Constraint Based Local Search for Distance and Capacity Bounded Network Design Problem

Alejandro Arbelaez*, Deepak Mehta*, Barry O'Sullivan* and Luis Quesada*
*Insight Centre for Data Analytics
University College Cork, Cork, Ireland
Email: {alejandro.arbelaez,deepak.mehta,barry.osullivan,luis.quesada}@insight-centre.org

***Abstract*—Many network design problems arising in the fields of transportation, distribution and logisitics require clients to be connected to facilities through a set of carriers subject to distance and capacity constraints. Here a carrier could be a cable, vehicle, salesman etc. The distance from a facility to client using a carrier could be expressed as signal loss, time spent, path length, etc. The capacity of a carrier could be interpreted as the maximum number of commodities that a carrier can carry, the maximum number of clients or links that a single carrier can visit, etc. The main decisions are to determine the number of carriers, assign clients to carriers, and design a network for each carrier subject to distance, capacity and some side constraints. In this paper, we focus on Cable Routing Problem (CRP), which is NP-hard. We present a constraint-based local search algorithm and two efficient local move operators. The effectiveness of our approach is demonstrated by experimenting with 300 instances of CRP taken from real-world passive optical network deployments in Ireland. The results show that our algorithm can scale to very large size problem instances and it can compute good quality solutions in a very limited time.**

I. Introduction

Many network design problems arising in the fields of transportation, distribution and logistics require clients to be connected to facilities through a set of carriers subject to distance and capacity constraints. Here a carrier could be a cable, vehicle, salesman etc. The distance of a carrier from a facility to a client could be expressed as signal loss, time spent, path length, etc. The capacity of a carrier could be interpreted as the maximum number of commodities that it can carry, the maximum number of clients or links that a carrier can be connected to, etc. Two well-known classes of such network design problems are multiple Travelling Salesmen Problem and Vehicle Routing Problem (VRP).

The multiple travelling salesman problem (mTSP) is a generalisation of the well-known travelling salesman problem (TSP) [1]. In mTSP we are given a set of clients, one site where salesmen are located, and a cost metric, the objective of the mTSP is to determine a tour for each salesman such that the total tour cost is minimised and that each client is visited exactly once by only one salesman. The number of salesmen may be a fixed number or may be determined by the solution but bounded by an upper bound. Some extensions of mTSP have distance constraints where the time required to complete any tour should not exceed a maximum allowed policy. Applications of this problem are school bus routing, crew scheduling, aircraft scheduling etc [2].

The mTSP is a restriction of Vehicle Routing Problem (VRP) which has numerous applications in real world [3]. In VRP a set of routes for a fleet of vehicles based at one or more sites must be determined for a number of geographically dispersed clients. The objective is to deliver a set of clients with known demands on minimum-cost vehicle routes originating and terminating at a site. There are several variants to VRP like the capacitated VRP (CVRP), VRP with Time Windows (VRPTW) [4]. In the CVRP, a fleet of identical vehicles located at a central facility has to be optimally routed to supply a set of customers with known demands subject to a given capacity. The objective of the VRPTW is to serve a number of customers within predefined time windows at minimum cost (in terms of distance travelled), without violating the capacity and total trip time constraints for each vehicle.

In both mTSP and VRP the main decisions are to determine the number of carriers, assign each client to a carrier, and determine which links are used for connecting clients to the facility subject to a set of constraints. One of the constraint in these problems is that the network design for each carrier is a tour (or graph cycle). We focus on a network design problem arising in optical networks where a bound is given on the number of carriers and each selected carrier is connected to a set of clients using a tree topology that respects distance and capacity constraints and additionally a set of side constraints. We define this problem as *Cable Routing Problem* (CRP).

In optical network a carrier is an optical cable fibre that originates from the central office and it is splitted in a number of cable fibres in order to connect to a given number of customers. The quality of the signal deteriorates due to length of the fibre and the splitting of the fibre to connect a number of customers. Depending on the technology used, there is a threshold on the allowed signal attenuation. Consequently, there is a tradeoff between the distance and capacity limits, i.e., as the distance limit (i.e., the length of the fibre) increases, the capacity limit (i.e., the number of clients that can be connected to the fibre) decreases. The goal is to resolve the trade-off between the distance and capacity limits optimally to design minimum cost network. The CRP is NP-hard since it involves finding minimum spanning trees with bounded path length [5]. We present a constraint-based local search algorithm and two efficient local move operators. The effectiveness of our approach is demonstrated by experimenting with 300 problem instances of CRP taken from real-world passive optical network deployments in Ireland. The size of these instances can vary from few tens of clients to few tens of thousands of clients. We study the trade-off between distance and size limits and show that our approach can compute very good quality solutions for very large size instances in a very

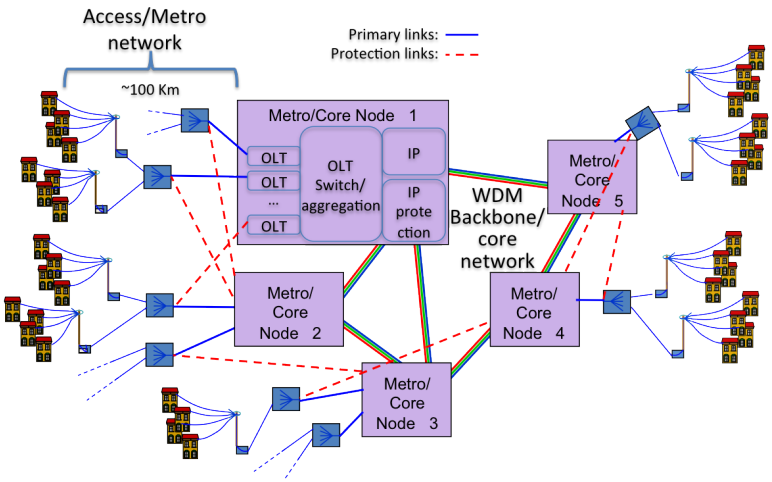


Fig. 1. Example of a Long-Reach Passive Optical Network

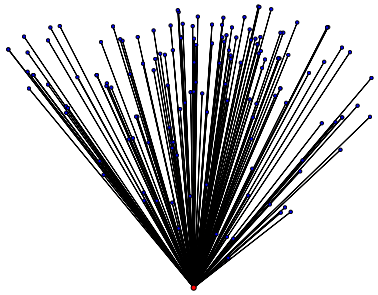
limited time.

II. Problem Description

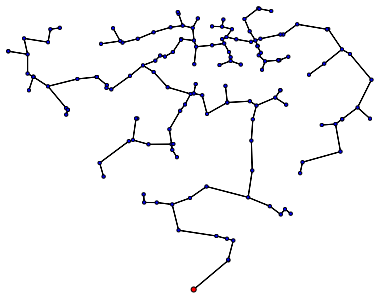
The motivation behind the work presented in this paper comes from a real-world problem arising in the domain of optical networks. Long Reach Passive Optical Networks (LR-PONs) are gaining increasing interest as they provide a low cost and economically viable solution for fibre-to-the-home network architectures [6]. The LR-PON architecture consists of three subnetworks: (1) Optical Distribution Network (ODN) for connecting customers to exchange-sites, (2) Backhaul Network for connecting exchange-sites to metro-nodes, and (3) Core Network connecting pairs of metro-nodes. An example of a Long-Reach PON is shown in Figure 1.

We focus on ODN part of LR-PON consisting of exchange-sites and customers. Notice that an exchange-site is a facility, a customer is a client and fibre is a carrier. In Figure 1 the ODN network consists of nodes corresponding to blue boxes and houses. Here the biggest blue boxes can be seen as exchange-sites and houses are customers. In LR-PON an exchange-site could be connected to tens of thousands of customers. In the optical distribution network an optical cable fibre is routed from an exchange site to a set of customers which forms a tree distribution network. A PON is a tree network associated with each cable. The optical signal attenuation in PON is due to the number of customers in the PON and the maximum length of the fibre between the exchange site and the customer. As signal attenuation is allowed up to some threshold, the upper-bound on the length of the cable fibre from the exchange site to any customer varies with respect to the size of the PON. In [7] the authors show the relationship between the size and the maximum length of the PON,

In Figure 2 we show two ways of connecting a given set of customers to an exchange-site. In the first case (Figure2(a)) we are directly connecting each customer to the exchange site. Certainly the option of connecting each customer directly to the exchange site leads to shorter connection paths. However, the drawback of connecting each customer directly is the total amount of fibre cable used. In the second case (Figure 2(b)) we are computing a minimum spanning tree rooted at the exchange-site. Certainly this option minimises the total length of fibre cable but the drawback is that we might be violating the maximum fibre cable length allowed between the exchange-site and any of its customers.



(a)



(b)

Fig. 2. (a) Customers are directly connected to an exchange-site. (b) Customers are connected to the exchange-site through a spanning tree.

We are interested in both restricting the length of the paths and the total amount of fibre cable used. Keeping both requirements is known to be a hard problem [5]. There has been a significant amount of work on this bounded version of the spanning tree problem (see [8] for a short summary

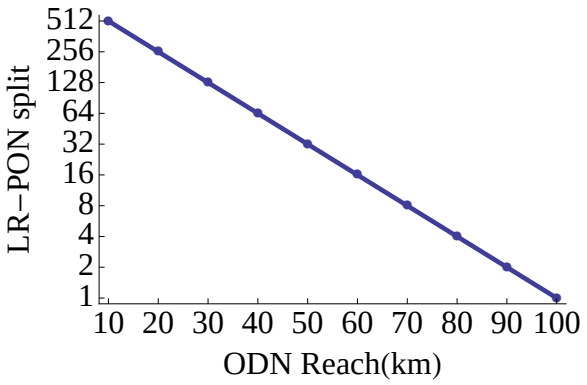


Fig. 3. Relationship between the number of customers and the maximum fibre length

of the most relevant approaches). Constraint programming based techniques has also been suggested for tackling this problem [8], [9].

The Cable Routing Problem (CRP) that we are tackling in this paper is more complicated than the bounded spanning tree problem. In CRP we want to determine a set of distance bounded and capacity bounded cable tree networks such that each tree is rooted at exchange-site, each customer is present in exactly one tree and the total cost is minimised. The number of trees denotes the number of optical fibre that run from the exchange-site to the customers. The deterioration in the quality of the optical signal can not surpass a given threshold. Notice that the signal deterioration depends both on the maximum length of the fibre cable and the number of customers of a given tree. Therefore, when the former increases the latter decreases and vice-versa. The relationship between them is shown in Figure 3. In other words, in CRP we are given a set of customers, an exchange-site and a cost function, and the objective is to determine a number of fibre cables that start at the exchange-site and create a tree distribution network for each fibre cable such that each customer is included in exactly one tree, the total cost of all trees is minimised and both distance and capacity bounds are respected.

III. Constraint Optimisation Formulation

In the following we present some notations, the formal definition of the problem and the constraint optimisation model of the CRP:

- $G = (V, E)$ is a given network
- $V = \{v_0, v_1, \ldots, v_n\}$ is a set of vertices
- v_0 is the facility and $V \backslash \{v_0\}$ is a set of clients
- $E = \{(v_i, v_j) | v_i, v_j \in V; i \neq j\}$ is an edge set
- C is a matrix of non-negative costs c_{ij} between customers v_i and v_j
- D is a matrix of distances d_{ij} between customers v_i and v_j
- m is the number of cables leaving central facility
- T_i is the tree distribution network for cable i

The Cable Routing Problem consists in finding a set of m bounded trees of minimum total cost, starting at facility, such that every vertex in $V \backslash \{v_0\}$ is included exactly in one tree. A feasible solution is composed of a partition $R_1, \ldots, R_m$ of $V \backslash \{v_0\}$ and a tree $T_i = (R_i \cup \{v_0\}, L_i)$ with root v_0 and $L_i \subseteq E_{\downarrow R_i \cup \{v_0\}}$,which means that T_i is a tree whose root (v_0) has one outgoing link. T_i is feasible if the distance from the facility to the client does not exceed a given bound and the size of the tree in terms of number clients does not exceed a given capacity limit.

Unlike other problems where the distance and capacity bounds are independent, in this work the bound on the size of the tree varies with respect to that of the maximum path length from the root-node to any leaf-node. Below is the constraint optimisation formulation of the cable routing problem arising in LR-PON.

Variables

- Let x_{ij} be a Boolean variable for each $\langle i, j \rangle \in V^2$ that denotes whether a link between node i and node j exists in the tree.
- Let y_{ij} be a non-negative integer variable that denotes the number of customers in the sub-tree emanating from node j.
- Let z_i be a non-negative integer variable that denotes the length of the path from the facility to the client v_i.

Constraints

Each node (except root-node) must have one incoming link:

$$\sum_{v_i \in V} x_{ij} = 1, \quad \forall_{v_j \in V \backslash \{v_0\}}$$

The root-node is connected to at least one another node:

$$\sum_{v_j \in V \backslash \{v_0\}} x_{0j} \geqslant 1$$

If there is a link from $v_i \in V$ to $v_j \in V$ then the length of the path from root-node to v_j is equal to the length of the path from root-node to v_i plus the distance between nodes v_i and v_j:

$$x_{ij} = 1 \Rightarrow z_j = z_i + d_{ij}, \quad \forall_{\{v_i, v_j\} \subseteq V}$$

The number of customers relying on any link node is one more than the number of nodes in the sub-tree emanating from that node.

$$y_{ij} = \sum_{v_k \in V} y_{jk} + 1, \quad \forall_{\{v_i, v_j\} \subseteq V}$$

If there is no link between nodes i and j then of course no customers are relying on this link in the tree

$$\forall_{\{v_i, v_j\} \in V} : x_{ij} = 0 \implies y_{ij} = 0$$

The number of customers in any tree is dependent on the maximum distance between a customer and the facility. For example, if the maximum path length is greater than 10 then the cable tree network cannot contain more than 512 customers. Below we show constraints for 10, 20, 30 and 40

KMs and the similar constraints for 40 to 100 in steps of 10 KMs can be enforced.

$$z_i > 10 \implies y_{ij} < 512, \quad \forall_{\{v_i,v_j\}\subseteq V}$$
$$z_i > 20 \implies y_{ij} < 256, \quad \forall_{\{v_i,v_j\}\subseteq V}$$
$$z_i > 30 \implies y_{ij} < 128, \quad \forall_{\{v_i,v_j\}\subseteq V}$$
$$z_i > 40 \implies y_{ij} < 64, \quad \forall_{\{v_i,v_j\}\subseteq V}$$

Objective Function The main objective is to minimise the total cable lengths.

$$\min \sum_{v_i \in V} \sum_{v_j \in V} d_{ij} \cdot x_{ij}$$

One might also be interested in minimising the number of cable tree networks:

$$\min \sum_{v_j \in V \setminus \{v_0\}} x_{0j}$$

IV. Local Search Algorithm

In this paper, we explore an iterative constraint-based local search [10], [11] algorithm to tackle the problem. The general scheme of the algorithm (depicted in Algorithm 1) comprises two phases. First, in a local search phase, the algorithm improves the current solution, little by little, by performing small changes. Generally speaking, it employs a move operator in order to move from one solution to another in the hope of improving the value of the objective function. Second, in the perturbation phase, the algorithm perturbs the incumbent solution (s*) in order to escape from difficult regions of the search (e.g., a local minima). Finally, the acceptance criterion decides whether to update s* or not. To this end, with a probability 5% s'* will be chosen, and the better one otherwise.

Our algorithm starts with a given initial solution where a given set of clients is partitioned and each element of that partition is associated with a cable tree network that satisfy all constraints (i.e., the maximum path length and the maximum number of nodes in the tree). The initial solution is also computed using the local search algorithm which is described in Section IV-D. We switch from the local search phase to perturbation when a local minima is observed; in the perturbation phase we perform a given number of random moves (20 in this paper). The stopping criteria is either a timeout or a given number of iterations.

A solution is represented by a tree whose root-node is the facility and the number of immediate successors of the root-node is the number of cables starting at the facility. Notice that the facility acts as the root-node of each cable tree network. Without loss of generality we add a set of dummy clients (or copies of the facility) to the original set of clients for the purpose of ease of representation to distinguish the cable tree-networks. More precisely the set of clients, $\{v_0, \ldots, v_n\}$, is modified to $\{v_0, v_1, \ldots, v_m, v_{1+m} \ldots, v_{n+m}\}$. Recall that n is the number of clients and m is the upper bound on the number of cables that can start at the facility. In the latter set v_0 is the original facility, each $v_i \in \{v_1 \ldots v_m\}$ denote the starting point of a cable tree network, and $\{v_{1+m}, \ldots, v_{n+m}\}$ denote the original set of clients. We further enforce that each dummy client is connected to v_0 and the distance between the dummy client and the facility is zero, i.e., $\forall 1 \leqslant i \leqslant m, d_{0i} = 0$.

Algorithm 1 Iterated Constraint-Based Local Search

```
1: s_0 := Initial Solution
2: s* := ConstraintBasedLocalSearch(s_0)
3: repeat
4:    s' := Perturbation(s*)
5:    s'* := ConstraintBasedLocalSearch(s')
6:    s* := AcceptanceCriterion(s*, s'*)
7: until No stopping criterion is met
```

A. Move-Operators

In this section we propose *node* and *subtree* move operators. We use T to denote the entire tree network of a given facility and T_i to denote the cable tree network associated with i^{th} cable starting at node v_i that covers the set of clients R_i. An edge between two clients v_p and v_q is denoted by $\langle v_p, v_q \rangle$.

Node operator (Figure 4(b)) moves a given node v_i from the current location to another in the tree. As a result of this, all successors of v_i will be directly connected to the predecessor node of v_i. v_i can be placed as a new successor for another node or in the middle of an existing arc in the tree.

Subtree operator (Figure 4(c)) moves a given node v_i and the subtree emanating from v_i from the current location to another in the tree. As a result of this, the predecessor of v_i is not connected to v_i, and all successors of v_i are still directly connected to v_i. v_i can be placed as a new successor for another node or in the middle of an existing arc.

Edge Operator (Figure4(d)). In this paper we limit our attention to moving a node or a complete subtree. [12] proposed to move arcs in the context of the Constrained Optimum Path problems. Pham et al. move operator (Figure 4(d)) chooses an arc in the tree and finds another location for it without breaking the flow.

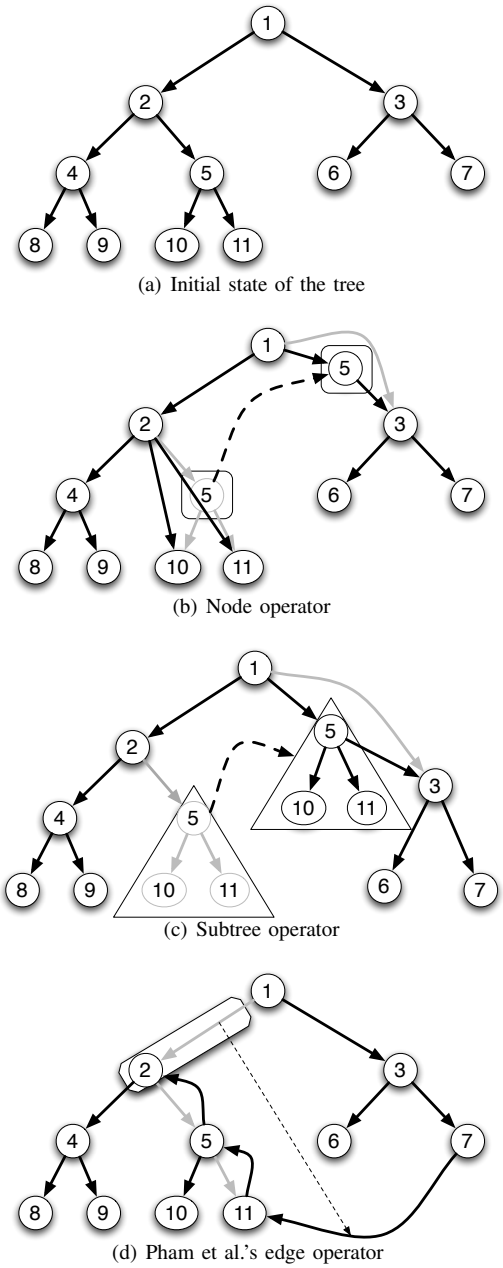
B. Operations and Complexities

We first present the complexities of node and subtree operators as they share similar features. For an efficient implementation of the move operators, it is necessary to maintain f_j denoting the length of the path from the root-node to client j and b_j denoting the length of the path from v_j down to the farthest leaf in the subtree emanating of node v_j. This information will be used to maintain the path-length constraint. We also maintain n_j denoting the number of nodes in the subtree emanating from v_j. We use $\lambda(n_i)$ to denote the maximum path length allowed for a subtree emanating at node v_i when it contains n_i number of clients. Let v_{p_j} be the immediate predecessor of v_j and let S_j be the set of immediate successors of v_j in T. Table I summarises the complexities of the move operators.

TABLE I. Complexities of different operations

	Node	Subtree	Arc
Delete	$O(n)$	$O(n)$	$O(1)$
Feasibility	$O(1)$	$O(1)$	$O(n)$
Move	$O(n)$	$O(n)$	$O(n)$
Best move	$O(n)$	$O(n)$	$O(n^3)$

Delete. Removing a node v_j from the tree T requires a linear complexity. For both operators, it is necessary to update $b_{j'}$ for



(a) Initial state of the tree

(b) Node operator

(c) Subtree operator

(d) Pham et al.'s edge operator

Fig. 4. Move operators

all the nodes j' in the path from the root-node to client v_{p_j} in T. In addition, the node operator updates $f_{j'}$ for all the nodes j' in the subtree emanating from v_i. After deleting a node v_j or a subtree emanating from v_j, the objective function is updated as follows:

$$obj = obj - c_{j,p_j}$$

Furthermore, the node operator needs to add to the objective function the cost of disconnecting each successor element of v_j and reconnecting them to v_{p_j}.

$$obj = obj + \sum_{k \in S_j} (c_{k,p_j} - c_{kj})$$

Furthermore, we decrease the number of nodes in the path from the root-node to v_{p_j} accordingly, that is, decreasing one unit for the node operator and n_j units for the subtree operator.

Feasibility. Checking feasibility for adding v_j in the cable tree network T_i can be performed in linear time. Let s be the number of nodes that will be added in the tree, i.e., 1 for the node operator and n_j for the subtree operator. If v_j is inserted between an arc $\langle v_p, v_q \rangle$ then we check the following:

$$\max(f_p + c_{pj} + c_{jq} + b_q, f_i + b_i) \leqslant \lambda(n_i + s)$$

If v_p is a leaf-node in the tree and v_j is placed as its successor then the following is checked:

$$\max(f_p + c_{pj} + b_j, f_i + b_i) \leqslant \lambda(n_i + s)$$

Move. A move can be performed in linear time. We recall that this move operator might replace an existing arc $\langle v_p, v_q \rangle$ with two new arcs $\langle v_p, v_j \rangle$ and $\langle v_j, v_q \rangle$. This operation requires to update f_j for all nodes in the emanating tree of v_j, and b_j in all nodes in the path from the facility acting as a root node down to the new location of v_j. The objective function must be updated as follows:

$$obj = obj + c_{pj} + c_{jq} - c_{pq}$$

In addition to modifying the objective, f_j and b_j for a certain set of nodes, we also update the n_{p_j} after applying the move-operators. On one hand, the subtree (reps. node) operator increase n_j units (reps. one unit) in all nodes in the path from the root-node to n_{p_j}. Whenever the node operator is breaking an existing link $\langle v_p, v_q \rangle$, n_j would be set to n_p + 1.

Best Move. Selecting the best move involves traversing all clients associated with the facility and selecting the one with the maximum reduction in the objective function.

Now we switch our attention to the edge operator. This operator does not benefit by using b_j. The reason is that moving a given arc from one location to another might require changing the direction of a certain number of arcs in the tree as shown in Figure 4(d). Deleting an arc requires constant complexity, this operation generates two separated subtrees and no data structures need to be updated. Checking the feasibility of adding an arc $\langle v_{p'}, v_{q'} \rangle$ to connect two subtress requires linear complexity. It is necessary to actually traverse the new tree to obtain the distance from $v_{q'}$ to the farthest leaf in the tree. Performing a move requires a linear complexity, and it involves updating f_j for the new emanating tree of $v_{q'}$. Consequently, performing the best move requires a cubic time complexity, the number of possible moves is n^2 (total number of possible arcs for connecting the two subtree) and for each possible move it is necessary to check feasibility. Due to the high complexity ($O(n^3)$) of the arc operator to complete a move, hereafter we limit our attention to the node and subtree operators.

C. Constrained-Based Local Search

Algorithm 2 depicts general scheme for the constraint based local search algorithm, In each iteration of the loop (Lines 2-9), it selects a node v_j randomly and then identifies the best location for v_j. Instead of verifying that the local minima is reached by exhaustively checking all moves for all clients, we maintain a list of nodes that must be visited. The list is initialised in Line 1 with all nodes in the tree. Once a node is explored, it is removed from the list. The nodes would

Algorithm 2 ConstraintBasedLocalSearch(*move-op*,T)

1: *list* $\leftarrow \{v_i \in T\}$
2: **while** *list* $\neq \varnothing$ **do**
3: Select v_j randomly from *list*
4: $(v_q, S, Cost) \leftarrow BestMove(\textit{move-op}, T, v_j)$
5: **if** Cost $\neq \infty$ **then**
6: *list* $\leftarrow \{v_i \in T\}$
7: *list* $\leftarrow$ *list* $- \{v_j\}$
8: $T \leftarrow$Move(T,*move-op*, (v_q, S_j), v_j)
9: return T

be added back in the list when an improvement in the objective function has been observed (Lines 5-7). We recall that after reaching a local minima, the iterated constraint-based local search algorithm perturbs the solution by applying a certain number of random moves in the tree (either using node or subtree operators).

The pseudo-code to compute the best move is depicted in Algorithm 3. The input parameters of the algorithm are: the move-operator which is itself a function, tree T, and the node v_j. It first delete v_j from T. In each iteration of the loop (Lines 4–11), it identifies a best location for v_j. A location in a tree T is defined by (v_q, S) where v_q denotes the parent of v_j and S denotes the set of successors of v_j after the move is performed.

Broadly speaking, there are two options for the new location: (1) Breaking an arc $\langle v_p, v_q \rangle$ and inserting v_j in between them such that the parent node of v_j would be v_p and the successor set would be singleton, i.e., $S = \{v_q\}$; (2) Adding a new arc $\langle v_p, v_j \rangle$ in the tree in which case the parent of v_j is v_p and $S = \varnothing$. *Locations* returns all the locations relevant with respect to a given move-operator (Line 4). Line 5 verifies that the new move is not breaking any constraint and *CostMove* returns the cost of applying such a move using a given move-operator (Line 6).

If the cost of new location is the best so far then the best set of candidates is reinitialised to that location (Lines 7-9). If the cost is same as the best known cost so far then the best set of candidates is updated by adding that location (Lines 10-11). The new location for a given node is randomly selected from the best candidates if there are more than one (Lines 12). The best move returns the best move for v_j, the one that minimises the overall cost (i.e., sum of edge's cost) of building the distance and size constrained rooted minimum spanning tree.

D. Finding Initial Solution

Similar to other meta-heuristics, our local search approach needs to be equipped with an initial solution. The quality of the initial solution can have a significant impact on the the quality of the final solution when the solution time is limited. We use a constraint-based local search algorithm for computing a good quality initial solution that uses the earlier proposed move operators. The pseudo-code is shown in Algorithm 4. The algorithm starts by creating a tree rooted at the facility, and then connect all the dummy clients with the root-node (Lines 1-2). Afterwards, it iteratively add one client randomly at a time (Lines 4-12). It invokes Algorithm 3 to compute the best location with respect to the cost for adding the client in the tree. If the node cannot be added the algorithm removes k (5 in this paper) randomly selected nodes from the current solution and perturbs the current state of the solution in order to allow a more diversified solution.

Algorithm 3 BestMove (*move-op*,T, v_j)

1: Delete v_j from T and update T
2: *Best* $\leftarrow \{(v_j, S_j)\}$
3: *cost* $\leftarrow \infty$
4: **for** (v_q, S) in *Locations*(*move-op*, T) **do**
5: **if** FeasibleMove(*move-op*, $(v_q, S), v_j$) **then**
6: $cost' \leftarrow$ CostMove((v_q, S), v_i)
7: **if** $cost' < cost$ **then**
8: *Best* $\leftarrow \{(v_q, S)\}$
9: *cost* $\leftarrow cost'$
10: **else if** $cost' = cost$ **then**
11: *Best* $\leftarrow$ *Best* $\cup \{(v_q, S)\}$
12: Select $(v_{q'}, S')$ randomly from *Best*
13: return $(v_{q'}, S', Cost)$

Algorithm 4 Initial-Solution(*move-op*, V,m)

1: Create a tree T rooted at v_0
2: $T \leftarrow T \cup \{\langle v_0, v_i \rangle | 1 \leqslant i \leqslant m\}$
3: *list* $\leftarrow \{v_j | v_j \in V \wedge m < j \leqslant n\}$
4: **while** *list* $\neq \varnothing$ **do**
5: Select v_j randomly from *list*
6: $(v_q, S, Cost) \leftarrow BestMove(\textit{move-op}, T, v_j)$
7: **if** Cost $\neq \infty$ **then**
8: $T \leftarrow$Move(T,*move-op*, (v_q, S_j), v_j)
9: *list* $\leftarrow$ *list* $- \{v_j\}$
10: **else**
11: remove a set of k original clients, V'', from T
12: *list* $\leftarrow$*list*$\cup V''$
13: $T \leftarrow$ perturbation(T)
14: **return** T

The cable routing problem has a multi-criterion objective function: minimising the number of cable tree networks and the total cable fibre cost. These two objectives might be in conflict, i.e., the solution providing the minimum number of trees might not give the overall minimum cable length cost and vice-versa. Therefore, we explore two approaches to tackle the problem. In one we fix the number of cable tree networks, and another one we provide providing an upper bound on the number cable tree networks. In both approaches, we apply the constraint-based local search framework and minimise the total cable cost of the problem.

In order to fix the number of trees for a given problem instance, we compute the number of customers that are within a certain distance from the facility (e.g., *Customers(v_0, 10km)*), and the near-optimal number of cable tree networks for customers within this distance of the facility would be set to *Customers(v_0,10km)*/512. Here 512 is the maximum size of the cable tree network when the maximum path length is 10KM). The same procedure is repeated for other sizes of cable tree networks. Taking this into account, the value of m is passed as a parameter to Algorithm4 for compute the optimal number of PONs. To this end we create dummy nodes whose location is the same as the facility. These nodes are added into the tree

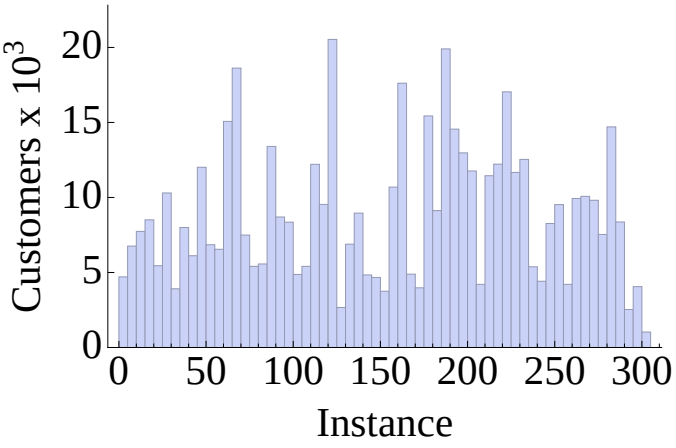


Fig. 5. Distribution of total number of customers per instance

as the direct successors of the root-node. Moreover, since we always want to maintain a fixed number of nodes during the search, no other nodes can be added as successor of the root-node, and of course dummy nodes always remain at the same position in the tree.

V. Empirical Analysis

In the experiments we explore the two versions of the local search algorithm. (i) Using a fixed near-optimal number of cable tree networks and (ii) a flexible number of cable tree networks. We recall that other algorithms to compute minimum distance constrained spanning trees (e.g., [13], [14], and [5]) cannot be used for solving our problem as these algorithms rely on the use of the shortest path from the root-node to every other node, and this path might not be available for some nodes because the distance limit varyies with the number of nodes in the tree. All the experiments where performed on a 39-node cluster, each node features a 2 Intel Xeon E5430 processors at 2.66Ghz and 12 GB of RAM memory. For each instance we report the average performance of 10 independent executions with a 5-minute time limit. We would like to remark that we experimented with the MIP model using CPLEX, but the complete solver does not scale up to more than a few tens of customers per exchange site (usually 30).

In this paper we present experimental results for 300 instances corresponding to exchange sites from real networks from Ireland (randomly selected out of 1121 ones in the country). The selected exchange sites cover 543697 customers (out of 2189120) distributed among the country. In preliminary experiments not presented in this paper, we observed that the subtree operator outperformed the node operator, and therefore we limit our attention to only subtree move operator. Figure 5 shows the number of customers per instance, as it can be observed due to the geographic of the country the size of the instances vary from dense areas with up to 11000 customers per instance and rural areas with at least 70 customers per instance. Figure 6 shows an example of the resulting tree after applying the proposed constraint-based local search algorithm, this instance contains 6419 customers and the solution features 16 cable tree networks (or PONs).

Figure 7 depicts the histogram of the distribution of the distances from the facility to the customers using the flexible

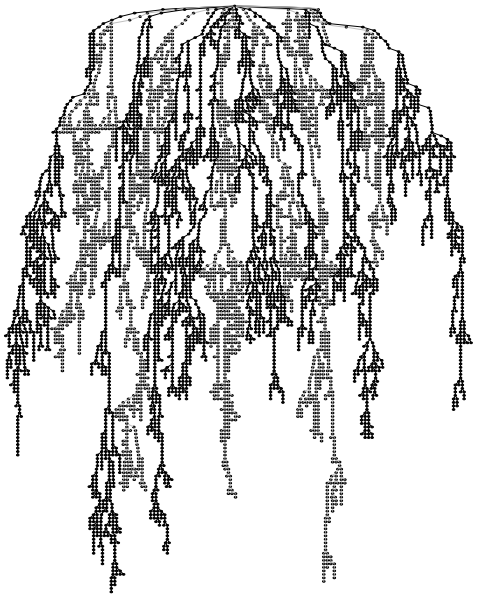


Fig. 6. Solution example for a local exchange with 6419 customers

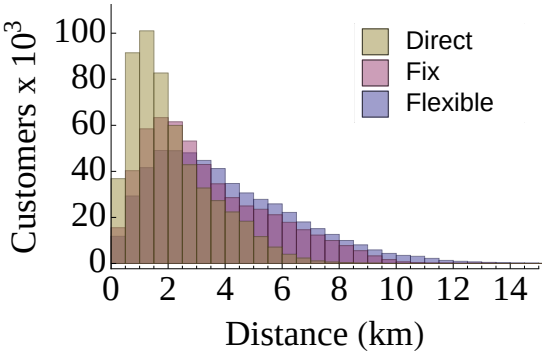


Fig. 7. Histogram of distances from customers to the facility

and fixed number of cable tree networks as well as the direct connection of the customer to the facility. As one might have expected the distribution exhibits a long tail where a vast majority of users are within the first few kms of the facility. Nearly all instances are about 5km (direct connection), 10 km (fix number of cable tree networks), and 13 km (flexible number of cable tree networks).

Table V reports total number of cable networks for each capacity type, the number of customers covered, and the average min, max, average distance from the facility (ES) to any customer. In general the fixed approach uses a fewer number of cable networks with larger capacities. Indeed this approach only uses cable networks with capacities 512, 256, 64, and 8, while the flexible approach spreads the use of all cable networks of different capacities in order to improve in the overall cable fibre cost. Additionally, as expected 512 cable networks are covering the majority of customers about 95% (Fixed) and 79% (Flexible) of the sampled population of Ireland.

In addition to the summary of the overall usage of cable networks of different capacity types, Figure 8 depicts the histogram with the distribution of the number of networks

TABLE II. CABLE TREE NETWORK USAGE STATISTICS

Type	Alg	Cable Network	Customers	Min	Max	Avg
512	Fix	1106.9	519803.0	0.32	8.39	3.91
	Flex	979.4	432066.8	0.36	8.04	3.89
256	Fix	147.0	32625.5	0.39	11.19	5.54
	Flex	569.3	125031.2	0.42	10.4	5.28
128	Fix	15.0	1660.2	0.53	11.01	4.88
	Flex	132.4	13564.7	0.49	7.54	3.87
64	Fix	1.0	53.0	0.29	4.60	2.17
	Flex	44.1	2273.0	0.45	3.71	2.08
32	Fix	–	–	–	–	–
	Flex	33.3	792.2	0.34	1.93	1.13
16	Fix	–	–	–	–	–
	Flex	22.3	277.4	0.42	2.23	1.26
8	Fix	2.0	11.0	2.59	2.95	2.75
	Flex	31.5	201.4	0.43	1.35	0.83
4	Fix	–	–	–	–	–
	Flex	27.4	90.5	0.23	0.48	0.36
2	Fix	–	–	–	–	–
	Flex	25.7	51.5	0.29	0.51	0.40
1	Fix	–	–	–	–	–
	Flex	34.6	34.6	0.20	0.20	0.20

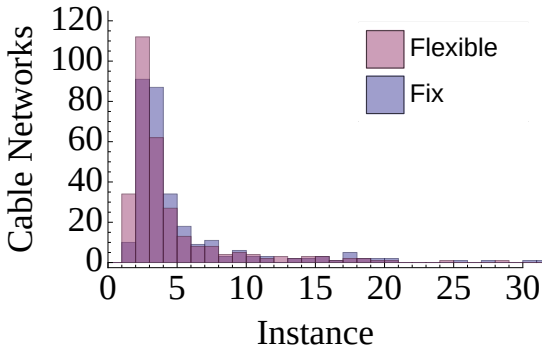


Fig. 8. Histogram of number of cable tree networks per facility

TABLE III. PERFORMANCE SUMMARY, TOTAL NUMBER OF CABLE TREE NETWORKS AND AVERAGE CABLE FIBRE COST

Algorithm	#Cable Networks	Cost
Direct	5436797.0	3863.6
Fix	**1271.9**	241.6
Flexible	1877.7	**161.3**

per instance. As expected since a majority of customers are grouped into individual exchange sites (i.e., instances), the majority of the cable networks are required to cover about 30 instances. It is worth noticing that the remaining 270 instances require a very low number of cable networks and for illustrative proposes those cable networks were removed in the figure.

In Table III we report a performance summary of the average performance of the fix and flexible approaches. In this table, we observe the trade-off between minimising number of cable networks and their cable cost in terms of distance. In general, we expect that a flexible approach would provide a better global cable cost and the fix approach would be the best one w.r.t. the number of cable networks. In these experiments we observed that the fixed approach reports a 32% fewer number of tree networks at a cost of 33% increasing in the total cable cost w.r.t. the flexible approach. As expected the direct connection of customer to the exchange site reports the worse results for both objectives.

VI. CONCLUSIONS AND FUTURE WORK

We have presented an efficient local search algorithm for solving Distance and Capacity Bounded Network Design Problems. We presented two novel move operators along with their complexities and incremental evaluation of the neighborhood and the objective function. The effectiveness of our approach is demonstrated by experimenting with a set of problem instances taken from real-world networks form Ireland.

In the future we would like to experiment with the option of optional nodes, since this is a requirement in several applications, and also plan to experiment with instances for bigger countries such as Italy and the UK. Finally, we also plan to use the proposed framework to tackle other problems in the context of the multiple travelling salesman and vehicle routing problems.

REFERENCES

[1] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

[2] R. Matai, S. P. Singh, and M. L. Mittal, "Traveling salesman problem: An overview of applications, formulations, and solution approaches," *Traveling Salesman Problem, Theory and Applications*, pp. 1–24, 2010.

[3] S. N. Kumar and R. Panneerselvam, "A survey on the vehicle routing problem and its variants," *Intelligent Information Management*, vol. 4, p. 66, 2012.

[4] L. C. Yeun, W. R. Ismail, K. Omar, and M. Zirour, "Vehicle routing problem: models and solutions," *Journal of Quality Measurement and Analysis*, vol. 4, no. 1, pp. 205–218, 2008.

[5] J. Oh, I. Pyo, and M. Pedram, "Constructing minimal spanning/steiner trees with bounded path length," *Integration*, vol. 22, no. 1-2, pp. 137–163, 1997.

[6] D. B. Payne, "FTTP deployment options and economic challenges," in *Proceedings of the 36th European Conference and Exhibition on Optical Communication (ECOC 2009)*, 2009.

[7] M. Ruffini, L. Wosinska, M. Achouche, J. Chen, N. J. Doran, F. Farjady, J. Montalvo, P. Ossieur, B. O'Sullivan, N. Parsons, T. Pfeiffer, X.-Z. Qiu, C. Raack, H. Rohde, M. Schiano, P. D. Townsend, R. Wessäly, X. Yin, and D. B. Payne, "Discus: an end-to-end solution for ubiquitous broadband optical access," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 24–56, 2014.

[8] T. F. Noronha, A. C. Santos, and C. C. Ribeiro, "Constraint programming for the diameter constrained minimum spanning tree problem," *Electronic Notes in Discrete Mathematics*, vol. 30, pp. 93–98, 2008.

[9] N. Beldiceanu, P. Flener, and X. Lorca, "Combining tree partitioning, precedence, and incomparability constraints," *Constraints*, vol. 13, no. 4, pp. 459–489, 2008.

[10] H. Hoos and T. Stützle, *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann, 2005.

[11] P. V. Hentenryck and L. Michel, *Constraint-based local search*. The MIT Press, 2009.

[12] P. Q. Dung, Y. Deville, and P. V. Hentenryck, "Constraint-based local search for constrained optimum paths problems," in *CPAIOR*, 2010, pp. 267–281.

[13] M. Ruthmair and G. R. Raidl, "A kruskal-based heuristic for the rooted delay-constrained minimum spanning tree problem," in *EUROCAST*, ser. Lecture Notes in Computer Science, R. Moreno-Díaz, F. Pichler, and A. Quesada-Arencibia, Eds., vol. 5717. Springer, 2009, pp. 713–720.

[14] H. F. Salama, D. S. Reeves, and Y. Viniotis, "The delay-constrained minimum spanning tree problem," in *ISCC*, 1997, pp. 699–703.