Two level Evolutionary Algorithm for Capacitated Network Design Problem

Meriem Khelifi
Laboratoire Réseaux et Systèmes
Université Badji Mokhtar, Algerié
Email:khelifi.meriem@lrs-annaba.net

Saadi Boudjit
L2TI, Institus Galilée
Université Paris 13, France
Email: boudjit@univ-paris13.fr

Mohand Yazid Saidi
L2TI, Institus Galilée
Université Paris 13, France
Email: saidi@univ-paris13.fr

***Abstract*—Efficient design of networks topologies is challenging, especially with the arrival of the virtualization in these last years. In this paper, we deal with the Capacitated Network Design Problem (CNDP) with modular link capacities to design minimum cost network while satisfying the flow demands. We propose a two levels Genetic Algorithm (GA) based model that can deal with several variations of CNDP. Our proposition defines a new encoding scheme to treat the modular case. Extensive simulation results on Atlanta, France and Germany network instances show that the proposed algorithm is much more efficient than the Iterative Local Search algorithm.**

keywords: Network Design Problem; two level optimization; Genetic algorithm; Modular capacity; Multicommodity flow problem.

I. Introduction

To save resources (routers, optical fibers, etc.) networks should be efficiently designed. Diverse networks models were then defined and used to represent a wide range of issues in transportation, telecommunications, logistics, production and distribution networks. All these models consider a graph composed of nodes and edges (optical fibers, cables, etc). For a better use of these resources, networks designers should solve the modular Capacitated Network Design Problem (CNDP) which consists of selecting edges and the optimal capacities to allocate to route a set of commodities between a source and destination pairs. Each edge of the graph has a potential set of module capacities with their associated costs, a fixed cost that is incurred only if the edge is selected, and a routing cost which is proportional to the amount of flows along the edge. Each commodity is defined by an origin and a destination node and the amount to be routed. The objective is to minimize three criteria: edge cost, modules and routing. These capacitated network design problems are NP-hard and very difficult to solve in practice. The CNDP is a particular case of the well known Multicommodity Network Design problem (MNDP), in which we distinguish an important number of special cases and extensions [1]. The most studied ones are:

- The unsplittable variant where the flow of each commodity is required to follow one route between the origin and the destination, which increases the difficulty of the problem [2].
- The expansion variant, where some edges already have an existing capacity.
- The fixed charge MNDP [3][4] in which the link capacities are known. Solving MNDP consists to determine the set of edges that should be opened in the final topology.
- The capacitated MNDP, where the number of modules to install on the edges are modeled by integers [1]
- The Network Loading Problem (NLP), where the number of module types is limited, each one with a given unit cost and capacity.

Various heuristics and exact approaches have been developed for designing capacitated networks. However, the heuristic approaches are more likely to be trapped in local optima, while the exact approaches are applied only to small or medium size problems. Due to the weaknesses of the two approaches and the increasing popularity of metaheuristic approaches, we have witnessed many metaheuristics being applied to network optimization problems. In this paper, we propose a novel metaheuristic that is based on Genetic Algorithms (GAs), which has extensively been used to solve many difficult combinatorial optimization problems in industrial engineering and operations research. Genetic algorithms are one of the most powerful and broadly applicable stochastic search and optimization techniques and have achieved great advancement in related research fields, such as network optimization, combinatorial optimization, multi-objective optimization, and on so on. Our contribution consists on an efficient two level evolutionary algorithm that uses the GA and the Linear Programming (LP) to solve a general model that can deal with diverse variants of capacitated network design problems. We define a Modular Implicit Encoding (IME) to encode individuals which is a very flexible encoding scheme.

The remainder of this paper is structured as follows: related work is introduced in section 2, notations and mathematical formulation of the addressed problem are given in section 3. Section 4 describes and explains in details our proposed algorithm. Experimental results are discussed in section 5 where we compare our proposition against Iterative Local Search (ILS) technique. Finally, section 6 concludes the paper.

II. Related work

Capacitated Network Design Problem is one of the major research area in network optimization. It is related to two issues: Network Design Problem (NDP) and Network Loading Problem (NLP). In the NDP, the goal is to identify the network topology by selecting routers and links that interconnect

them. Thus, the objective function aims to minimize the total constructive cost under some topological constraints. In this class of problems, the flow is not modeled, consequently it is considered as uncapacitated. In the NLP, it is assumed that the topology is already established. Thus, solving NLP consists to search for the set of resources to allocate for the network components. These problems are complementary. Generally, NDP and NLP are solved separately though they are combined together in some cases. One can say that most of network optimization problems can be seen as a kind of (1) NDP, (2) NLP or (3) a combination of both where the objective and the constraints may differ from one problem to another: connectivity [5] [6], limited budget [7], hop limit [8] [2], delay [9] [10], reliability [8] [9], and survivability [10]. The NLP in capacitated or uncapacitated case and with both single or multiple facilities is a special case of the well known Multicommodity Network Design Problem (MCND). Previous works on this problem can be classified as:

- Uncapacitated network design problems where on each link of the network, it is only possible either to open the link with infinite capacity and a given fixed cost, or the cost is zero and no capacity is available [11].
- Single facility capacitated network loading problem where, the capacity can be done by installing on each link an integer unit of a given basic facility [12].
- Two facilities capacitated network loading problems where the capacity can be achieved by means of two types of modules, each capacity has a specific cost citerefKerivin.
- Multi-type facility capacitated network loading problems where various types of capacities can be installed on each link, each facility has a specific cost [7].

The early works on capacitated modular network problems were focused on the approximation methods. These methods define residual capacity and cut-set inequalities for single commodity and multicommodity cases on directed, undirected and bidirected link models [13][14]. Since these works consider that the underling network is established, they focus only on the determination of the facilities allowing the accommodations of flow demand. Their effectiveness depends on the size of the problem instance.
With the appearance of metaheuristics, both the NLP and the NDP have attracted some attention. The authors benefit from their efficiency to deal with more complex variants with real size instances. In [15], the author compared several neighborhood structures to solve the uncapacitated facility location problem. In [10], the authors proposed an evolutionary approach for capacitated network design considering cost, performances and survivability. The objective is to minimize network cost and packet delay. Kleeman et al. [9] used an evolutionary algorithm to solve multicommodity capacitated network design problem with an objective function optimizing costs, delay, robustness, invulnerability and reliability. A tabu search heuristic algorithm with real costs on facilities is developed in [16]. A firefly algorithm is proposed by Ragheb et al [7], they combined facility location and network design problem with multi-type of capacitated link and limited budget on facilities. Contreras et al. [17] presented a unified framework of general network design problems which combine location decision and network design decision.

III. Mathematical Formulation

Let $G = (V, E)$ be an undirected network where V is the set of vertices and E is the set of undirected edges. Let K be the set of commodities. For each one $k \in K$, P^k is the set of paths associated to commodity k, and d^k the flow demand of commodity k. Let f_{ij} be the fixed cost of including edge (i, j) in the network, r_{ij} the unit variable flow cost on (i, j), and p_{ij} the pre-installed capacity on the edge (i, j).
The formulation of CNDP is shown below:

$$Min \quad z(x,y,n) = \sum_{(i,j)\in E} r_{ij} \sum_{k\in K} \sum_{P\in P^k} x_p^k + \sum_{(i,j)\in E} f_{ij} y_{ij} + \sum_{(i,j)\in E} \sum_{l\in L} c_{ij}^l n_{ij}^l$$

$$\sum_{p\in P(k)} x_p^k = t(k), \qquad \forall k \in K\,. \tag{1}$$

$$\sum_{k\in K} \sum_{p\in P^k} x_p^k \le p_{ij} + \sum_{l\in L} m_l\, n_{ij}^l\, y_{ij}, \qquad \forall (i,j) \in E \tag{2}$$

$$\sum_{k\in K} \sum_{p\in P^k} x_p^k \le d^k y_{ij}, \qquad \forall (i,j) \in E, \forall k \in K \tag{3}$$

$$y_{ij} \in \{0,1\}, \qquad \forall (i,j) \in E \tag{4}$$

$$x_p^k \ge 0, \quad \forall k \in P^k, k \in K \tag{5}$$

$$n_{ij}^l \in Z^+, \qquad \forall l \in L, \forall (i,j) \in E \tag{6}$$

This formulation is a mixed integer linear program which uses three types of variables: the first type is a binary design variable, which is defined as $y_{ij} = 1$ if (i, j) is included in the network and $y_{ij} = 0$ otherwise. The second type is a continuous path flow variable x_p^k, which represents the amount of flow of commodity k routed on $p \in P^k$. The third type is an integer allocation module variable, which is defined by n_{ij}^l. It represents the number of module type l allocated on edge (i, j), where L is the set of potential modules. Each $l \in L$ is characterized by a capacity m_l and an installation cost c_{ij}^l. A positive capacity of edge (i, j) implies that it is used to route demands in the two directions: from i to j or from j to i. This formulation corresponds to a general model that can deal with several variants of capacitated network design problems.

The objective function correspond to the sum of the variable flow costs of commodities, the fixed costs of edges and the allocated module costs. These costs are relative to the problem that we deal with and are not all aggregated in some cases. For instance, the fixed charge problem MNDP includes only the edge costs. The modules and routing costs on edges are nil. Constraints (1) consist of flow conservation equations, which represent the fact that the sum of path flows of commodity k is equal to the demand. Constraints (2) provide the capacity constraints, which prohibit flowing if the edge is excluded, $y_{ij} = 0$, and allow for flow up to the edge capacity

if the edge is included, $y_{ij} = 1$. Constraints (3) provide the forcing constraints, which prohibit flowing commodity k if the edge is excluded, and allow for flow up to the demand if the edge is included. Constraints (4) and (5) express respectively the binarity and the non-negativity of variables decisions. Constraints (6) show that the modules facility are allocated in a discrete amounts.

Recall that when the edges and the modules are fixed and known (i.e. $\bar{y}, \bar{n}$ is already determined) the path based formulation of CNDP becomes a Capacitated Multicommodity Flow Problem (CMFP) that is solvable in polynomial time and formulated bellow:

$$Min \;\; z(x(\bar{y},\bar{n})) = \sum_{(i,j)\in E(\bar{y})} r_{ij} \sum_{k\in K} \sum_{P\in P^k} x_p^k$$

Subject to (1) and

$$\sum_{k\in K} \sum_{p\in P^k} x_p^k \leq p_{ij} + \sum_{l\in L} m_l \; n_{ij}^l \; y_{ij}, \qquad \forall (i,j) \in E(\bar{y}) \quad (7)$$

$$\sum_{k\in K} \sum_{p\in P^k} x_p^k \leq d^k y_{ij}, \qquad \forall (i,j) \in E(\bar{y}), \forall k \in K \quad (8)$$

$$x_p^k \geq 0, \quad \forall k \in P^k, k \in K \quad (9)$$

A solution to the CNDP can be viewed as a binary assignment $(\bar{y})$ to each design variable, an integer vector assignment $(\bar{n})$ to the allocation module design variables and the optimal flow of the corresponded multicommodity minimum cost flow problem $x^*(\bar{y},\bar{n})$. So the objective function value associated to a solution $(\bar{y},\bar{n},x^*(\bar{y},\bar{n}))$ is the sum of the fixed cost of the open edges in $(\bar{y})$, the cost of the modules allocated $(\bar{n})$ and the objective function value of the CMFP associated to $x(\bar{y},\bar{n})^*$.

In the next section, we use the genetic algorithms to explore different potential solution areas by choosing various vector values $(\bar{y},\bar{n})$. Then, we apply the CMFP model (the above linear program) to solve CNDP.

IV. Genetic Algorithm for CNDP

Genetic algorithms introduced by Holland [18], are based on the mechanics of natural selection and natural genetics. They start with an initial set of random solutions, called a population. Each individual in the population is called a chromosome, representing a solution to the problem. The initial population evolves through successive iterations, called generations. A measure of fitness defines the quality of an individual chromosome. In each generation, chromosomes are evaluated by a fitness function, also called an evaluation function. After a number of generations, highly fit individuals, which are analogous to good solutions to a given problem, will emerge. Genetic algorithms consist of five components:

1. A method for encoding potential solutions into chromosomes;
2. A means of creating the initial population;
3. An evaluation function that can evaluate the fitness of chromosomes;
4. Genetic operators that can create the next generation population;
5. A way to set up control parameters; e.g., population size, the probability of applying a genetic operator, etc.

A. Individual representation

In the design of genetic algorithms, the encoding is the most important task. There are some methods to encode each individual in a population, such as binary encoding, integer encoding, etc. In this paper, we define a new encoding method called IME (Implicit Modular Encoding) that is relative to our modular case. An individual built by IME is shown in figure 1. Each individual I is a matrix $I_{n,m}$, where n and m corresponds to the number of modules and to the number of edges respectively. Hence, $I[l_i][e_j]$ gives the number of module types l_i allocated on edge e_j.

I	e_1	e_2	e_3	e_4	.....			.....	.....	.....	e_m
l_1	0	1	4	3	.....			.....	1	1	0
l_2	3	2	1	0	.....			.....	3	0	2
⋮	⋮	⋮								⋮	⋮
l_n											

Fig. 1: Individual with Implicit Modular Encoding

Our encoding represents the decision vector n and implicitly the decision vector x. For example, the case of one module type, $T[l_1][e_i] = 2$, means that we allocate two modules on the edge e_i. Thus, we implicitly deduce that edge e_i exists in the final topology. However, edge e_j will not be opened in the final network since $T[l_1][e_j] = 0$. When multiple types of modules are allowed, the edge exists if at least one module is allocated on it, i.e:

$$\begin{cases} x_e = 1 & if \quad \sum_{l_i=1}^{n} T[l_i][e] > 0 \\ x_e = 0 & otherwise \end{cases}$$

B. Initial population

The initial population is generated according to the algorithm depicted in figure 2. There are two ways to generate an initial population; random initialization and heuristic initialization. In our case,we used the heuristic one. We started with an initial feasible solution, which is obtained by an Iterative Local search (ILS) algorithm [19]. We encoded this solution according to IME and we considered it as the initial individual I_0. Then, we apply some perturbation on I_0 (by deleting one module) to generate other individuals, wherein there are some not feasible. For each new generated individual, we used the LP solver (CPLEX optimizer) to check for the existence of feasible flows. An individual is added to the initial population if feasible flows are determined.

Algorithm 1 InitialPopulation

Inputs: S_0 , K
Local variables: P_0
$P_0 \longleftarrow \emptyset$
Generate initial individual I_0 throughout encoding ILS solution with IME encoding
$P \longleftarrow I_0$
foreach *module* $\{l_n \in L\}$ **do**
  foreach *link* $\{e_m \in E\}$ **do**
    if $I_0[l_n][e_m] > 0$ **then**
      Generate new individual $I^{'} \longleftarrow I$
      $I^{'}[l_n][e_m] = I_0[l_n][e_m] - 1$
      if $(CplexSolver(I^{'}, K)$ == $True)$ **then**
        Add individual $I^{'}$ to P_0
      end
    end
  end
end
Return P_0

Fig. 2: S_0 is the ILS solution, K is the set of flow demands. P_0 is the initial population. L is the set of capacity modules. E is the set of links. I_0 is the initial individual. $I^{'}$ is the new individual. $CplexSolver()$ is the procedure that solve the $MCFP$ on the network represented by individual $I^{'}$, returns $True$ if it find a feasible flow.

C. Fitness function

The fitness function corresponds to the objective function of CNDP. It is computed as the sum of the allocated module costs, the fixed edge costs and the routing costs. Note that the first two costs are deduced from the individual representation whereas the routing costs are given by solving the CMNP linear program.

D. Genetic operator

We randomly chose two integers ($0 < x_1 \leq x_2 \leq m$) and two individuals (I_1 and I_2) in the current population. Then, we used two-point crossover operator to generate new individuals (see figure 3). Typically, a new individual I_{new} is generated by selecting the modules of edges in $[e_1, e_{x_1}[\cup [e_{x_2}, e_m]$ from I_1 and $[e_{x_1}, e_{x_2}]$ from I_2.

E. The genetic algorithm

After explaining and detailing the basic components of our proposed genetic algorithm, we describe below its operation (see figure 4 for instructions). In our algorithm, we first initialize the population through $InitialPopulation()$ procedure. Then N successive populations are generated by applying the two-point crossover operator ($Crossover()$).

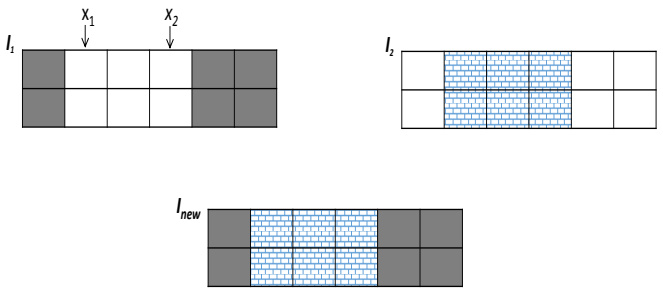

Fig. 3: Individual with Implicit Modular Encoding

Algorithm 2 Genetic-Algorithm

Inputs: S_0, K
Local variables: i, P , $Gbest$
$P \longleftarrow InitialPopulation(S_0, K)$
$Gbest \longleftarrow$ best individual in the population P
$Termination \longleftarrow false$
while ! *Termination* **do**
  $i \longleftarrow$ size of population P
  while $i <$ *Max-Size* **do**
    $(I_1, I_2) \longleftarrow$ RandomSelection(P)
    $(x_1, x_2) \longleftarrow$ RandomSelection(P-size)
    $(x_1, x_2) \longleftarrow Order(x_1, x_2)$
    $I_{new} \longleftarrow Crossover(I_1, I_2, x_1, x_2)$
    if $(CplexSolver(I_{new}, K) == True)$ **then**
      Add individual I_{new} to population P
      Update Gbest
      i $\longleftarrow$ i+1
    end
  end
  P $\longleftarrow$ CleanPop (P)
  Update(Termination)
end
Return $Gbest$

Fig. 4: S_0 is the initial solution, K is the set of flow demands. $Gbest$ is the best value . P is the current population. $MaxSize$ is the fixed size of the population. I is the individual. $CplexSolver()$ is the procedure that solves the $CMFP$ on the network represented by individual $I^{'}$. It returns the value $True$ if it founds a feasible flow.

As said previously, only individuals allowing a feasible multicommodity flow solution are added to the current population. This is verified by the running of $CplexSolver()$ procedure. The best solution $Gbest$ is updated at each generation and returned when the termination condition is satisfied. $CleanPop()$ procedure allows to switch from one population to another by selecting individuals from the first population. It is based on elitist strategy. The algorithm stops its running after a fixed number of generations or when the result is not improved after a certain number of generations.

V. RESULTS

In our experiments, we used three real world instances of network topologies including Atlanta, France and Germany50. All can be downloaded from $http://sndlib.zib.de$ [20]. We followed the model filter specified in table I. The population size is 50 and the number of generations is 15.

TABLE I: The model filter

Demand model	Undirected demand (U)
Link model	Undirected links (U)
Link capacity model	Modular link capacities(M)
Fixed-charge model	No fixed-charge cost (N)
Routing model	Continuous (C)
Admissible path model	All paths (A)
Hop limit model	No hop-limits (N)
Survivability model	No survivability (N)

Each instance is characterized by the number of nodes $|V|$, the number of potential links $|E| = m$ and the number of traffic demands $|K|$. Table II summarizes the instance specification details. We classify them into two categories; instances with Single Facility allocation (SF) and instances with Two Facilities allocation (TF). The set of capacity modules *L* differs from one network instance to another. The allocation cost *MCost* is variant on links and it is fixed, however, in France instance. Atlanta instance assumes a Pre-installed capacities p_{ij} on their potential links with a unit routing cost r_{ij}. See [20], for more details on the filter model and on the setting parameters.

TABLE II: The Instance Setting Parameters

Problem Instance	$\|V\|$	$\|E\|$	$\|K\|$	Nbr	L	$MCost$	p_{ij}	r_{ij}
Atlanta	15	22	210	TF	1000, 4000	variant	yes	yes
France	25	45	300	SF	2500	fixed	no	no
Germany50	50	88	662	SF	40	variant	no	no

In table III, ILS and GA corresponds to the solutions obtained by iterative local search algorithm and genetic algorithm respectively. We examine the quality of a given algorithm A (A could be GA or ILS) by computing its optimality gap (see equality 10) that is defined as the ratio between the difference of the A's cost and the Best Solution (BS) cost. Note that BS corresponds to the best solutions published in [20].

$$GAP(A) = \{Cost(A) - Cost(BS)\}/Cost(BS) * 100 \quad (10)$$

TABLE III: The ILS and GA solutions

Instance	BS	ILS	$Gap\%$	GA	$Gap\%$
Atlanta	86492550	92904547	7.41	87959303	1.69
France	20200	21400	5.94	20600	1.98
Germany50	645520	719060	11.39	667840	3.45

As depicted in Table IV, GA is better than ILS since it determines solutions more close to the optimums than those of ILS. Concretely, the mean gap obtained with ILS is 3.5 times higher than the mean gap obtained with GA. This can

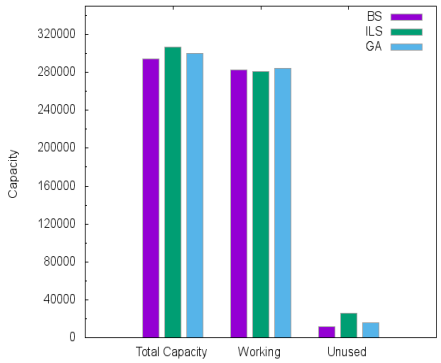

Fig. 5: Atlanta network

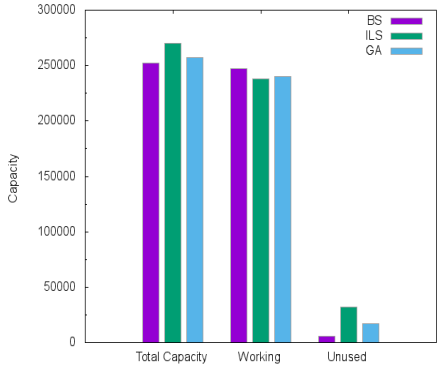

Fig. 6: France network

be explained by the exploration of multiple solution areas with GA while ILS determines only a local minimum.

Figures 5, 6 and 7 show the allocated capacities and their usage for Atlanta, France and Germany50 networks respectively. We compared the total installed link capacities, the total working capacities and the total unused capacities for BS, ILS and GA solutions. We remark that the total installed link capacities in ILS and GA are more larger than the BS ones. This justifies the cost gap. On the other hand, ILS uses much more working capacities than BS because ILS wastes module resources. Indeed, instead of splitting flows and exploring the small unused capacities on links, ILS routes the majority of demands on single shortest paths. With GA, the CPLEX optimizer try to exploit the residual quantities on the allocated modules to route flows. This leads to a bifurcation of demands on multiple paths that could be arbitrary long (although routing costs slightly limit the path lengths).

VI. CONCLUSION

In this paper, we proposed a two level evolutionary approach to solve several special cases and variants of the capacitated network design problem. Our algorithm has two levels, the higher one is the genetic algorithm, that deals with the link selection and the modules allocation decisions. The lower level is the LP solver (CPLEX optimizer), which fixes the routing decision by searching for a feasible flow according to the

TABLE IV: Working and Unused capacities

Instance	Atlanta			France			Germany50		
	BS	ILS	GA	BS	ILS	GA	BS	ILS	GA
Total installed link capacities	294000	307000	300000	252500	270000	257500	7200	8000	7440
Total working flow	282338.50	281188	284503	246938	237952	240351	7140	7024	7265.83
Total Unused flow	11661.5	25812	15497	5562	32048	17149	60	976	174.17

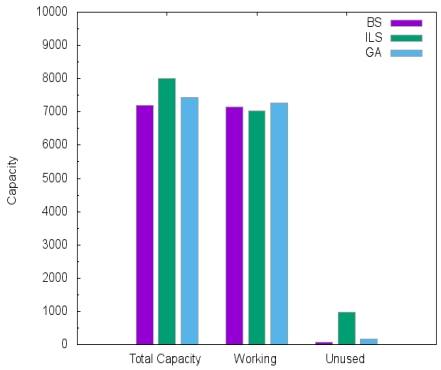

Fig. 7: Germany50 network

network configuration made in the higher level. For efficiency, we rigorously defined the main components of the genetic algorithm. The initial solution is generated by an iterative local search algorithm, which is combined with an heuristic procedure to construct the initial population. To better explore the solution space, we defined a very flexible and meaningful encoding scheme called IME (Implicit Modular Encoding), two point crossover operator and an elitist population strategy. The results are very satisfactory. Indeed, the basic idea of combining genetic algorithms and linear programming for solving the problem in two levels is effective. Simulations show that our two level approach is better than the iterative local search approach since it determines solutions close to the best known ones.

REFERENCES

[1] A. Frangioni and B. Gendron, "0â1 reformulations of the multicommodity capacitated network design problem," *Discrete Applied Mathematics*, vol. 157, no. 6, pp. 1229 – 1241, 2009.

[2] B. Thiongane, J.-F. Cordeau, and B. Gendron, "Formulations for the nonbifurcated hop-constrained multicommodity capacitated fixed-charge network design problem," *Computers Operations Research*, vol. 53, no. 0, pp. 1 – 8, 2015.

[3] I. Rodríguez-Martín and J. J. Salazar-González, "A local branching heuristic for the capacitated fixed-charge network design problem," *Computers & Operations Research*, vol. 37, no. 3, pp. 575–581, 2010.

[4] D. C. Paraskevopoulos, T. Bektaş, T. G. Crainic, and C. N. Potts, *A Cycle-Based Evolutionary Algorithm for the Fixed-Charge Capacitated Multi-Commodity Network Design Problem*. Tech. rep., Centre interuniversitaire de recherche sur les réseaux dâentreprise, la logistique et le transport, Montreal, 2013.

[5] A. Balakrishnan, M. Banciu, K. Glowacka, and P. Mirchandani, "Hierarchical approach for survivable network design," *European Journal of Operational Research*, vol. 225, no. 2, pp. 223–235, 2013.

[6] M. Abd-El-Barr, "Topological network design: A survey," *Journal of Network and Computer Applications*, vol. 32, no. 3, pp. 501 – 509, 2009.

[7] R. Rahmaniani and A. Ghaderi, "A combined facility location and network design problem with multi-type of capacitated links," *Applied Mathematical Modelling*, vol. 37, no. 9, pp. 6400–6414, 2013.

[8] T. Liu, W. Yang, and J. Huang, "Reliable network design problem under node failure with benders decomposition," *Applied Mathematics*, vol. 5, p. 241, 2014.

[9] M. Kleeman, G. Lamont, K. Hopkinson, and S. Graham, "Solving multicommodity capacitated network design problems using a multiobjective evolutionary algorithm," in *IEEE Symposium on Computational Intelligence in Security and Defense Applications*, 2007, pp. 33–41.

[10] A. Konak and A. Smith, "Capacitated network design considering survivability: an evolutionary approach," *Engineering Optimization*, vol. 36, no. 2, pp. 189–205, 2004.

[11] T. Magnanti, P. Mireault, and R. Wong, *Tailoring Benders decomposition for uncapacitated network design*. Springer, 1986.

[12] F. Barahona, "Network design using cut inequalities," *SIAM Journal on optimization*, vol. 6, no. 3, pp. 823–837, 1996.

[13] A. Atamtürk, "On capacitated network design cut–set polyhedra," *Mathematical Programming*, vol. 92, no. 3, pp. 425–437, 2002.

[14] A. Atamtürk and D. Rajan, "On splittable and unsplittable flow capacitated network design arc–set polyhedra," *Mathematical Programming*, vol. 92, no. 2, pp. 315–333, 2002.

[15] D. Ghosh, "Neighborhood search heuristics for the uncapacitated facility location problem," *European Journal of Operational Research*, vol. 150, no. 1, pp. 150–162, 2003.

[16] O. Brun, J. Garcia *et al.*, "A tabu search heuristic for capacitated network design," in *IEEE Symposium on Computers and Communications,ISCC*, 2009, pp. 461–467.

[17] I. Contreras and E. Fernández, "General network design: A unified view of combined location and network design problems," *European Journal of Operational Research*, vol. 219, no. 3, pp. 680–697, 2012.

[18] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine learning*, vol. 3, no. 2, pp. 95–99, 1988.

[19] M. Khelifi, A. Nakib, and B. Malika, "Hybrid heuristic for capacitated network design problem," in *Proc. Of the Int. Conf. on Metaheuristics and Nature Inspired Computing, META*, 2014, pp. October 26–31.

[20] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "Sndlib 1.0 survivable network design library," *Networks*, vol. 55, no. 3, pp. 276–286, 2010.