given solution an exact approach is able to verify optimality, and if a solution is not optimal, it is able to provide a quantification of how far the solution is from optimal.

In this paper an exact solution approach, the branch-and-bound algorithm, was used for solving SDM(α, β, γ). Details on the solver applied for generating solutions can be found in the following section.

V. Computational Results

For the purpose of measuring the improvements obtained in using the proposed model SDM(α, β, γ), it was necessary to compare the results to the Concatenation Model (CM) as presented in Section II-B. The CM approach was implemented in MATLAB whereas the SDM(α, β, γ) was solved using the commercial solver CPLEX (see [9]). Some code was written in C# for generating random networks and for translating SDM(α, β, γ) into a model formulation and passing it to the CPLEX solver.

The random networks used for empirical tests comprised three sets of data with varying link densities, where each set consisted of problem instances having between 10 and 85 nodes, with 5 node increments. That is, for the first data set, the instances $10, 15, 20, \ldots, 85$ were created with a 10% link

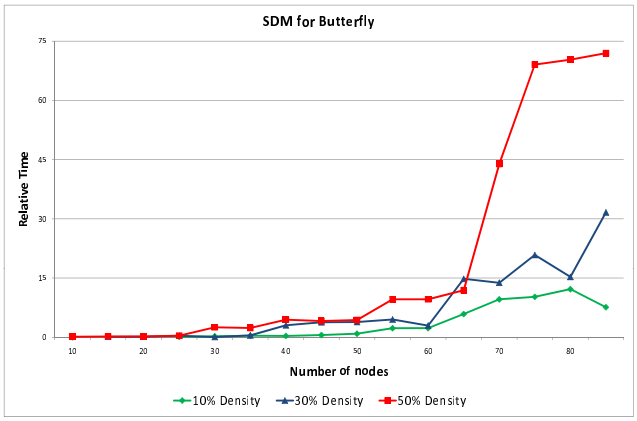

Fig. 2. Execution times for obtaining optimal Butterfly topologies

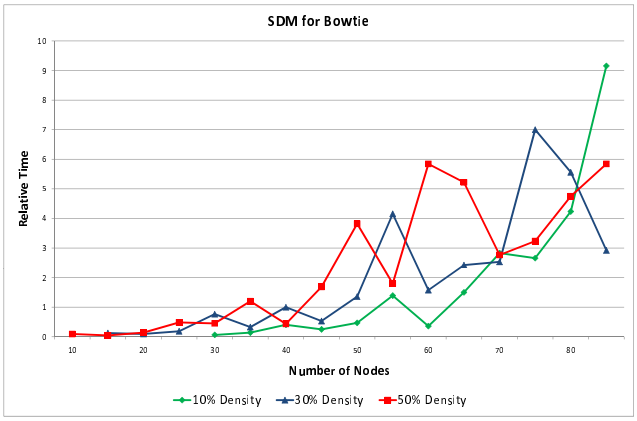

Fig. 3. Execution times for obtaining optimal Bowtie topologies

density, for the second data set, the instances $10, 15, 20, \ldots, 85$ were created with a 30% link density, and finally for the third data set, the instances $10, 15, 20, \ldots, 85$ were created with a 50% link density.

The data sets were subsequently used as input to both the CM and SDM(α, β, γ) in order to search for the "best" Bowtie or Butterfly topology within the generated networks respectively.

The CM, while efficient, is only scalable in sparsely connected networks (approximately 10% link density). In networks where the link density is around 50%, the computing time jumped from 1 millisecond for 15 nodes to 60 seconds for 20 nodes, after which the computer had insufficient memory. This made the CM method impractical since 60 seconds is considered to be very unfavorable for node decision making within network applications. This observation was evident for both the Bowtie and Butterfly topologies.

Results generated by applying the SDM(α, β, γ) appeared to be more favorable. Graphs of the computing times for both the Butterly and Bowtie topologies are presented in Figures 2 and 3 respectively. For the Butterfly topology it is clear that the SDM(α, β, γ) is scalable in computing optimal topologies of up to 50 nodes for networks with 10%, 30% and 50% link densities respectively. Thereafter, it becomes much more difficult to obtain optimal solutions within reasonable time. For instance, with a link density of 50%, the SDM(α, β, γ) finds it difficult to maintain scalability for networks with more than 50 nodes. This can be attributed to the fact that when we consider networks with a high link density, more opportunities exist for feasible topologies making the search space bigger. On the other hand, with low link density networks, less opportunities exist, and therefore, the search space is smaller. It should be noted, however, that there is a higher probability that no feasible solution exist when considering low link density networks compared to high link density networks.

For the Bowtie problem instances, Figure 3, the SDM(α, β, γ) succeeded in identifying optimal topologies within reasonable time for all the problem instances. It is, however, unclear from the graph whether different link densities had a significant influence on scalability. It might be that for problem instances having more than 85 nodes, a clearer trend could be observed. Not withstanding, computing times for the Bowtie problem instances appear to be very favorable.

VI. Summary and Conclusion

The main contribution of this paper is the development of a new approach for identifying suitable topologies to be used in deterministic network coding. This paper is an extension of the work done by Grobler [1], with the added advantage that with the new proposed model we are able to find an optimal topology from a given network based on predefined link weights. These weights may represent available bandwidth, distance or some combination of several metrics. Furthermore, the new model is able to identify a broader class of topology patterns in addition to the Butterfly and Bowtie topologies.

The scope of this paper is limited to the formulation of the problem as a mixed integer programming problem and solving it using a commercial available solver. Future research could include more algorithmic oriented work for reducing computing times and improving scalability.

References

[1] M.J. Grobler, "Using topological information in opportunistic network coding," M.eng dissertation, North-West University, South Africa, 2008.
[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, 2000.
[3] Christina Fragouli, Jean-Yves Le Boudec, and Jörg Widmer, "Network coding: an instant primer," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 63–68, 2006.
[4] T. Ho, R. Koetter, M. Mdard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *In Proceedings of the IEEE International Symposium on Information Theory*, 2003.
[5] Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Médard, and Jon Crowcroft, "Xors in the air: practical wireless network coding," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 4, pp. 243–254, 2006.
[6] M.J. Grobler, A.S.J. Helberg, H.Marais, and C.C. Naudé, "An application of deterministic network coding in MANETs," in *Proc. of the Southern African Networks and Applications Conference (SATNAC)*, 2008.
[7] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, *The traveling salesman problem*, John Wiley and Sons, 1985.
[8] L.A. Wolsey, *Integer Programming*, John Wiley & Sons, 1998.
[9] "Cplex 10.1 reference manual," http://www.cplex.com.

Leenta Grobler received her B.Eng degree in Computer and Electronic Engineering in 2006 from the North-west University and her M.Eng degree in Computer Engineering in 2008 from the same institution. She is currently working as a lecturer at the North-west University while pursuing her PhD degree in the TeleNet Communications Research Group. Her research interests include Information Theory, Engineering Education and Research Management.

Albert Helberg is a professor in the School for Electrical, Electronic and Computer Engineering at the North-west University in Potchefstroom, South-Africa. He received a B.Eng, M.Eng (cum laude) and D.Eng in Electrical and Electronic Engineering from the Rand Afrikaans University. His research interests include Information Theory, Coding Techniques for Communication Networks and Software Defined Radio.

Fanie Terblanche received the B.Sc, B.Sc Hons, M.Sc (cum laude) and Ph.D in Computer Science all from the North-West University (Potchefstroom), South Africa. He is currently employed as a senior lecturer in the Centre for Business Mathematics and Informatics at the North-West University and his main research interests include Combinatorial and Network Optimization.