A Constraint-based Local Search for Designing Tree Networks with Distance and Disjoint Constraints

Alejandro Arbelaez, Deepak Mehta, Barry O'Sullivan, 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*—In many network design problems clients are required to be connected to a facility under path-length constraints and budget limits. Each facility is associated with a tree network where the root is the facility itself and the remaining nodes of the tree are its clients. An inherent feature of these networks is that they are vulnerable to a failure. Therefore, it is often important to provide some resiliency in the network. We focus on a problem where we want to ensure that all clients are connected to two facilities so that if one facility fails then all clients can still be served by another facility. Optionally, one might require that each client is resilient to a single link or node failure by enforcing that the paths used to connect a client to its two facilities are either edge-disjoint or node-disjoint respectively. In this paper we use local search to evaluate the trade-off between cost versus resiliency and coverage versus resiliency for a real-world problem in the field of optical networks.**

I. Introduction

Many network design problems arising in areas as diverse as VLSI circuit design, QoS routing, traffic engineering, and computational sustainability require clients to be connected to a facility under path-length constraints and budget limits. Here the length of the path can be interpreted as distance, delay, signal loss, etc. For example, in a multicast communication setting where a single node is broadcasting to a set of clients, it is important to restrict the path delays from the server to each client. In Long-Reach Passive Optical Networks (LR-PON) a Metro-Core node (MN) is connected to the set of Local-Exchange sites (LEs) sites via optical fibres, the length of the fibre between a LE and its MN is bounded due to signal loss, and the goal is to minimise the cost resulting from the total length of fibres [1]. In VLSI circuit design path delay is a function of maximum interconnection path length while power consumption is a function of the total interconnection length [2]. In package shipment service guarantee constraints are expressed as restrictions on total travel time from an origin to a destination, and the organisation wants to minimise the transportation costs [3]. In wildlife conservation, which is an application from computational sustainability [4], the landscape connectivity is key to resilient wildlife populations in an increasingly fragmented habitat matrix where landscape connectivity is a function of the length of the path in terms of landscape resistance to animal movement.

In many of these network design problems each facility and its clients can be modelled as Rooted Distance-Constrained Minimum Spanning-Tree Problem (RDCMST) [2] which is NP-hard. The objective is to find a minimum cost spanning tree with the additional constraint that the length of the path from a specified root-node (or facility) to any other node (client) must not exceed a given threshold.

Many networks are complex systems that are vulnerable to a failure. A major fault occurrence would be a complete failure of the facility which would affect all the clients connected to the facility. Therefore it is important to provide resiliency in the network. In this paper we focus on the networks where all clients are required to be connected to two facilities so that whenever a single facility fails all clients are still connected to another facility. Given an association between a given set of facilities and their clients, this problem can be solved by solving the RDCMST problem for each facility independently by designing a tree for each facility. The solution is a set of trees, each facility is associated with one unique tree and each client appears in two trees associated with its two facilities.

Depending on the requirements of the network one can provide different levels of resiliency and in general there is a tradeoff between the cost of the network design and the level of required resiliency. If each client is connected to two facilities via two node-disjoint (respectively edge-disjoint) paths then each client is resilient to any node (respectively link) failure. We define this problem as the Rooted Distance-Constrained Minimum Spanning-Trees Problem with Node-disjoint Constraints (RDCMSTN) and the Rooted Distance-Constrained Minimum Spanning-Trees Problem with Edge-disjoint Constraints (RDCMSTE) respectively. Notice that enforcing node-disjointness is stronger than enforcing edge-disjointness. Both are resilient to any single link failure, but only node-disjointness can guarantee resistance to any single node failure. Nevertheless, it might be more expensive to enforce node-disjointness.

Given a set of facilities and a set of clients such that each client is associated with two facilities, the problem is to find a set of distance-constrained spanning trees rooted from each facility with minimum total cost. Furthermore, each client can be connected to its two facilities via two node-disjoint paths or edge-disjoint paths. In general, RDCMSTN (resp. RDCMSTE) is more complex than RDCMST because it not only involve finding a set of distance-constrained spanning trees but it also required that the two paths between any client and its two facilities in these trees must be node (resp. edge) disjoint.

We present a general mixed integer programming model for solving RDCMST, RDCMSTE, and RDCMSTN problems. Previous works on RDCMST [5], [6] have focused on dedicated algorithms which are hard to extend for solving RDCMSTN and RDCMSTE. We therefore use a constraint-based local search algorithm for solving these problems [7] which can easily be extended to apply widely. We use an incremental way of maintaining information required to checking distance, node-disjoint and edge-disjoint constraints efficiently and an

algorithm to move from one solution to another efficiently in the context of local search. Notice that our local search algorithm is able to solve all RDCMST, RDCMSTE and RDCMSTN problems. The effectiveness of our approach is demonstrated by experimenting with a set of very large size instances taken from a real-world problem arising in the domain of optical network. More precisely, the instances correspond to the optical nework designs for three countries: Ireland, the UK, and Italy. We investigate the impact of coverage on the cost of resiliency and the trade-off between the cost and the desired level of resiliency. EmpiricEmpiricalal results show that our approach is scalable and has a good anytime behaviour.

This paper is structured as follows. Section II presents a formal description of the problem and the complexity. Section III presents a constraint optimisation formulation of RDCMST(N/E) problem. Section IV presents the constraint-based local search approach for solving this problem. Section V describes the real-world problem whose instances are used to evaluate the developed approach. Section VI reports on the experimental results, and Section VII concludes with some research perspectives.

II. Formal Specification and Complexity

Let $\mathcal{M}$ be the set of facilities. Let $\mathcal{E}$ be the set of all clients. Let $E_i \subseteq \mathcal{E}$ be the set of clients that are associated with facility $m_i \in \mathcal{M}$. Notice that we assume that each client $e_k \in \mathcal{E}$ is associated with two facilities $\{m_i, m_j\} \subseteq \mathcal{M}$ such that $e_k \in E_i$ and $e_k \in E_j$. We use $\mathcal{N}$ to denote the set of nodes, which is equal to $\mathcal{M} \cup \mathcal{E}$. We also use $N_i = E_i \cup \{m_i\} \subseteq \mathcal{N}$ to denote the set of nodes in T_i.

Given a directed graph G with a set of nodes, and a set of edges, let T_i be a spanning tree of G rooted at m_i, T_i is a subgraph g of G such that g contains a directed path from m_i to any client and contains no cycles. We use λ to denote the maximum path-length from a facility to any of its clients.s

Rooted Distance-Constrained Minimum Spanning-Tree Problem (RDCMST). Given a facility $m_i \in \mathcal{M}$, the set of clients E_i, a set of feasible links $L_i \subseteq N_i^2$, two real numbers, a cost c_l and a distance d_l for each link $l \in L_i$, and a real number λ, the RDCMST is to find a spanning tree T_i with minimum total cost such that the length of the path from the facility m_i to any $e_j \in E_i$ is not greater than λ.

Rooted Distance-Constrained Minimum Spanning-Trees Problem with Node-Disjoint Constraints (RDCMSTN). Given a set of facilities $\mathcal{M}$, a set of clients $\mathcal{E}$, a set of feasible links $\mathcal{L} \subseteq \mathcal{N}^2$, two real numbers, a cost c_l and a distance d_l for each link $l \in \mathcal{L}$, an association of clients with two facilities $\pi : \mathcal{E} \rightarrow \mathcal{M}^2$, and a real number λ, the RDCMSTN is to find a spanning tree T_i for each facility m_i such that:

1) The length of the path from the facility m_i to any of its clients is not greater than λ.
2) For each client e_k, the two paths connecting e_k to m_i and m_j, where $\pi(e_k) = \langle m_i, m_j \rangle$, are node disjoint.
3) The sum of the costs of the edges in all the spanning trees is minimum.

RDCMSTE can be defined by simply considering the edge-disjoint constraint instead of node-disjoint in the second condition of the definition of RDCMSTN.

Figure 1 shows an example with two facilities m_1 and m_2 and $\mathcal{E}$={a, b, c, d, e, f}, black and grey color edges are used to show the trees corresponding to the facilities m_1 and m_2 respectively, and the value for λ is 12. In this figure we depict three scenarios satisfying the length constraint, however, each scenario shows different level of resiliency. Dashed lines indicate edges leading to a conflict, i.e., violating node or edge disjointness, and grey nodes indicate the conflicting node when violating node-disjointness. The total solution cost for Figure 1(a) is 46. The indicated solution satisfies the length constraint (i.e., the distance from the facilities to any node is less or equal to λ=12) and it also satisfies node-disjoint constraints and consequently edge-disjoint constraints. Here we observe that failure of any single link or node would not affect the remaining clients. Figure 1(b) does not satisfy node-disjoint constraints but edge-disjoint constraints. Here we observe that a single node failure could disconnect one or more clients from the facilities m_1 and m_2, i.e., f would be disconnected from m_1 and m_2 whenever a fails because f relies on the link $\langle$a, e$\rangle$ for connecting to m_1 and $\langle$a, f$\rangle$ for connecting to m_2. Nevertheless, the solution is resilient to any single facility or single link failure. Figure 1(c) shows a solution that violates both node and edge disjoint constraints. In this example the link $\langle e, f \rangle$ appears in both trees and the path from f is neither edge-disjoint nor node-disjoint but it is resilient to a failure of a single facility.

Complexity. RDCMSTN involves finding a rooted distance-bounded spanning tree for every facility whose total cost is minimum. This problem is known to be NP-complete [2]. The same would hold for RDCMSTE.

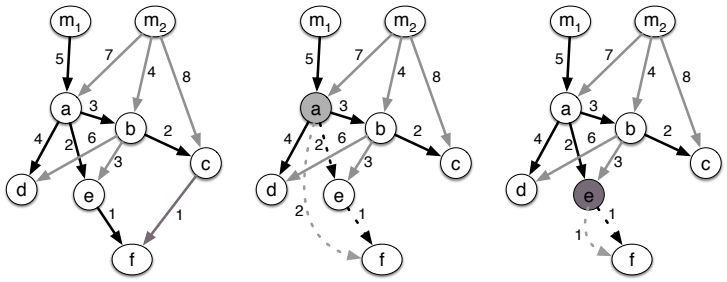
III. Constraint Optimisation Formulation

In this section we present a constraint optimisation formulation of RDCMST(N/E). Without loss of generality, in the formulation of the problem we assume full connectivity in the graph associated with a facility and its clients, and non-existing links can be added in $\mathcal{L}$ by associating them with very large distances (with respect to λ). The model only relies on integer variables and all constraints are linear, and therefore it can be encoded into existing MIP solvers.

Variables.

- Let x^i_{jk} be a Boolean variable that denotes whether an arc between clients $e_j \in E_i$ and $e_k \in E_i$ of facility $m_i \in \mathcal{M}$ is selected or not. Each arc (i, j) has an associated cost c_{ij}.[1]
- Let y^i_{jk} be a Boolean variable that denotes whether the node corresponding to a client j is in the path from client k to facility $i \in \mathcal{M}$.
- Let f^i_j be a variable that denotes the length of the path from the facility i to its client j.

[1]We assume that the cost is symmetrical, i.e., the c_{ij}=c_{ji}.



(a) Node and Edge disjoint (b) Not Node but Edge Disjoint (c) Neither Node nor Edge Disjoint

Fig. 1. Two rooted distance-constrained minimum spanning trees with and without node and edge disjoint constraints

- Let b^i_j be a variable that denotes the maximum length of the path from the client j to any client in the tree of facility i that is acting as a leaf-node.

We remark that the partial order enforced by f or b help to rule out cycles in the solution.

Constraints for RDCMST(N/E). Each client associated with each facility has only one incoming arc:

$$\forall_{m_i \in \mathcal{M}} \forall_{e_k \in E_i} : \quad \sum_{e_j \in N_i} x^i_{jk} = 1 \tag{1}$$

Each facility is connected to at least one of its clients:

$$\forall_{m_i \in \mathcal{M}} : \quad \sum_{e_j \in E_i} x^i_{ij} \geqslant 1 \tag{2}$$

The total number of arcs in any tree T_i is equal to $|E_i|$. This is a redundant constraint but its inclusion can improve the quality of LP relaxation.

$$\forall_{m_i \in \mathcal{M}} : \quad \sum_{e_j \in N_i} \sum_{e_k \in E_i, e_j \neq e_k} x^i_{jk} = |E_i| \tag{3}$$

If an arc from $e_j \in E_i$ to $e_k \in E_i$ is selected then the length of the path from m_i to e_k is equal to the sum of the lengths from m_i to e_j plus the length between e_j and e_k:

$$\forall_{m_i \in \mathcal{M}} \forall_{\{e_j, e_k\} \in N_i} : \quad x^i_{jk} = 1 \Rightarrow f^i_k = f^i_j + d_{jk} \tag{4}$$

If an arc from $e_j \in E_i$ to $e_k \in E_i$ is selected then the length of the path from e_j to any leaf-node through e_k is greater than or equal to the sum of the lengths from e_k to any of its leaf-node plus the length between e_j and e_k:

$$\forall_{m_i \in \mathcal{M}} \forall_{\{e_j, e_k\} \in N_i} : \quad x^i_{jk} = 1 \Rightarrow b^i_j \geqslant b^i_k + d_{jk} \tag{5}$$

At any node in the tree the length of the path from a facility m_i to a client e_j and the length of the path from e_j to the farthest client should be less than or equal to λ:

$$\forall_{m_i \in \mathcal{M}} \forall_{e_j \in N_i} : \quad f^i_j + b^i_j \leqslant \lambda \tag{6}$$

Constraints (1)-(6) are required for solving RDCMST(N/E) problem. In addition to these we also need to enforce constraints (7)-(9) for enforcing node-disjointness and constraint (10) for enforcing edge-disjointness.[2]

[2]We use the Big-M method to translate implications to linear constraints

Constraints for RDCMSTN. If an arc from $e_j \in E_i$ to $e_k \in E_i$ is selected then it means that the node j is used by node k to reach to the facility i:

$$\forall_{m_i \in \mathcal{M}} \forall_{\{e_j, e_k\} \in N_i} : \quad x^i_{jk} \Rightarrow y^i_{jk} \tag{7}$$

If a node j is in the path between the facility i and k and k is in the path between the facility i and l then it means j is also in the path between the facility i and l:

$$\forall_{m_i \in \mathcal{M}} \forall_{\{e_j, e_k, e_l\} \in N_i} : \quad y^i_{jk} \wedge y^i_{kl} \Rightarrow y^i_{jl} \tag{8}$$

If m_k and $m_{k'}$ are the facilities of the client k, then any client j can only appear in at most one path: Therefore, we enforce the following constraint:

$$\forall_{\{m_i, m_{i'}\} \subseteq \mathcal{M}} \forall_{\{e_j, e_k\} \in E_i \cap E_{i'}} : y^i_{jk} + y^{i'}_{jk} \leqslant 1 \tag{9}$$

Constraints for RDCMSTE. We would like to remark that replacing Constraints 7-9 with the following would result in enforcing edge-disjointness between the two paths:

$$\forall_{\{m_i, m_{i'}\} \subseteq \mathcal{M}} \forall_{\{e_j, e_k\} \in E_i \cap E_{i'}} : x^i_{jk} + x^{i'}_{jk} \leqslant 1 \tag{10}$$

The above constraint enforces that if m_i and $m_{i'}$ are the facilities of the client j, and if there exists any path in the subnetwork associated with the facility i that includes the arc $\langle e_j, e_k \rangle$, then facility i' cannot use the same arc.

Objective. The objective is to minimize the total cost:

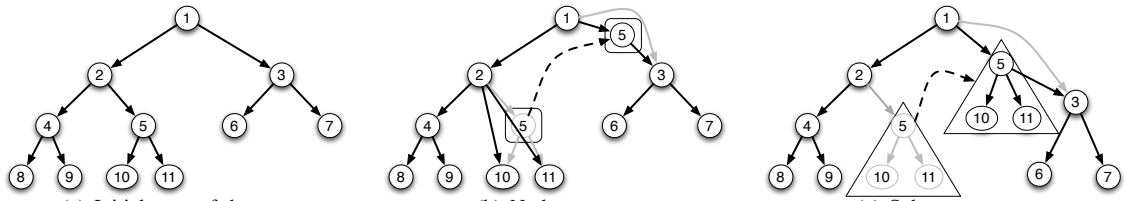
$$\min \sum_{m_i \in \mathcal{M}} \sum_{\{e_j, e_k\} \subseteq E_i} c_{jk} \cdot x^i_{jk} \tag{11}$$

IV. Constraint-based Local Search

Constraint-based local search has been recently introduced as an alternative solution to efficiently tackle many network design problems [7], [8] ranging from telecommunications to transportations, and VLSI circuit design.

The local search (LS) algorithm starts with an initial solution and iteratively improves the solution by performing small changes. A move operator guides the algorithm towards better solutions. [7] defines the *node* and *subtree* operator.

Node operator. (Figure 2(b)) moves a given node e_i from the current location to another in the tree. As a result of this, all successors of e_i will be directly connected to the predecessor node of e_i. e_i can be placed as a new successor for another node or in the middle of an existing arc in the tree.



(a) Initial state of the tree (b) Node operator (c) Subtree operator

Fig. 2. Move operators

Subtree operator. (Figure 2(c)) moves a given node e_i and the subtree emanating from e_i from the current location to another in the tree. As a result of this, the predecessor of e_i is not connected to e_i, and all successors of e_i are still directly connected to e_i. e_i can be placed as a new successor for another node or in the middle of an existing arc.

Empirical results suggest that the subtree operator outperforms the node operator [7]. In the remaining part of the paper we therefore focus on using the subtree operator when enforcing node-disjoint and edge-disjoint constraints.

We recall that the LS algorithm to provide edge/node disjoint solutions has been previously described in [7] and [9]. In this work we extend this framework to evaluate the impact of varying the population coverage and mixed levels of disjointness in the global cost of the solution.

A. Distance/Node-disjoint/Edge-disjoint Constraints

The LS algorithm uses the move-operators in two different phases. In the intensification phase a subtree is selected and it is moved to another location that provides the maximal improvement in the objective function. In the perturbation phase both a subtree and its new location are selected randomly in order to diversify the search by adding noise to the current incumbent solution. In order to complete the intensification and diversification phases, the sub-tree operator requires four main functions: *removing* a subtree ; checking whether a given solution is *feasible* or not; *inserting* a subtree; finding the *best location* of a subtree in the current tree network. The basic idea of the LS algorithm in the intensification phase is described as follows:

1) Randomly select a node (e_i) and facility (m_i) from the current solution;
2) Delete the emanating subtree of e_i in m_i;
3) Identify the best location, i.e., a new predecessor e_p and a potential successor e_s, for e_i in m_j satisfying all constraints;
4) Insert e_i as a new successor of e_p, and if needed, add e_s as a new predecessor of e_p.

During the diversification phase the third step performs a random selection to compute the location of e_i in the tree. additionally, the algorithm is equipped with an incremental way of maintaining information related to checking constraints and computing the cost of the assignment. For the length constraint it is necessary to maintain for each node e_j its f^i_j, i.e., the length of the path from a facility i to its client (or node in the tree) j, and b^i_j, i.e., the length of the path emanating from e_j down to the farthest leaf from e_j. These two lengths are updated after removing and inserting a subtree in the tree. For enforcing the edge-disjoint constraint between the two paths of a client we need to maintain the predecessor of each client in each tree associated with each facility and enforce that the two predecessors of each client must be different in the two trees associated with its two facilities. For enforcing node-disjoint constraint between the paths of each client to its two facilities it is necessary to maintain all the nodes occurring in each path except source and target. This is done by maintaining the transitive graph and ensure that there is no common node.

We now describe the complexities of the remove, insert, feasibility check and best location operations performed during search w.r.t. distance, edge-disjoint and node-disjoint constraints respectively

Remove. To remove a subtree emanating from a client e_j of the tree associated with the facility m_i it is necessary to update $b^i_{j'}$ for all the nodes j' in the path from the facility m_i to the predecessor of the client e_j. Thus, the complexity of this operation for the distance-constraint is linear w.r.t. the number of nodes in the tree. For edge-disjoint constraint the complexity is constant as it would require to update the predecessor of the node e_j. For the node-disjoint constraint, the transitive graph is updated. This is done by removing all the transitive links between any node in the path from the facility m_i down to the client e_j and all the nodes occurring in the subtree starting at node e_j. The time-complexity for the node-disjoint constraint is therefore quadratic.

Insert. The complexity of inserting a subtree as a child of node e_j is the same as the complexity of the remove operation. For the node-disjoint constraint, it involves adding the transitive links between each node in the path from the facility down to e_j and each node in the subtree emanated from e_j. For the edge-disjoint constraint the operation involves updating the predecessor of the node e_j. For the distance-constraint this operation requires to update $f^i_{j'}$ for all the nodes j' in the subtree emanating at e_j, and $b^i_{j'}$ for all the nodes j' occurring in the path from the facility node down to the node e_j.

Feasibility. To verify the consistency of the node-disjoint constraint we need to check that any node in the path from the facility down to the potential predecessor of e_j is not transitively connected to any node of the subtree in any other tree corresponding to other facilities. This is done by checking the occurrence of the links corresponding to the pairs of nodes occurring in the transitive graph. As the number of links can be quadratic, the time complexity is quadratic. Additionally, if the node e_j is breaking an existing arc $\langle e_p, e_q \rangle$, it is necessary to check that e_j is not transitively connected to any node in the

TABLE I. COMPLEXITIES OF SUBTREE OPERATOR FOR DISTANCE, EDGE-DISJOINT AND NODE-DISJOINT CONSTRAINTS

	Distance	Edge-disjoint	Node-disjoint	Amortised Node-disjoint
Remove	$O(n)$	$O(1)$	$O(n^2)$	$O(n^2)$
Feasibility	$O(1)$	$O(1)$	$O(n^2)$	$O(n)$
Insert	$O(n)$	$O(1)$	$O(n^2)$	$O(n^2)$
Best move	$O(n)$	$O(n)$	$O(n^3)$	$O(n^2)$

subtree emanating from e_q. The complexity of checking edge-disjoipredecessornt constraint is constant as it involves only checking the predecessor node of e_j in the trees associated with its two facilities. The complexity of checking distance constraint is constant as it involves only checking if the sum of f_j^i and b_j^i is less than λ.

Best Location. Selecting the best location for a given subtree involves traversing all the nodes of the tree associated with the facility and selecting the one that maximises the reduction in the cost. Therefore, the time complexities of finding the best location with respect to distance and edge-disjoint constraint are linear in the number of nodes of a given tree as their complexities of feasibility checking is constant, The time complexity of finding the best location with respect to node-disjoint constraint is cubic in the number of nodes of a given tree as the complexity of feasibility is quadratic.

The complexities for feasibility checking and finding the best location as described above for the node-disjoint constraint do not make any assumption on the order in which different locations of the tree are explored. Here by locations we mean the set of nodes which can be valid parent candidates of the subtree starting from e_j and the set of links $\langle e_p, e_q \rangle$ where e_p is a valid parent candidate and e_q is a valid child candidate of e_j. It can be shown that the amortised complexities can be reduced to quadratic if the nodes in the tree are traversed in a top-down (e.g., depth-first) manner and the links are traversed in a bottom-up manner [9]. The basic idea is to first explore the successors of a node e_p in a top-down manner only when there is no existing link in the transitive graph between e_p and any node of the subtree in a top-down manner. For exploring links, the basic idea is to explore a link $\langle e_p, e_q \rangle$ only if e_p is a valid parent candidate for the node e_j of a given subtree and there is no link from e_j to e_q in the transitive graph. If $\langle e_j, e_q \rangle$ is exists in the transitive graph then subsequently all the links involved in the path from the facility to e_p would not be feasible either. Thus, the amortised time complexity of feasibility checking is linear and, consequently, the amortised complexity of *BestLocation* is quadratic.

B. Related Work

[8] defines the *arc* operator. The algorithm moves a given arc from one location to another in the tree. This operator has been used in the context of the routing and wavelength assignment problem [10] and to find edge disjoint paths in a given graph [11]. We remark that `node` and `subtree` move-operators are specifically designed for tree structures whereas the arc operator is more general and it can be applied to more general graphs. Nevertheless, the former are more efficient when we are dealing with trees and the latter is more expensive. We limit our attention to using the subtree move-operator as it provides a better time complexity than that of the arc operator, i.e., $O(n)$ vs. $O(n^3)$ when considering the distance constraint.

Other related work includes [6] where the authors proposed a dedicated algorithm for the RDCMST. The algorithm removes an arc in the tree, and find the cheapest arc to reconnect the two sub-trees. Nodes are sequentially added in the tree using Prim's algorithm, and nodes violating the length constraint are included in the tree using a pre-computed route from the root to any node. This framework is difficult to extend with side constraints, and therefore the algorithms cannot be extended to solve our problem with node/edge disjoint paths. For instance, the operators rely on a pre-computed alternative route from the root to any node. However that route might violate disjoint constraint.

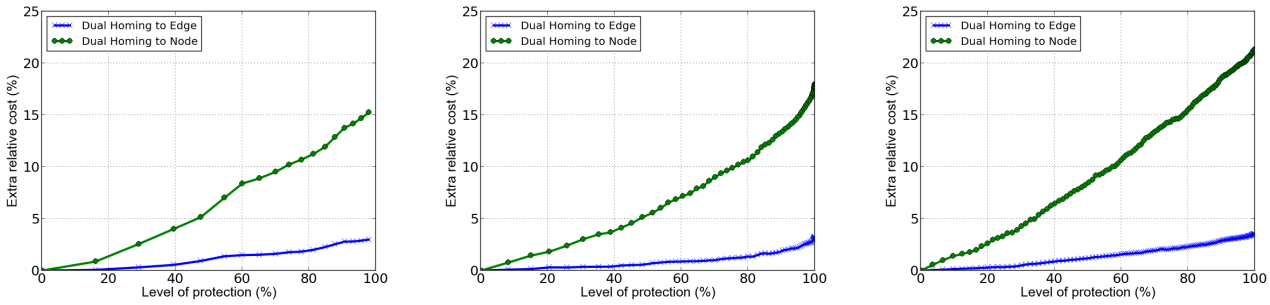
V. APPLICATION: LONG-REACH PASSIVE OPTICAL NETWORK DESIGN

We now describe a real-world application whose instances are used for evaluating our approach. Long-Reach Passive Optical Networks (LR-PONs) provides an economically viable solution for fibre-to-the-home network architectures [1]. In LR-PON fibres are distributed from the Metro-Nodes (MNs) to the local-exchange sites (LEs) through cables that forms a tree distribution network. The distance between a LE and its MN can be up to 90 km. A major fault occurrence in LR-PON would be a complete failure of the MN, which could affect tens of thousands of customers. The dual homing protection mechanism for LR-PON enables customers to be connected to two MNs, so that whenever a single MN fails all customers are still connected to a back-up node.

In this paper we focus on optimising connection cost of dual homing protection mechanism for LR-PON. Broadly speaking, there are two protection strategies for dual homing. One approach is protecting links in the tree distribution network, e.g., a fibre cut, amplifier failure, or other equipment can fail. This strategy, known as edge-disjoint solution, allows switching to an alternative path whenever a link in the distribution network fails. Alternatively, protection can be provided at the node level with node-disjoint paths between MNs and LEs, when an entire LE fails and all adjacent links to the node will be affected. This strategy, known as node-disjoint solution, provides a stronger protection mechanism and allows switching to an alternative path whenever a node fails. The selection of one protection mechanism or another is up to the network operator, generally higher protection means higher deployment cost.

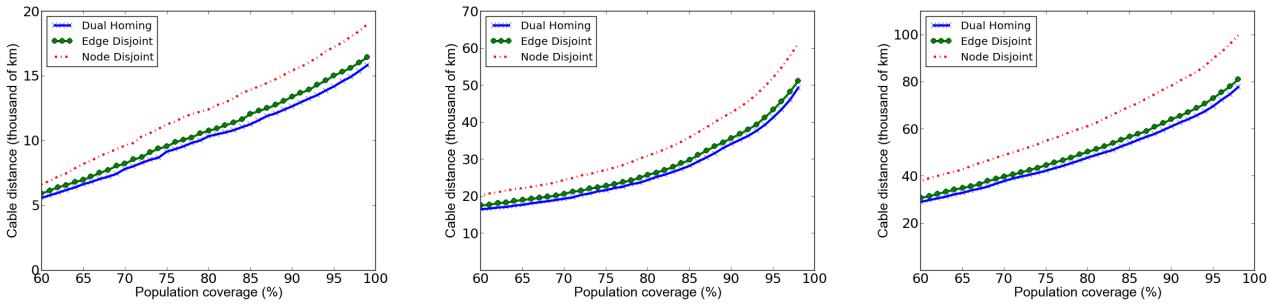
We aim at routing cables such that there are two (node/edge) disjoint paths from each LE to its two MNs, the length of each path must be below a given threshold and the total cable length required for connecting each LE to two MNs is minimised. Notice that here metro-nodes are facilities and exchange-sites are clients.

We study three real networks from three EU countries: Ireland with 1121 LEs, the UK with 5393 LEs, and Italy with 10708 LEs. For each dataset we use [12] to identify the position of the MNs and computed four instances for each country. Ireland with 18, 20, 22, and 24 MNs; the UK with 75, 80, 85, and 90 MNs; and Italy with 140, 150, 160, and 170 MNs. We use λ=90km. For each LE we use the closest MN for the primary connection and the second closest MN for the backup connection. Additionally, customers are associated



(a) Ireland with 18 MNs (b) UK with 75 MNs (c) Italy with 140 MNs

Fig. 3. Mixed protection levels, moving from a fully dual-homed solution to a edge/node one



(a) Ireland with 18 MNs (b) UK with 75 MNs (c) Italy with 140 MNs

Fig. 4. Trade-off between limiting customer coverage and total cable fibre of the network

with their closest LE. The local search algorithm starts with the trivial solution which consists in connecting all LEs (clients) to their MNs (facility) directly.

A. Mixed protection and population coverage levels

In order to evaluate the cost of mixed levels of protection and population coverage, we start by computing a solution with the weakest protection level, i.e., dual-homing, and compute solutions increasing the level of node/edge disjointness, to this end use the following methodology:

Disjointness versus cost

1) Compute the best solution using LS with dual-homing protection;
2) Sort LEs based based of customers in increasing order;
3) Systematically provide edge/node disjointness each LE from the top of the list and compute the impact in the global cost.

Population coverage versus cost

1) Compute the best solution using LS with a given protection level (i.e., dual-homing, node/edge disjointness) using all LEs;
2) Sort LEs based on the number of customers in increasing order;
3) Systematically remove LEs from the top of the list and compute the impact of removing these LEs in the global cost.

VI. Empirical Evaluation

In this section we evaluate the cost of the network as the total required fibre cost in km for a particular solution. To evaluate the trade-off between (a) cost and protection and (b) customer coverage and cost in the network design, we study the following three scenarios: (1) cost when 100% of the customers are dual-homed, edge and node disjoint protected; (2) cost of mixed dual-homed with edge and node disjoint protection; (3) cost of dual-homed, edge and node disjoint protections with a limited population coverage. The first scenario can be seen as a reference baseline with fully protected networks. The second scenario intends to study the impact in the cost of mixed levels of protection, i.e., starting with a fully dual-homed solution and systematically protect customers with edge/node-disjoint paths until reaching a fully edge/node-disjoint solution. In the third scenario we evaluate the cost of each protection mechanism by limiting the service coverage, e.g., providing a fully edge-disjoint solution to 70% of the population and the remaining 30% is dual-homed protected.

Table II reports the solution for dual homing, edge and node disjoint solutions with different MN configurations. As expected, the total required fibre cable decreases as we increase the number of MNs, e.g., a dual-homed solution with 18 MNs requires 16625 km of cable while 24 MNs only requires 15700 km. Additionally, it is worth noting that the extra cost of a fully edge-disjoint solution w.r.t. dual homing is on average 2.9% for Ireland, 3.0% for the UK, and 3.4% for Italy, while the strongest protection (i.e., node disjoint) increases the cost w.r.t. dual homing on average 15.4% for Ireland, 17.8% for the UK, and 21.0% for Italy. Additionally, we would like to

TABLE II. SOLUTION COST FOR EACH PROTECTION MECHANISM

	|MN|	Dual Homing	Edge Disj.	Node Disj
	18	16625	17161	19736
Ireland	20	16421	16885	19508
$\|E\|$ = 1121	22	16251	16778	19156
	24	15700	16179	18482
	75	64347	66521	78544
UK	80	63779	65579	77269
$\|E\|$ = 5393	85	62366	64357	76003
	90	60941	63001	74146
	140	88071	91265	111885
Italy	150	86701	89780	110178
$\|E\|$ = 10708	160	86736	89891	109906
	170	85728	88865	108122

recall that the LS greatly outperforms CPLEX with the MIP formulation described in Section III, and the solutions of the LS algorithm are very close to the optimal ones, e.g., [7] indicates that the GAP varies from 9% to 13% w.r.t. the best known lower bounds.

We now move our attention to Figure 3. In this figure we evaluate the cost of mixed levels of protection. We analyse the cost of providing edge/node-disjoint protection to a certain percentage of the customers in a given country and the remaining portion of the customers are dual-homed protected. The x-axis indicates the percentage of customers node/edge disjoint protected (resp. 100-x indicates the percentage of customers dual-homed protected), and the y-axis indicates the extra cost of the indicated protection level w.r.t. to a fully dual-homed solution.

In this figure, we observe that the extra cost of providing edge-disjoint protection to 20% of the customers is nearly negligible w.r.t. a fully dual-homed solution for the three countries (i.e., 0.13% for Ireland, 0.2% for the UK, and 0.3% for Italy), and the relative extra cost of a fully edge-disjoint solution is only up to 2.9% for Ireland, 3.0% for the UK, and 3.4% for Italy. On the other hand, the extra cost of providing node-disjoint solutions to 20% of the customers is up to 2.6% for the three countries, and extra cost slightly increases to up to 11% when providing node-disjointness to 60% of the customers, and up to 15.5% extra cost is observed when providing node-disjoint protection to 80% of the customers.

Finally, Figure 4 studies the trade-off between limited customer coverage (e.g., at-least 60% of the customers) and the required cable to provide fully dual-homed, edge/node disjoint solutions. Interestingly, limiting coverage to 80% of the customers requires a total of 12100 km (node-disjoint), 10570 km (edge-disjoint) and 10310 km (dual-homing) of fibre cable for Ireland. Overall, we have observed that when limiting coverage to 80% of the customers for Ireland we reduce the total cost (w.r.t. the solution covering all customers) to 61.2% (dual-homing), 57.3% (edge-disjoint), and 37.4% (node-disjoint); for the UK the reduction for the same percentage of customers is 166.6% (dual-homing), 157.9% (edge-disjoint), and 111.1% (node-disjoint); and for Italy the reduction for the same percentage of customers is 85.8% (dual-homing), 77.9% (edge-disjoint), and 44.7% (node-disjoint).

VII. CONCLUSIONS AND FUTURE WORK

In this paper we use an efficient constraint-based local search algorithm to study the trade-off between the cost of the network and mixed levels of protection. The extra cost of edge-disjoint protection is relatively small w.r.t. to fully dual-home solution, and as expected increasing node-disjoint support also increases the deployment cost of the network. For instance, the extra cost of mixed levels of protections ranges 0.13% to 0.3% (reps. 2.9% to 3.4%) when providing 80% of the customers with dual-homing support and the remaining 20% with edge-disjointness (resp. node-disjointness). Additionally, we also studied the impact in the cost of limiting coverage to a certain portion of the customers. It is important to highlight the reduction in the total cost of up to 166% (w.r.t. full coverage with dual-homing) when limiting coverage to 80% of the customers for the Italian network.

VIII. ACKNOWLEDGMENTS

This work was supported by DISCUS (FP7 Grant Agreement 318137), and Science Foundation Ireland (SF) Grant No. 10/CE/I1853. The Insight Centre for Data Analytics is also supported by SFI under Grant Number SFI/12/RC/2289.

REFERENCES

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

[2] 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.

[3] M. Ruthmair and G. R. Raidl, "A kruskal-based heuristic for the rooted delay-constrained minimum spanning tree problem," in *Computer Aided Systems Theory-EUROCAST 2009*. Springer, 2009, pp. 713–720.

[4] C. P. Gomes, "Computational sustainability: Computational methods for a sustainableenvironment, economy, and society," *The Bridge*, vol. 39, no. 4, pp. 5–13, 2009.

[5] M. Leitner, M. Ruthmair, and G. R. Raidl, "Stabilized branch-and-price for the rooted delay-constrained steiner tree problem." in *INOC*, ser. Lecture Notes in Computer Science, J. Pahl, T. Reiners, and S. Vo, Eds., vol. 6701. Springer, 2011, pp. 124–138.

[6] M. Ruthmair and G. R. Raidl, "Variable neighborhood search and ant colony optimization for the rooted delay-constrained minimum spanning tree problem." in *PPSN (2)*, ser. Lecture Notes in Computer Science, R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, Eds., vol. 6239. Springer, 2010, pp. 391–400.

[7] A. Arbelaez, D. Mehta, B. O'Sullivan, and L. Quesada, "A constraint-based local search for edge disjoint rooted distance-constrainted minimum spanning tree problem," in *CPAIOR'15*, pp. 31–46.

[8] Q.-D. Pham, Y. Deville, and P. V. Hentenryck, "LS(Graph): a constraint-based local search for constraint optimization on trees and paths," *Constraints*, vol. 17, no. 4, pp. 357–408, 2012.

[9] A. Arbelaez, D. Mehta, and B. O'Sullivan, "Constraint-based local search for finding node-disjoint bounded-paths in optical access networks," in *To appear in CP'15*, 2015.

[10] B. Mukherjee, *Optical WDM Networks (Optical Networks)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[11] M. Blesa and C. Blum, "Finding edge-disjoint paths in networks: An ant colony optimization algorithm," *Journal of Mathematical Modelling and Algorithms*, vol. 6, no. 3, pp. 361–391, 2007.

[12] M. Ruffini, D. Mehta, B. O'Sullivan, L. Quesada, L. Doyle, and D. B. Payne, "Deployment strategies for protected long-reach pon," *Journal of Optical Communications and Networking*, vol. 4, pp. 118–129, 2012.