SOLVING THE CLUSTERING PROBLEM IN MANETS USING SAT & ILP TECHNIQUES

Syed Zahidi, Fadi Aloul, Assim Sagahyroon
American University of Sharjah, UAE

Wassim El-Hajj
American University of Beirut, Lebanon

ABSTRACT

Recently there have been several improvements in the performance of Integer Linear Programming (ILP) and Boolean Satisfiability (SAT) solvers. These improvements have encouraged the modeling of complex engineering problems as ILP problems. One such problem is the *Clustering Problem* in Mobile Ad-Hoc Networks (MANETs). The Clustering Problem in MANETs consists of selecting the most suitable nodes of a given MANET topology as clusterheads (CH), and ensuring that regular nodes are connected to clusterheads such that the lifetime of the network is maximized. This paper focuses on developing an improved ILP formulation for the Clustering Problem, and assessing the feasibility of using such a formulation, together with state-of-the art Generic ILP and SAT solvers, in a real-life practical environment.

Index Terms — Integer Linear Programming, Boolean Satisfiability, Mobile Ad-Hoc Networks, Clustering Problem, Optimization.

1. INTRODUCTION

Over the past decade, Integer Linear Programming (ILP) and Boolean Satisfiability (SAT) solvers have improved significantly, through the introduction of new intelligent algorithms that allowed the solvers to handle a wider range of challenging Engineering problems. There are several real-life applications which can be solved using ILP and SAT. While generic-based ILP Solvers have been applied to solving ILP models of several real-life optimization problems, relatively few similar attempts have been made using SAT solvers. One such problem is the *clustering problem* in Mobile Ad-Hoc Networks (MANETs). MANETs are used in a wide-range of applications such as battlefield communication, law enforcement operations, and disaster recovery [1]. The proposed solution to the scalability issue in *flat* MANET networks is the concept of *clustering*. Clustering involves the creation of a hierarchical network where the network is divided into clusters, with certain nodes in each cluster being chosen to be *clusterheads*. The process of selecting which nodes would be best suited to be clusterheads and which regular nodes should be assigned (connected) to which clusterhead is known as the *clustering problem*. The clustering problem can be modeled as an optimization problem in ILP. The primary objective of this paper is to present an improved ILP formulation of the clustering problem in MANETs. Additionally, this paper presents an evaluation of the performance of the state-of-the-art generic-based and 0-1 SAT-based ILP solvers in handling the proposed ILP formulation.

This paper is organized as follows. Section 2 presents background information on ILP, SAT and its applications as well as MANETs and the clustering problem. Section 3 covers the existing work done in the use of ILP formulation in modeling the clustering problem. Section 4 describes the proposed ILP formulation of the clustering problem in MANETs. Section 5 presents the tests conducted and an evaluation of the results obtained. The paper is concluded in Section 6.

2. BACKGROUND

This section provides background information on Integer Linear Programming (ILP), Boolean Satisfiability (SAT), Mobile Ad-Hoc Networks and a detailed look at the clustering problem.

2.1. Integer Linear Programming and Boolean Satisfiability

Integer Linear Programming (ILP) involves maximizing or minimizing a function with respect to certain constraints where the optimal function and constraints are linear and the used variables can only take integer values [2]. Cases where the integer values are restricted to (0-1) are referred to as Binary ILP Problems. In Boolean Satisfiability (SAT) the constraints between variables are represented using what is called propositional logic. Propositional logic involves the use of AND, OR and NOT operations to construct formulas in the Products-of-Sums form (also called the Conjunctive Normal Form (CNF)). The variables can only take Binary values (0-1). Given constraints expressed in CNF, the goal is to identify a variable assignment that will satisfy all constraints in the problem or prove that no such assignment exists. In a propositional formula, given n variables, there are 2^n different possible variable assignments. In order to *solve* or rather *satisfy* the formula, SAT will go through the search space and determine whether or not there is a satisfying variable assignment. Advanced decision heuristics and intelligent conflict diagnosis techniques can be used to avoid searching through the entire tree of 2^n assignments.

While SAT solvers have traditionally been used to solve *decision* problems, recently SAT solvers have been extended to handle pseudo-Boolean (PB) constraints [3, 4] which are simple inequalities that are equivalent to 0–1 ILP constraints. PB constraints can replace an exponential number of CNF constraints. Another key advantage of PB constraints is the ability to express *optimization* problems which were traditionally handled as ILP problems. Studies have shown that 0-1 SAT-based ILP solvers can compete with the best available generic-based ILP solvers in solving 0-1 ILP problems arising in specific applications [3, 4]. The recent advances in SAT solvers as well as the availability of increasingly affordable high computational power, have allowed larger problem instances to be solved in different applications domains. Such applications include Power Optimization [5], FPGA [6], Communications [7], Access Control [8], Cryptography [9], Application Mapping [10], Genetics [11] and Scheduling [12].

2.2. Mobile Ad-Hoc Networks and the Clustering Problem

MANETs are wireless, self-organizing networks consisting of mobile nodes with generally a limited supply/store of energy. These nodes can be for example, laptops, mobile radio terminals or other devices, generally those which are used by humans [13]. Several challenges are faced in enabling MANETs to communicate through a stable, scalable, flexible topology. Over the years much research has been undertaken in enabling MANETS to operate in the optimum state, i.e., minimizing energy consumption and essentially attempting to achieve the maximum network lifetime through optimizing cluster formation, routing and communication.

978-1-4673-0382-8/12/$31.00 ©2012 IEEE

Initially MANET topologies were *flat* networks or non-hierarchical networks where all nodes had identical roles. Through various tests and simulations conducted, it was proven that as the number of nodes in flat networks increases, the throughput falls drastically [14]. In addition several factors such as frequent route breakage, unpredictable topology changes, routing overhead make it difficult for a *flat* topology to be scalable [15]. The concept of *clustering* was introduced to overcome the scalability limitations of a flat network. Clustering involves dividing the network into clusters with certain nodes in each cluster being chosen to be *clusterheads*. The clusterheads have the responsibility of managing communication and routing for their particular cluster and because of this, the selection of clusterheads is particularly important [16].

Selection of clusterheads is not trivial. There are several issues that need to be considered when selecting clusterheads. One issue is that the clusterheads are not selected for the lifetime of the network but rather are re-selected at certain intervals. This is because of the fact that clusterheads are responsible for routing and communication and as a result they use more energy than regular nodes. If they remain clusterheads they will be the first nodes to be depleted. In order to maximize network lifetime, the responsibility of being a clusterhead is rotated between nodes. Another reason for re-clustering is that since the nodes are mobile, some nodes may move out of range of one clusterhead and in range of another and so the topology must adjust accordingly.

3. EXISTING ILP FORMULATIONS OF THE CLUSTERING PROBLEM

ILP formulations of the clustering problem in MANETs are limited. The earliest contribution, to the best of our knowledge, can be traced to 2004, where the authors in [15] formulated a non-ILP algorithm clustering mechanism called Virtual Grid Architecture (VGA) and a corresponding ILP formulation to use as a basis for comparison. The objective of the ILP formulation was to find the minimum set of connected clusterheads. The topologies generated were 1-hop, and due to the capabilities of the solver, the network size was limited to 30 nodes when comparing the performance of VGA to ILP. Solving the ILP formulation for their 30 node network using CPLEX took 1011.5 seconds (~17 minutes). Test results showed that topologies generated through the ILP formulation had fewer clusterheads as compared to those generated with VGA. While the work put forward by the authors in [15] could be considered the first attempt at using ILP formulation in relation to the clustering problem, the first truly significant attempt at applying ILP formulation to the clustering problem was the work put forward by the authors in [17] in 2006. Unlike the model presented in [15], the authors did not focus on obtaining the minimum number of clusterheads but rather focused on the selection of a specified number of clusterheads, the interconnection of regular nodes and clusterheads, and the interconnection of clusterheads in a backbone, such that a specified maximum cluster size was not exceeded, and such that the maximum possible network lifetime was obtained.

The authors, in [17], proposed three different ILP formulations, each with a different approach to the creation of a backbone. The first formulation, Energy Efficient Clustering - Fully Connected Backbone (EEC-FCB), involved connecting the backbone of selected clusterheads through a mesh topology. The second formulation, Energy Efficient Clustering - Connected Backbone (EEC-CB), relaxed the constraints requiring mesh interconnectivity of the backbone of clusterheads, thereby reducing the number of redundant connections. The third formulation, Energy Efficient Clustering (EEC-R), formulated the application of a backup clusterhead for each selected clusterhead. Figure 1 shows the topologies generated by the three formulations.

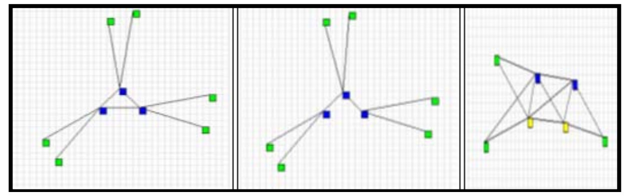

Figure 1. Different network topologies: Fully Connected Backbone, Connected Backbone, and Redundant Models [17].

The EEC-FCB model produced too many redundant links in the backbone, particularly when generating configurations with a large number of clusterheads. The EEC-CB model used a Master Clusterhead (MCH) which reduced the number of redundant links but introduced a possibility of the MCH being a central point of failure. Due to the complexity of the ILP formulations and the limitations of the generic-based ILP solver used, only ILP formulations of networks with up to 9 nodes could be solved. The proposed formulations did not undergo significant testing with a variety of generic-based ILP solvers. Additionally, the coverage radius of nodes was not considered. It was assumed that all nodes could communicate with each other. This work represented the first significant ILP formulation of the clustering problem, and provided a platform to enhance significantly.

4. PROPOSED ILP FORMULATION

This paper proposes an ILP formulation of the clustering problem, building on the ideas and assumptions put forward in the EEC-CB model presented in [17]. This model improves on weaknesses present in the EEC-CB model and adds redundancy through the use of a *Star-Ring* backbone. Additionally, a proposed enhancement allows coverage to be taken into account.

4.1. Proposed Base Model

The variables used in [17] are maintained as follows:

- N: Total number of nodes in the network (predetermined)
- P: Number of clusters heads (predetermined)
- d_{ij}: Euclidean distance between nodes i and j
- K_j: Max number of nodes that can be connected to CH j (predetermined)
- c_{ij}: Cost of connecting a regular node i to CH j (proportional to d_{ij}^2)
- h_{jk}: Cost of connecting CH j to CH k (proportional to d_{jk}^3)
- x_{ij}: Variable. 1 if node i is connected to CH j; 0 otherwise
- z_{ij}: Variable. 1 if CH i is connected to CH j; 0 otherwise
- y_j: Variable. 1 if node j is chosen to be a CH; 0 otherwise
- w_{ij}: Variable. 1 if x_{ij} = 1and y_j = 1; 0 otherwise.
- b_j: Weight associated with CH j.

The following assumptions which were made in the ILP formulations in [17] are also applicable to the proposed ILP formulation. The variable b, in the objective function, which represents the level of the node's capability to act as a clusterhead, gets its value from an external source (algorithm, tool, etc). This is useful as multiple approaches/algorithms, which determine the suitability of a node in acting as a clusterhead, can be combined with this model without changing the equations, although this is out of the scope of our research.

It is assumed that nodes are able to determine each other's position, either through the use of GPS, or other localization techniques. All connections are single hop.

Equation 1 is the objective function to be minimized. The structure of the objective function is kept similar to the one used in the EEC-FCB and EEC-CB models in [17].

$$Min\,(x,y,z){:}\,(\sum_{i=1}^{N}\sum_{j=1}^{N} c_{i,j}x_{i,j} + \sum_{j=1}^{N} b_j y_j + \sum_{j=1}^{N} b_j M_j + \sum_{j=1}^{N}\sum_{k=1}^{N} h_{j,k} z_{j,k} \quad (1)$$

The first term in the objective function represents the connections between nodes and clusterheads. The second term represents the selection of nodes to be clusterheads. In the proposed formulation, the Master clusterhead is not also a regular clusterhead. This results in the need for a separate term, as the selection cost of the Master clusterhead still has to be taken into account. This is the third term in Equation 1. The last term represents the connections between clusterheads. The objective function aims to minimize the cost of sending/receiving data along these connections.

Constraint 2 enforces the restriction that there is only one Master clusterhead.

$$\sum_{j=1}^{N} M_j = 1 \quad (2)$$

Constraint 3 enforces the restriction that the total number of CHs is *P* - 1. That is to say that if there are a total of *P* clusterheads, there will be 1 Master clusterhead and *P*-1 regular clusterheads.

$$\sum_{j=1}^{N} y_j = P - 1 \quad (3)$$

Constraint 4 is the upper limit on the total number of connections a node has. If a node is a regular node it can at most be connected to one other node (this node will be clusterhead as enforced by later constraints). If a node is a clusterhead, it will be connected at most to *K* other regular nodes (this enforces the restriction of maximum cluster size).

$$\sum_{i=1}^{N} x_{i,j} \le 1 + (K-1)y_j \quad \forall j \quad (4)$$

Constraint 5 is the lower limit on the total number of connections a node has. If a node is a regular node it must be connected to at least one other node (which will be a clusterhead as enforced by later constraints). If a node is a clusterhead it must support at least one node. If a node is a Master clusterhead it is not restricted to '1 connection to a regular node'. Rather, it can have (and in this case it should have) no connections to regular nodes.

$$\sum_{i=1}^{N} x_{i,j} \ \ge\ 1 - M_j \quad \forall j \quad (5)$$

Constraint 6 is the upper limit on the maximum number of backbone connections. If a node is a clusterhead it cannot have more than 3 backbone connections. (1 will be to a Master cliusterhead for the star connection, and 2 will be to other regular clusterheads in order to establish the ring links). If a node is a Master clusterhead, it will be connected to all the regular clusterheads (*P*-1).

$$\sum_{\substack{k=1 \\ j\ne k}}^{N} z_{j,k} \le (P-1)M_j + 3y_j \quad \forall j \quad (6)$$

Constraint 7 is used to enforce the lower limit on the number of backbone connections. If a node is a regular clusterhead then it has to be connected to at least two other nodes, one other regular clusterhead and one master clusterhead. If a node is a Master clusterhead, it has to be connected to all the regular clusterheads (*P-1*).

$$\sum_{\substack{k=1 \\ j\ne k}}^{N} z_{j,k} \ge (P-1)M_j + 2y_j \quad \forall j \quad (7)$$

Constraint 8 is used to enforce the restriction that backbone connections are only between the master clusterhead and regular nodes, or between regular clusterheads. The connections between regular nodes and clusterheads are not counted as backbone connections.

$$\sum_{\substack{k=1 \\ j\ne k}}^{N} z_{j,k} \le \frac{M_j + y_j + M_k + y_k}{2} \quad \forall j \quad (8)$$

Constraint 9 is used to enforce the restriction that if a node is selected to be a regular clusterhead, it cannot be the master clusterhead and vice versa. The node can only be one of the two.

$$\sum_{j=1}^{N} (y_j + M_j) \le 1 \quad (9)$$

Constraint 10 is used to ensure that nodes are not connected to themselves and Constraint 11 is used to diagonalize the matrix *x* which represents the connections between regular nodes and regular clusterheads. That is to say that if clusterhead 1 is connected to node 2, it is the same as saying that node 2 is connected to clusterhead 1. Constraint 12 does the same for the *z* matrix which represents the interconnections between clusterheads.

$$\sum_{i=1}^{N} x_{ii} = 0 \quad (10)$$

$$x_{i,j} = x_{j,i} \quad \forall i, \forall j \quad (11)$$

$$z_{i,j} = z_{j,i} \quad \forall i, \forall j \quad (12)$$

Constraint 13 restricts the total number of connections between regular nodes and clusterheads to the same number as the number of regular nodes; each regular node must be connected to at least one other clusterhead.

$$\sum_{i=1}^{N}\sum_{j=i+1}^{N} x_{i,j} = (N-P) \quad (13)$$

Constraint 14 is used to restrict the total number of backbone connections to *2(P-1) - 1*. 1 is deducted because the ring will be left 'open' as described earlier.

$$\sum_{i=1}^{N}\sum_{j=i+1}^{N} z_{i,j} = 2(P-1) - 1 \quad (14)$$

Constraint 15 is used to ensure that clusterheads do not connect to themselves.

$$\sum_{i=1}^{N} z_{ii} = 0 \quad (15)$$

Constraint 16 is used to ensure that regular nodes are not connected to each other. When *x* (non-backbone) connections are made, at least one of the nodes must be a clusterhead.

$$\sum_{\substack{k=1 \\ j\ne k}}^{N} x_{j,k} \le \frac{1 + y_j + y_k}{2} \quad \forall j \quad (16)$$

All topologies must have at least one master clusterhead and two regular clusterheads. All regular clusterheads must support at least one regular node. Master clusterheads do not connect to any regular nodes and are not counted as regular clusterheads.

Figure 2 is an example of a topology generated through this proposed model, with a Star-Ring Backbone. The 'Star' is formed through the connections of Node 2 (Master Clusterhead) to the other regular clusterheads (Nodes 1, 3, 4, 5, and 6) and the 'ring' is formed through connections between nodes 1-6, 6-5, 5-4, and 4-3.

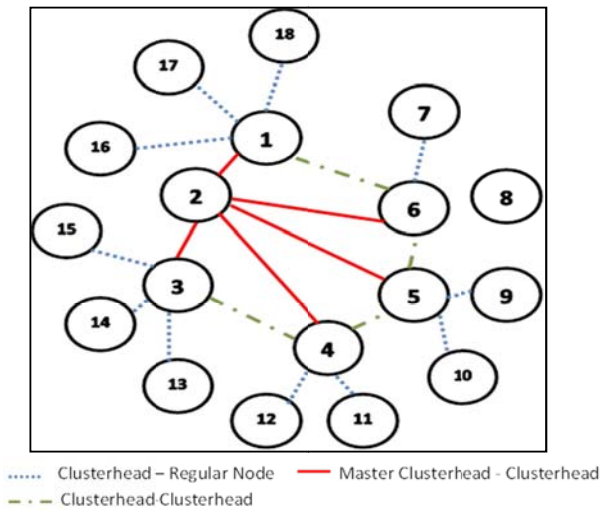

Figure 2. Star-Ring Topology

Had this been generated using the EEC-FCB model [17], all clusterheads would have been interconnected (mesh) resulting in additional unnecessary links.

4.2. Proposed Coverage Enhancement

The proposed Base Model can be extended to take into account the coverage radius of the nodes in the network, and ensure that connections are established only between nodes that are within each other's coverage radius. Similar to the manner in which distances between nodes are used to determine the cost of the connections, they can also be compared to the coverage radius of each node and used to obtain a matrix of nodes to which each node can connect to and to which it can't. This is done by making use of the matrix *d*, which is used to keep track of the distance between the nodes. Two new matrices are introduced; matrix *nc* and matrix *cv*. In equation 17, the variable $cv_{i,j}$ is the binary value which represents whether or not nodes *j* and *i* are in each other's coverage radius. This is determined by subtracting the distance between nodes from each node's coverage radius. If both results are positive then they can communicate, otherwise not. The actual establishment of the connection will depend on the cost (which is proportional to the distance).

$$cv_{i,j} = \begin{cases} 1, & \text{if node } i \text{ and node } j \text{ can connect} \\ 0, & \text{otherwise} \end{cases} \quad (17)$$

These values can then be used to enforce the possibility of connectivity between nodes using constraints 18 and 19. It is important to keep in mind, that it is being assumed that external localization techniques such as GPS are being used. Constraints 18 and 19 ensure that two nodes may only be connected if they lie within each other's coverage radius.

$$\sum_{i=1}^{N} x_{i,j} \leq cv_{i,j} \quad \forall j \quad (18)$$

$$\sum_{i=1}^{N} z_{i,j} \leq cv_{i,j} \quad \forall j \quad (19)$$

4.3. Illustrative Example

The example topology shown in Figure 3 will be used to illustrate the difference when the Star-Ring model is used without coverage and when the coverage constraints are included. In Figure 4 (without coverage), node 5 and node 1 were connected and node 3 and node 7 were connected, even though they were outside of each other's coverage radius. These connections are not made when coverage constraints are enforced and all nodes connected are within each other's coverage radius.

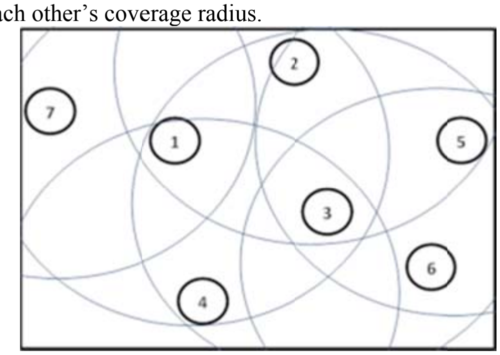

Figure 3. Sample MANET topology.

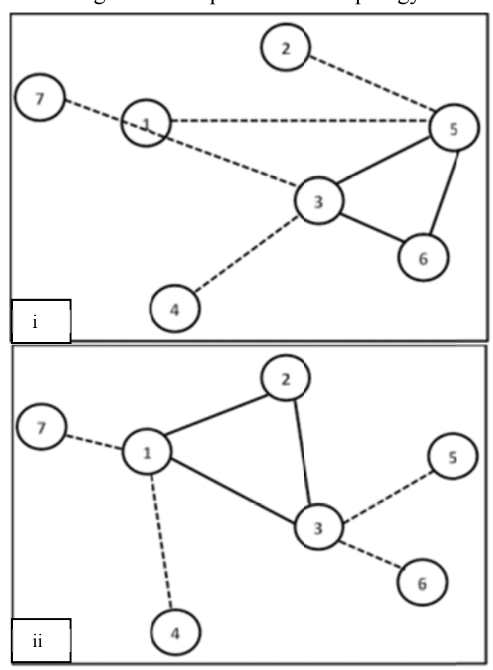

Figure 4. Topology generated by the ILP formulation i) Without coverage constraints and ii) with coverage constraints.

5. TESTING AND RESULTS

Testing was carried out using the following solvers: commercial generic-based ILP solver CPLEX [18], non-commercial generic-based ILP solver SCIP [19], 0-1 SAT-based ILP solvers BSOLO [20], Pueblo [21] and Minisat+ [22]. The SAT solvers were among the winners in recent SAT competitions. All experiments were conducted on an Intel Xeon 3.2 Ghz workstation running Linux with 4 GB of RAM. Testing was carried out for various network configurations. For a fixed number of nodes, the number of clusterheads, and the maximum supported clustersize were varied to analyze their effect on the solvers' performance in solving the ILP formulation. The performance of the set of solvers was assessed in solving the proposed ILP Star-Ring model.

TABLE I: SOLVER PERFORMANCE IN SOLVING THE EEC-FCB AND SR ILP FORMULATIONS OF THE CLUSTERING PROBLEMS. '-' AND '!' REPRESENTS 'CANNOT SOLVE' AND 'TIMEOUT', RESPECTIVELY.

Network Configurations			**Solver Times (seconds)**				
			Proposed SR Model				
# N	***#C H***	***#M CS***	***CPLEX***	***SCIP***	***BSOLO***	***Pueblo***	***Minisat +***
5	3	1	0.257	0.014	0.002	0.001	0.038
7	3	2	0.285	0.023	0.007	0.008	0.180
9	3	3	0.376	0.060	0.026	0.040	1.180
11	3	4	0.468	0.148	0.063	0.349	5.79
13	3	5	0.637	0.428	0.281	-	31.85
15	3	6	0.725	1.017	0.950	-	242.52
40	3	19	**5.73**	121.18	!	!	!
45	3	21	**8.27**	162.58	!	!	!
50	3	24	**16.29**	303.25	!	!	!
7	4	1	0.356	0.051	0.013	0.011	0.37
9	4	2	0.546	0.150	0.055	0.072	6.73
11	4	3	0.571	0.297	0.152	0.531	76.19
13	4	3	0.795	0.967	1.030	-	349.54
15	4	4	0.903	1.709	4.753	-	!
40	4	12	**45.33**	304.85	!	!	!
45	4	14	**94.39**	385.25	!	!	!
50	4	16	**271.92**	665.64	!	!	!
9	5	1	0.532	0.353	0.098	0.107	8.59
11	5	2	0.745	0.9	0.366	1.833	200.95
13	5	2	0.834	2.058	1.633	-	!
15	5	3	1.035	3.212	7.433	-	!
40	5	9	**112.92**	489.4	!	!	!
45	5	10	**195.17**	!	!	!	!
50	5	12	**427.51**	!	!	!	!

The results are shown in Table I. For each model and each network configuration shown in Table I, 100 tests were generated and solved by the different solvers. The times shown for each network configuration in Table I are the average of the corresponding 100 instances. A timeout of 15 minutes (900 seconds) was set for all solvers. As can be seen from Table I, the SAT solvers such as BSOLO and Pueblo performed well for the smaller scale networks, however, CPLEX and SCIP proved to be faster as the size of the network increased. The comparison of times taken by CPLEX to solve a topology with a fixed number of nodes but with different specified clusterheads is shown in Figure 5. From Figure 5, it is observed that for a given number of nodes, CPLEX take a longer time to solve topologies which have a larger number of clusterheads. The other solvers behave similarly. It can be clearly seen, that for a fixed number of nodes, if topologies with a larger number of clusterheads is to be generated, the solvers will take more time to generate the solution. Similar, observations can be made for the other solvers used. Additionally, it is observed that Pueblo [23] is unable to handle certain instances and ends up in the "Cannot Solve" state shown by a '-' in Table I. This is due to Pueblo's inability to handle problems with large coefficients.

The large coefficients present in the ILP formulations are the costs associated with interconnecting nodes. (The cost of the link connecting a regular node to a clusterhead is proportional to the square of the distance between the nodes, and the cost of interconnecting clusterheads is proportional to the cube of the distance between the clusterheads [17].)

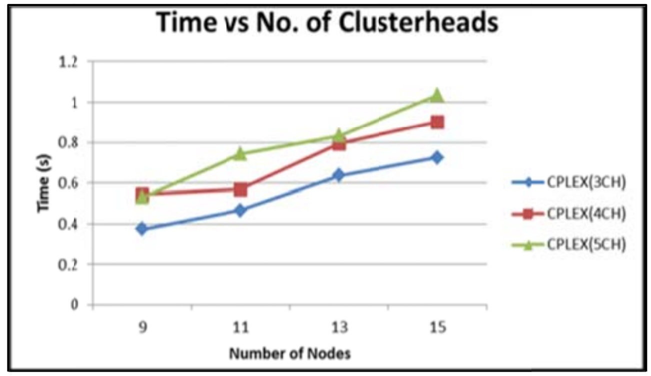

Figure 5. Dependency of CPLEX solver runtimes on the number of desired clusterheads in the solution.

Overall, it is observed that CPLEX and SCIP perform well, with MINISAT+ being the slowest solver for the presented benchmarks. Among the set of selected solvers, CPLEX and SCIP handle the larger networks well as they almost never timeout. In the case of the Star-Ring (SR) model, the SAT solvers BSOLO and Pueblo are very fast for the smaller networks, however as the size of the network increases, their time-to-solve increases faster than the generic ILP solvers, i.e. CPLEX and SCIP. MINISAT+ times out and CPLEX and SCIP are the fastest solvers for the SR models. The solver used to solve the ILP formulation presented in [19], timed out when solving for more than 9 node topologies. In the tests conducted with the proposed ILP formulations and enhancements, solvers such as CPLEX and SCIP were able to handle ILP formulations of networks up to 50 nodes. CPLEX in particular is far from timing out even at 50 node topologies. It is important to note that the timeout used in testing was 15 minutes (900 seconds) and does not accurately reflect a real-life setting, where clustering would need to be done much faster.

6. CONCLUSION

This paper put forward the formulation of a redundant ILP model to solve the clustering problem in MANETs. The proposed model presented the use of a redundant Star-Ring backbone. Additionally, the proposed formulation included the ability to enforce coverage constraints to ensure that only connections that are within the physical limitations of the node are established. Using the proposed ILP formulations and enhancements together with a custom designed tool, it was possible to test the performance and analyse the feasibility of Generic ILP and SAT solvers (CPLEX [18], SCIP [19], BSOLO [20], Pueblo [21] and Minisat+ [22]) in solving the clustering problem for MANETs. SAT solvers, BSOLO and Pueblo performed well for small scale networks while CPLEX and SCIP were able to handle the larger scale topologies. In most cases, CPLEX was the fastest solver from the selected set of solvers. While suitable for use in small scale MANETs, for larger networks time taken by the selected set of solvers to solve the enhanced formulations is too large to be feasible for practical environments.

11. REFERENCES

[1] S. Chinara and S. Rath, "Energy Efficient Mobility Adaptive Distributed Clustering Algorithm for Mobile Ad Hoc Network," in *International Conference on Advanced Computing and Communications*, 2008.

[2] Alexander Schrijver, "Theory of Linear and Integer Programming." *John Wiley & Sons Ltd*., 1999.
[3] F. Aloul, A. Ramani, I. Markov, and K. Sakallah, "Generic ILP versus specialized 0–1 ILP: an update," in *International Conference on Computer Aided Design (*ICCAD*)*, 450–457, 2002.
[4] D. Chai and A. Kuehlmann, "A fast pseudo-Boolean constraint solver," in *Design Automation Conference* (DAC), 2003.
[5] A. Sagahyroon, F. Aloul, and A. Sudnitson, "Using SAT-Based Techniques in Low Power State Assignment," in *Journal of Circuits, Systems, and Computers* (JCSC), 20(8), 1605-1618, 2011
[6] A. Ling, D. Singh, and S. Brown, "FPGA Programmable logic block evaluation using quantified Boolean Satisfiability," in *IEEE Computers and Digital Techniques*, 153(3), 165-172, May 2006.
[7] F. Aloul and M. El Tarhuni, "Multipath Detection Using Boolean Satisfiability Techniques," in *Journal of Computer Networks and Communications* (JCNC), vol. 2011, Article ID 365107, 2011.
[8] Graham Hughes and Tefvik Bultan, "Automated Verification of Access Control Policies using a SAT solver," in *International Journal on Software Tools for Technology Transfer* (STTT), 10(6), 503-520, October 2008.
[9] Ilya Mironov and Lintao Zhang, "Applications of SAT solver to Cryptanalysis of Hash Functions," in *International Conference on Theory and Applications of Satisfiability Testing*, 102-115, 2006.
[10] S. Tosun, O. Ozturk, and M. Ozen, "An ILP formulation for application mapping onto Network-on-Chips," in *AICT*, 2009.
[11] Ana Graca, Ines Lynce, Joao Marques-Silva, and Arlindo L. Oliveira, "Haplotype inference with pseudo-Boolean optimization," in *Annals of Operations Research*, 184(1), 137-162, 2011.
[12] Ahmed Wasfy and Fadi A. Aloul, "Solving the University Class Scheduling Problem," in *IEEE GCC Conference*, 2007.
[13] JA Garcia-Macias and Javier Gomez, "MANET versus WSN," in *Sensor Networks and Configuration: Fundamentals, Standards, Platforms, and Applications*. Springer, ch. 17, 369-388, 2006,.
[14] K. Xu, X. Hong, and M. Gerla, "An Ad Hoc Network with Mobile Backbones," in *IEEE International Conference on Communications*, 2002.
[15] Jamal N. Al-Karaki, Ahmed E. Kamal, and Raza Ul-Mustafa, "On the Optimal Clustering in Mobile Ad hoc Networks," in *Consumer Communications and Networking Conference*, 2004.
[16] Haidar Safa, Omar Mirza, and Hassan Artail, "A Dynamic Energy Eficient Clustering algorithm for MANETs," in *IEEE Int'l Conference on Wireless & Mobile Computing, Networking and Communnication,* 2008.
[17] Wasim El-Hajj, Dionysios Kountanis, Ala Al-Fuqaha, and Hani Harbi, "Optimal hierarchical Energy Efficient Design for MANETs," in *Int'l Conference on Communications and Mobile Computing*, 287-292, 2006.
[18] IBM ILOG CPLEX Optimizer, 2011. [Online]. *http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/*
[19] SCIP: An Open Source MIP Solver and constraint integer programming framework, 2011. [Online]. *http://scip.zib.de/*
[20] V. M. Manquinho and J. Marques-Silva. "On using cutting planes in pseudo Boolean optimization," in *Journal on Satisfiability, Boolean Modeling and Computation* (JSAT), 2:209–219, 2006.
[21] Hossein Sheini and Karem Sakallah, "Pueblo: A Hybrid Pseudo-Boolean SAT Solver," in *Journal on Satisfiability, Boolean Modeling and Computation*, 2(1-4), 165-189, 2006.
[22] N. Een and N. Sorensson, "An extensible SAT-solver," in *Int'l Conf. on Theory and Applications of Satisfiability Testing*, 502–508, 2003.