Degree Constrained Minimum Spanning Hierarchies in Graphs under Non Uniform Capacity Constraints. Exact Solutions

Miklós Molnár

Degree Constrained Minimum Spanning Hierarchies in Graphs under Non Uniform Capacity Constraints. Exact Solutions

Miklós Molnár

LIRMM, University of Montpellier, CNRS, Montpellier, France
molnar@lirmm.fr

Abstract. Degree constrained spanning tree problems are known and deeply analyzed when the graph nodes are provided with limited budgets. We examine the constrained spanning problem supposing heterogeneous degree bounds on nodes representing limited momentary capacities. In an undirected graph, different positive integer upper bounds are associated with nodes to limit their degree for each visit. The exact formulation of finding the minimum cost connected spanning structure satisfying these degree constraints is the subject of our work. The capacity constrained spanning problem with homogeneous constraints has been solved using spanning hierarchies which can be different from spanning trees. Here the case with non uniform constraints is analyzed, and we present an ILP based solution to this NP hard problem. The gain of applying hierarchies is demonstrated running ILPs for hierarchies and trees.
Keywords: Graph theory, spanning problems, degree constrained minimum spanning tree, hierarchy, degree constrained minimum spanning hierarchy, non uniform constraints, exact solution, ILP

1 Introduction

The construction of Minimum Spanning Trees (MSTs) in graphs is a well known polynomially solvable problem. The interest of such a solution is for instance in the diffusion of messages in networks, finding connections in VLSIs, etc. The Degree Constrained Minimum Spanning Tree (DCMST) was introduced as a spanning tree in which the degree of the nodes is limited by an integer value B and the sum of the edges (the cost) is the minimum possible. Finding a DCMST is one of the NP-hard problems mentioned in [6]. The interest of this problem resides in diffusion, where the nodes are doted by a limited duplication's/branching possibility. When the bound is equal to 2, the problem corresponds to the minimum Hamiltonian path problem and it is known the path does not always exist. More generally, it is not always possible to find a DCSMT [2]. The resolvability and the approximation of several degree constrained spanning tree and partial spanning tree problems have been analyzed based on two criteria : the degree of the nodes and the cost in [16]. Two versions of degree constraints were considered: uniform and non uniform degree constraints. Negative results have been formulated: constant factor approximations do not exist for the analyzed cases.

In most of the work, the degree constraints express a definitive limitation of the branching possibility of nodes. This kind of limitation is not always the case. In some real cases in communication, the limited capacity of the nodes is related to an instantaneous (momentary) capacity. For example, in a switch the number of copies of an incoming message can be limited, but if the same message returns several times to the switch, a new duplication is done. To distinguish the two limitations, we talk about limited *b*udget in the case of definitive limitation and limited (momentary) *c*apacity when the limitation concerns each individual passage. Since the spanning structure (the route) can return to nodes and can pass several times by some edges, a new graph related structure different from tree was needed to describe the solution. For this, hierarchies have been proposed using graph homomorphism in [11]. In the case of momentary capacities under uniform constraints in the node set, the hierarchy based solution has been presented in [13]. The analysis of the presented NP-hard problem shows that the hierarchy based solution always exists and the minimum spanning hierarchy can be approximated. In the recent paper the analysis of the cases where the momentary capacity constraints are not uniform but heterogeneous is analyzed and we propose the exact formulation of the problem based on an ILP.

In the following sections we rapidly present the used concepts and notation, followed be the related work (*cf.* Section 3) and some properties of the spanning hierarchies to help the exact formulation (*cf.* Section 4). The ILP based computation is presented in Section 5 and the results with their analysis is in Section 6. The last section gives the conclusion and some perspectives.

2 Notations and Definitions

At first, we recall the definition of a graph related structure. The homomorphic mapping between a tree T and a graph G defines an eventually non-elementary tree in G called hierarchy [11].

Definition 1 (Hierarchy) *Let $G_E = (V, E)$ be a graph and let $T = (W, F)$ be a tree. Let $h : W \to V$ be a homomorphic function which associates a node $v \in V$ to each node $w \in W$. The application (T, h, G) defines a hierarchy in G.*

Fig. 1 illustrates the mapping and the resulted hierarchy in an undirected graph (cf. Fig. 1/a). Some nodes (namely the nodes c and d) are present several times in the hierarchy illustrated by the labeled tree in Fig. 1/b).

Hierarchies generalize trees. Trees are hierarchies without repetition of nodes (applying an injective mapping h) and consequently inherit the properties of hierarchies. Let us notice that the sub-graph of G generated by a hierarchy (its image) can contain cycles in G but the hierarchy itself preserves important properties of trees. In the following, we use the term *hierarchy* to reference the defined tree-like structure and we use the term *image of the hierarchy* for the sub-graph implicated in the original graph.

Since a node $v \in V$ can be associated with several nodes in W, a hierarchy can "return" several times to nodes and pass several times edges in G (as it may

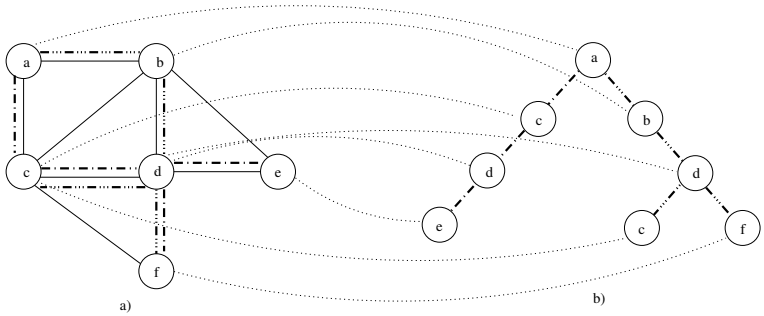

Fig. 1: Example of a hierarchy in a undirected graph

be the case in walks): it corresponds to a "non-elementary tree" in the graph G. In hierarchies some nodes are eventually branching nodes (they are the node occurrences corresponding to the branching nodes of the tree T). Hierarchies can be directed or not. In directed cases both the basic graph and the hierarchy (the tree associated with nodes) are directed and the mapping function preserves the direction of arcs. More details can be found in [11].

Two definitions follows before the problem formulation. Let $H_1 = (T_1, h_1, G)$ and $H_2 = (T_2, h_2, G)$ two hierarchies related to the same graph.

Definition 2 (Independence) *H_1 and H_2 are independent each from other if no node of G is associated with T_1 and T_2.*

In other words, H_1 and H_2 do not cover any common node.

Sub-hierarchies of a hierarchy can be independent or not.

Definition 3 (Inclusion) *H_1 includes H_2 if T_1 includes T_2 including labels: the common nodes reference the same nodes in G.*

In other words, H_2 is a sub-hierarchy of H_1.

In the paper we use the following notation.

$G = (V, E)$	: an undirected graph (to span)
V	: the set of nodes in G
E	: the set of edges in G
$\{n, m\} \in E$	: an edge between nodes n and m
$G_A = (V, A)$	: a directed graph equivalent to G in which two arcs $\overrightarrow{(a, b)}$ and $\overrightarrow{(b, a)} \in A$ correspond to the edge $\{n, m\} \in E$
A	: the set of arcs in G_A
$(n, m) \in A$	: an arc from node n to m
$T = (W, F)$	: a tree
W	: the set of nodes in T
F	: the set of edges in T
$c(n, m)$	: a positive cost associated with the edge $\{n, m\} \in E$ (in digraphs: $c(, n, m) == c(m, n)$
$D(v)$	: a positive integer associated with the node $v \in V$; upper bound for the node degree in any spanning structure
M	: the highest degree bound in V
$V_m \subseteq V$	: the sub-set of nodes with degree bound m
$d_G(v)$	: the degree of the node v in G
v^i	: the i-th occurrence of $v \in G$ in a hierarchy (*i.e.* in the tree T)
$\Theta(v)$	: the set of nodes in W corresponding to the node v in G
$W_m \subseteq W$	: the sub-set of nodes with degree bound m in the tree
$\Gamma^-(v)$	: the set of predecessor nodes at the node v in G_A
$\Gamma^+(v)$	: the set of successor nodes at the node v in G_A

As usual, $|X|$ is the number of elements in X. Trivially: $|V| = \sum_{m=1}^{M} |V_m|$ and $|W| = \sum_{m=1}^{M} |W_m|$ (generally they are different).

3 Degree Constrained Spanning Problems

It is important to highlight how the constraints on the degree of the nodes can be interpreted. Usually they are considered as budgets. A node can participate to the span until the exhaustion of its budget as it is presented in the following sub-section.

3.1 Minimum Spanning Trees under Degree Budget Constraints

The degree-constrained spanning problem was formulated for the first time in [5] and the mnimum spannng tree under degree constraints is one of the NP-hard problems listed in [6]. The nodes in an undirected graph $G = (V, E)$ are assigned positive integer values corresponding to limits representing the degree budget of the node. The degree bounds represents the maximum number of neighbors connected to the node in any spanning structure. They can be homogeneous or not in the node set. Often a cost or length function is also given. The problem of Degree (budget) Constrained Minimum Spanning Tree (DCMST) is to find the

minimum cost tree spanning the node set and satisfying the budget constraints. If it exists, the minimum cost solution is a spanning tree, since cycles are useless.

When the degree bound B is homogeneous, the maximum degree of any node in the spanning tree is at most B. In the case of $B = 2$, the solution (if it exists) is the minimum Hamiltonian path. In [16], the authors present the problem as a bi-criteria optimization and negative results are presented on the approximation of the optimum. Heuristic solutions were proposed in several works (cf. [3] [16] [7] [8] [15] [17] [18]).

The solution is different when the degree constraints correspond to momentary capacity limitations of the nodes. In this case the limited capacity is available for each visit of the node. So the (optimal) solutions can return several times in the nodes and profit from their available capacities.

3.2 Degree Capacity Constrained Spanning Problems

If the constraints are due to limited instantaneous capacities of the nodes, the minimum cost spanning structure is always a hierarchy [11]. Fig. 2 illustrates the interest of the hierarchies in the case of capacity constrained spanning. We are looking for the minimum cost connected spanning of this graph such that the degree of the nodes for each visit is limited to three. Trivially, there is no spanning tree but there is a spanning hierarchy satisfying the constraints. It uses the node b twice, and each occurrence of this node respects the degree constraint as it is shown in Fig. 2/b).

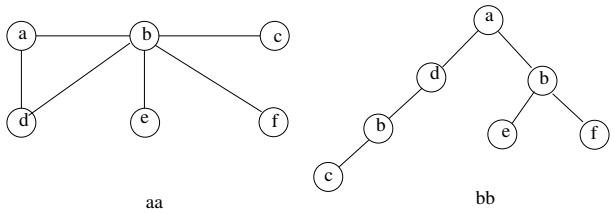

Fig. 2: A minimum cost spanning hierarchy

In [13] it has been demonstrated that the cost optimal solution with uniform capacity bounds corresponds to a spanning hierarchy. The problem is NP-hard but the solution always exists and an ILP based formulation has been proposed in [10]. Here, we examine the optimal solution with non uniform degree bounds.

Definition 4 (Node Capacity Constrained Minimum Spanning Problem) *Let $G = (V, E)$ be an arbitrary connected graph with positive integer degree bounds $D(v)$ representing the instantaneous maximum capacity for each node $v \in V$. Eventually positive costs $c(e)$ can be associated with edges $e \in E$. The problem is in finding the minimum cost hierarchy spanning the node set V s.t. the degree constraints are respected for each visit of the nodes.*

In the case of non uniform bounds, even the spanning hierarchy does not always exist. The conditions of existence for the solution have been analyzed in [12]. The following theorem formulates necessary and sufficient conditions for their existence in loop-free graphs.

Theorem 1. *A hierarchy spanning the whole node set of a connected graph and respecting heterogeneous degree constraints can be found, iff*
a) there is no separator in V_1 *(separator nodes having a degree bound* 1*) and*
b) $|V_1| \leq 2$ *(there are at most two nodes with degree bound* 1*) or*
c) $|V_1| > 2$ *and these nodes are neighbor nodes of a node* $w \in V_m$ *s.t.* $m \geq |V_1|$ *respect the degree constraints or*
d) $|V \setminus V_1| \geq 2$ *(there are at least two nodes with degree bound greater than* 1*) and* $|V \setminus V_2 \setminus V_1| \geq 1$ *(there is at least one node with degree bound greater than* 2*).*

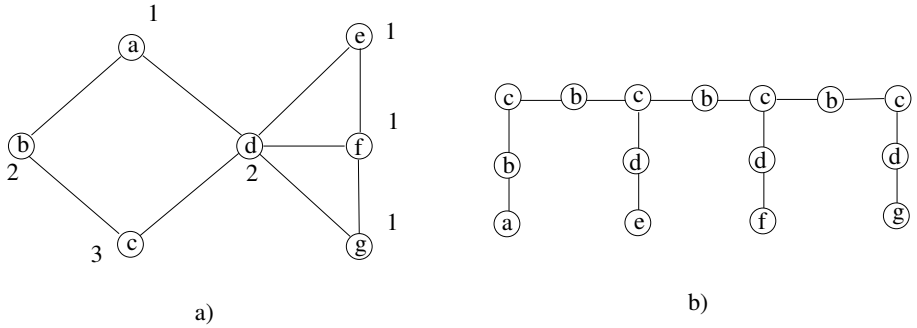

Fig. 3: Illustration of Theorem 1

Fig. 3 illustrates a spanning hierarchy corresponding to the condition *d)* of Theorem 1. The occurrences of the node c (having a degree bound 3) are the central nodes of spiders which are connected by paths through the node b. Trivially it is not the "best" spanning hierarchy. In some cases, one can easily find hierarchies containing less edges. Notice that loops on nodes with degree bound greater than 2 make possible the construction of the spanning hierarchies.

To find the cost minimum is an NP-hard problem.

Theorem 2. *The minimum cost spanning hierarchy problem respecting non-uniform degree constraints is NP-hard.*

Proof. Let $G' = (V', E')$ be the graph obtained from G by connecting $D(v) - 2$ new leaves to each vertex $v \in V$ s.t. $D(v) > 2$ as it is illustrated in Figure 4. Let L be the newly added set of leaves. Following the construction: $V_1' = V_1 \bigcup L$ and $V_i' = V_i, \forall i > 1$.

Let us suppose that the solution exists in G. Consequently, it exists also in G' Adding the new leaves does not create separators in V_1'. If $—V_i| = 0, \forall i > 2$, G admits a solution iff $|V_1| \leq 2$. Trivially, no new leaves are added and $|V_1| = |V_1'|$ The degree constrained minimum spanning hierarchy of G' contains all edges leaving to the new leaves. If $|V_1| > 2$, since G admits a solution, there exists at most one node in $V_i, i > 2$ and a second node with degree bound greater than 1, and the solution exists also in G'.

The degree constraints in the attachment vertices in G' are respected, iff G is covered by a Hamiltonian walk, in which the degree of nodes is at most 2. The computation of this latter is NP-hard [14]. ■

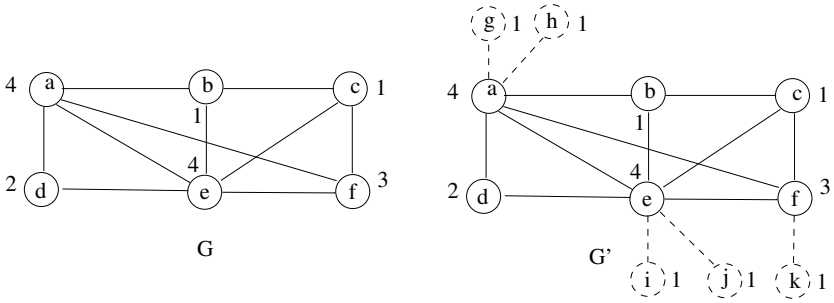

Fig. 4: The initial graph G and the completed graph G'

Several algorithms can be used to compute the optimal solution. Before the presentation of an ILP based procedure for finding it, we propose some useful properties of the spanning hierarchies.

4 Some Properties

Properties 4.1. - 4.3 in [10] have been formulated for the problem under uniform degree bounds and can be generalized as follows. The first property is true in all trees containing at least one edge.

Property 1. Let T be an arbitrary tree with at least two nodes. Let us suppose that the degrees of nodes are in $[1..M]$. The number of leaves in the tree is:

$$|W_1| = \sum_{m=2}^{M} |W_m| \cdot (m-2) + 2$$

Proof. The proof is trivial by induction. In the tree with only one edge, there are two leaves and the property is true. Let us suppose that the property is true

for k nodes. By adding a new node with degree m', new leaves are added to the tree (the simplest new sub-graph is a star). When the star is added at a leaf, $m'-1$ new leaves are created and an old leaf becomes an intermediate node. If the star is added between two existing ones, the number of new leaves is $m'-2$. ■

In the following we suppose that hierarchies spanning G and respecting the degree constraints exist (*i.e.* the conditions in Theorem 1 are met).

Property 2. In an optimal hierarchy let $\Theta(v) = \{v^i, i = 1..|\Theta(v)|\}$ be the set of nodes in T corresponding to $v \in V_m$ and let $d_T(v^i)$ be the degree of the node occurrence v^i in T. For the number of neighbors of v (all occurrences taken together) in the hierarchy the following inequality is held:

$$\sum_{i=1}^{|\Theta(v)|} d_T(v^i) > m \cdot (|\Theta(v)| - 1) + 1$$

Proof. *cf.* the proof of Properties 4.2. in [10]. ■

Trivially, to always respect the degree constraints the inequality can be completed:

$$\sum_{i=1}^{|\Theta(v)|} d_T(v^i) \leq m \cdot |\Theta(v)|$$

Property 3. Let $\mathcal{H}$ be the set of optimal hierarchies covering G. There exists an optimal hierarchy $H^* = (T^*, h^*, G) \in \mathcal{H}$ s.t. $\forall v \in V_m, m > 2$, there are at least $|\Theta(v)| - 1$ occurrences in T^* with degree m and related to v.

Proof. *cf.* the proof of Properties 4.3. in [10]. ■

With other words, there exists an optimal hierarchy in which all occurrences except eventually one related to a node $v \in V_m$ have the maximum capacity degree m of v.

Property 4. The maximum number of sub-hierarchies delimited by k occurrences of a node $v \in V_m, m > 1$ is equal to $n_v(k) = m + (k-1) \cdot (m-1)$. Consequently, the number of occurrences $|\Theta(v)|$ related to v to connect n_v sub-hierarchies delimited by these occurrences in a hierarchy is at least $\frac{n_v - m}{m-1} + 1$.

Proof. By induction, one can calculate the maximum number of sub-hierarchies delimited by occurrences of $v \in V_m$. Let $n_v(k)$ be the maximum number for k occurrences. One occurrence of v connects at most m sub-hierarchies that do not contain v.

$$n_v(1) = m$$

Let us suppose that a hierarchy contains $n_v(k-1)$ sub-hierarchies delimited by $k-1$ occurrences of v. By adding a new occurrence of v with the maximum

number of sub-hierarchies to the hierarchy, the number of sub-hierarchies is incremented by $m-1$. The following recurrence shows the incremental:

$$n_v(k) = n_v(k-1) + m - 1$$

which explicitly gives:

$$n_v(k) = m + (k-1) \cdot (m-1)$$

Inversely, if n_v is the number of sub-hierarchies delimted by the occurrences of v, at least $k = |\Theta(v)| \geq \frac{n_v - m}{m-1} + 1$ occurrences of v are needed to connect them. ■

Figure 5 illustrate the property. The sub-hierarchies delimited by node b are illustrated in Figure 5/b) in a degree constrained hierarchy spanning the graph shown with the degree bounds in Figure 5/a).

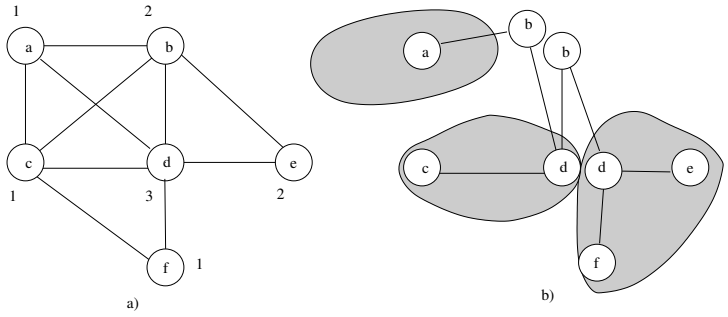

Fig. 5: A degree constrained spanning hierarchy and its sub-hierarchies delimited by node b

Property 5. Let $v \in V_m, m > 1$ be a node in G. If an occurrence v^1 related node in T^* is leaf in the optimal hierarchy, it is the only occurrence related to v.

Proof. Let us suppose that a second occurrence v^2 exists in the hierarchy (*i.e.* in the tree T^*). In this case, v^1 and its adjacent edge can be deleted and the coverage of the node is held. H^* can not be optimal. ■

Since a node $v \in V_1$ should be leaf in any DCMSH, and there is only one (leaf) occurrence corresponding to v in the DCMSH.

The following lemma resumes the upper bounds on the number of occurrences of nodes in DCMSHs.

Lemma 1. *In an MCDSH, the number $|\Theta(v)|$ of occurrences of a node v is (or bounded by):*

a) $|\Theta(v)| = 1$, *if* $v \in V_1$,

b) $|\Theta(v)| = |V| - 1$, *if* $v \in V_2 \wedge V \setminus V_1 \setminus V_2 = \emptyset$,

c) $|\Theta(v)| \leq B(v) = \lceil \frac{|V|-4}{m-2} \rceil$, *if* $v \in V_2 \wedge V \setminus V_1 \setminus V_2 \neq \emptyset$,

d) $|\Theta(w)| \leq B(w) = \frac{|V|-4}{m-2} + 1$, *if* $w \in V_m \wedge V \setminus V_1 \setminus V_2 \neq \emptyset$.

Proof. a) Trivial. A leaf can not be present twice in an optimal hierarchy (*cf.* Property 5).

b) In this case, there is no node with degree bound greater than two: $V \setminus V_1 \setminus V_2 = \emptyset$. At most two nodes in V_1 may exist in G and the optimal spanning hierarchy is a walk that can cross a node $v \in V_2$ several times. The maximum number of crosses is equal to $|V| - 1$, supposing that after visiting another node in V_2, the walk returns to v.

c) and d) Here $V \setminus V_1 \setminus V_2 \neq \emptyset$. Let us suppose that only one node $w \in V_m, m > 2$ exists and $V_2 = \emptyset$. Following Theorem 1), if $|V_1| \leq D(w)$, a single occurrence of w is sufficient in a DCMSH (which is a star). This case is very limited. Let us suppose that the number of nodes is large enough and several occurrences of w are needed. To cover the node set and to use multiple occurrences of w, at least a node $v \in V_2$ is needed. We propose the construction of an extreme topology that implicates the maximum participation of w and v (each of the nodes in V_1 has only one leaf occurrence). The maximum number of occurrences of w is obtained, when only occurrences of w are branching nodes in the hierarchy. (Let us suppose that there is another branching node w'. In this case, some nodes are connected to the occurrences of w' without passing by w and the number of occurrences of w in the hierarchy is not maximal. In summary, w is the unique node with degree bound greater than 2.) In the extreme topology, there is only one node $v \in V_2$. If each path delimited by occurrences of w and leaves contains also v, the number of occurrences of v is maximum. The number of leaves in the extreme solution is equal to $L = |V| - 2$. Since w is the only branching node, following Property 1, the maximal number of leaves is $|\Theta(w)| \cdot (m-2) + 2$. The smallest integer value satisfying this condition is: $|\Theta(w)| = \lceil \frac{|V|-4}{m-2} \rceil$.
Let us consider the paths between occurrences of w and leaves in the DCMSH. The maximum number of occurrences of v is given when v is part of all paths. The set of paths are the sub-hierarchies delimited by the occurrences of w. In the extreme case, there are as many occurrences of v as sub-hierarchies. Following Property 4, a limit can be formulated: $|\Theta(v)| \leq m + (|\Theta(w)| - 1) \cdot (m-1)$.
Finally: $|\Theta(v)| \leq \frac{|V|-4}{m-2} \cdot (m-1) + 1$.

The expression in Lemma 1 shows that the maximum possible numbers of node occurrences depend also on the degree bound of the branching nodes. The extreme bounds related to the size $|V|$ of the graph can be obtained when $m = 3$. In this case: $B(v) = 2 \cdot |V| - 7, v \in V_2$ and $B(w) = |V| - 4, v \in V_3$.

Figure 6 shows such a graph with $a \in V_3$ and $b \in V_2$. The corresponding minimum cost spanning hierarchy is plotted using a labeled tree. Each occurrence of a except of two covers $m - 2 = 1$ leaf passing by the node b.

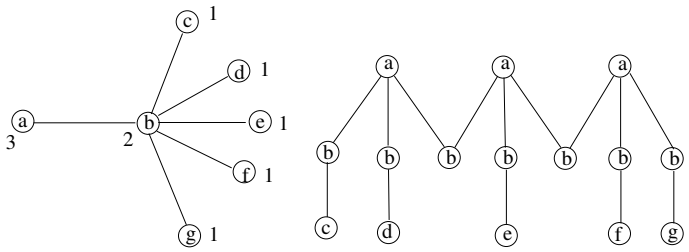

Fig. 6: A particular graph and its coverage by the DCMSH

The following property indicates an upper bound on the number of occurrences of an edges in optimal hierarchies.

Lemma 2. *The number of occurrences of an edge in an optimal hierarchy* $H = (T, h, G)$ *is limited by* $EB = \max_{i \in \{3 \ldots M\}} \max_{v \in V_i} [(i-1) \cdot |\Theta(v)|]$.

Proof. Let us suppose that each occurrence of a node $v \in V_m$ has exactly m adjacent edges in T. Moreover, $m-1$ adjacent edges of each occurrence of v correspond to the same edge $e \in E$. This edge e is associated to $(m-1) \cdot |\Theta(v)|$ edges in T. An upper bound for the edge duplications in a DCMSH of G is the maximal value of the possible duplications:
$EB = \max_{i \in \{3 \ldots M\}} \max_{v \in V_i} (i-1) \cdot |\Theta(v)|$.

The following trivial propoerties allow the computation of the solution as presented here after.

Let $G = (V, E)$ be an arbitrary graph with capacity like degree constraints and let $G_A = (V, A)$ be the equivalent digraph (defined on the same node set, and replacing each edge of G by a couple of arcs, one of both direction). Let the degree constraints be the same in the two graphs and let the length of arcs be the same as the one of the corresponding edge. Let s be an arbitrary (root) node in G_A. The directed degree constrained minimum hierarchy spanning G_A exist (without loss of generality, one can suppose that there is only one optimum). The length of this directed optimal hierarchy is the same as the length of the DCMSH in G.

Let I_A be the set of arcs in the image of a hierarchy in a digraph. A positive integer value is associated with each arc in I_A, indicating the number of occurrences of the arc in the hierarchy. This valuated image permits the definition / reconstruction of the directed hierarchy.

5 Computation of the Exact Solution

Similarly to the computation in [10], we propose an ILP-based computation of the solution. Remember that the optimal spanning hierarchy, if it exists, can return several times to some nodes and can pass several times by some

edges. Since the solution is not a sub-graph, the proposed ILP computes the image of the solution and in an equivalent digraph. The computation associates with each arc of the image an integer value indicating how many times the arc is used in the solution. Taking these values into account, the cost of the image corresponds to the cost of the spanning hierarchy. The computed image is with minimal cost and permits the reconstruction of the optimal hierarchy. This reconstruction separates the eventual different passes in the nodes such that each pass corresponds to a separated node occurrence in the solution and the degree constraints are always respected.

The computation has four steps and illustrated by Fig. 7.

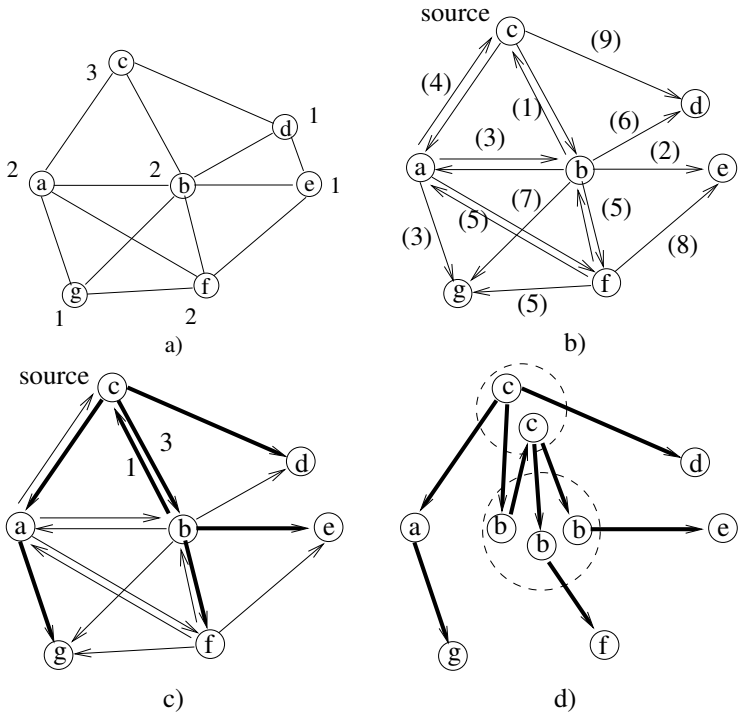

Fig. 7: The steps of the construction of a DCMSH

the details of the steps follow.

5.1 Verification for the Existence

Theorem 1 gives the necessary and sufficient conditions for the existence of spanning hierarchies under non uniform degree constraints. Before the computation of an eventual solution, these conditions must be verified. The verification is based on three steps.

– At first, it must be controlled that there is no separator in V_1. If there is at least one node in V_1 all the neighbors of which are in V_1, this node remains isolated, the computation exits without solution.
– Even if there is no separator inside V_1, subsets of V_1 can be separators for the graph. If there is a separator subset in V_1, trivially V_1 is also a separator. This latter can be verified using a DFS algorithm. After the deletion of V_1 from G, the number of connected components can be calculated. If it is greater than 1, there is a separator and the solution does not exists.
– If there is a single node $v \in V_m$ s.t.$m < |V_1|$, there is no solution.
– If $|V_1| > 2$ and the other nodes are in V_2, there is no solution.

After the verification, if the solution exists, the next phase prepares a directed graph for the computation of the optimal image of the spanning hierarchy.

5.2 Creation of an Equivalent Directed Graph

A hierarchy can passe by edges and nodes several times. To facilitate the control of the constraints, we propose a directed graph in which each edge of G is duplicated and replaced by two arcs, one in each direction and having the same cost as the original edge. In the proposed directed model incoming and outgoing flows can be distinguished in the nodes. and the number of the flows transiting an arc can be represented by an integer value associated with (*cf.* Subsection 5.3). Using these valeus, a simple verification to respect the degree constraints is possible. For an incoming flow crossing a node $v \in V_m$, at most $m - 1$ outgoing flows can correspond.

The computation of the image uses flows, one flow to each destination from an arbitrary chosen root s. The selection of s influences the simplifications of the digraph. The following simplifications are made. If $s \in V_1$, it can have only one successor and consequently only outgoing arcs. The other nodes in V_1 can have only incoming arcs (there is no departure from a leaf). Useless incoming and outgoing arcs for nodes in V_1 can be deleted before the computation of the solution.

The initial values of the variables can influence the rapidity of the computation. For this reason, we propose the initialization of the flows corresponding to a spanning tree which can be computed in polynomial time. [1]

The simplified digraph of the example is illustrated in Fig. 7/b).

5.3 Computation of the Image of the Minimum Flow

In this section, the ILP formulation of the valued image of the solution in the directed auxiliary graph is proposed. Remember, the values associated with the

[1] It is the Minimum Spanning Tree with Spacified (desired) Leaves. In this tree the degree constrains are respected only in the nodes of V_1. In the rest of the graph, a simple MST is computed and the leaves are connected with their best cost adjacent edges.

arcs of the result mean the number of occurrences of the arcs in the directed hierarchy.

ILP variables:

$x(m,n)$: Integer variable. Equal to the number of the occurrences of the arc (m,n) in the resulted hierarchy

$F(m,n)$: Commodity flow variable. Denotes the quantity of flow transiting on the arc (m,n).

Objective:

We minimize the cost of the spanning structure respecting the constraints.

$$Minimize \sum_{m \in V} \sum_{n \in \Gamma_m^+} x(m,n) \cdot C(m,n) \tag{1}$$

The values associated with the arcs make it possible to follow the number of incoming and outgoing arcs in all the nodes. The degree constraints can be formulated as follows.

At the arbitrary selected source s, there are returns iff the degree $D(s)$ is not sufficient to be the start point of the outgoing messages:

$$\sum_{n \in \Gamma_s^+} x(s,n) \leq D(s) + (D(s)-1) \cdot \sum_{n \in \Gamma_s^-} x(n,s) \tag{2}$$

If the degree bound of the source is equal to one, there is only one outgoing arcs.

At the other nodes, since each occurrence of the node m can be the start point of $D(m)-1$ outgoing messages:

$$\sum_{n \in \Gamma_m^+} x(m,n) \leq (D(m)-1) \cdot \sum_{n \in \Gamma_m^-} x(n,m) \qquad \forall m \in V \setminus \{s\} \tag{3}$$

Notice that for nodes with degree bound one, there is no outgoing arc used.

In an optimal hierarchy there is a strictly minimal number of occurrences of any node. In our directed model, the number of occurrences of a non source node is equal to the number of its predecessors. For the source node this value is incremented by one (if there are several occurrences of the source, the first one has not predecessor). The number of occurrences should be sufficient for the outgoing messages. We can establish the following (not necessary but practical) inequalities. At the source:

$$\sum_{n \in \Gamma_s^+} x(s,n) > D(s) + (D(s)-1) \cdot \sum_{n \in \Gamma_s^-} x(n,s) \tag{4}$$

In an other nodes m with degree bound ($D(m) > 1$ the following inequality is true):

$$\sum_{n \in \Gamma_m^+} x(m,n) > (D(m)-1) \cdot (\sum_{n \in \Gamma_m^-} x(n,m) - 1) \qquad \forall m \in V \setminus \{s\}, D(m) \neq 1 \tag{5}$$

Every node except the source has a predecessor in the hierarchy. This condition guarantees the full coverage.

$$\sum_{n \in \Gamma_m^-} x(n,m) \geq 1 \qquad \forall m \in V \setminus \{s\} \tag{6}$$

Full coverage does not mean full connectivity. Flows can be used to express the connectivity. The arbitrary source is the start point of one flow per destination. T he destinations are the other $|V| - 1$ nodes.

$$\sum_{n \in \Gamma_s^+} F(s,n) = |V| - 1 \tag{7}$$

Each non source node is the destination of one and only one flow.

$$\sum_{n \in \Gamma_m^+} F(m,n) = \sum_{n \in \Gamma_m^-} F(n,m) - 1 \qquad \forall m \in V \setminus \{s\} \tag{8}$$

Flows transit by the occurrences of arcs.

$$F(m,n) = 0 \qquad \forall m,n \in V \ if \ x(m,n) = 0 \tag{9}$$

$$F(m,n) > 0 \qquad \forall m,n \in V \ if \ x(m,n) > 0 \tag{10}$$

If an arc does not belong to the solution, it does not carry any flow. Moreover each arc should carry non-zero flow if it is used in the output graph, and the value of this flow should not be beyond the total flows transmitted by m.

The number of occurrences of edges (and consequently of arcs) is limited by the value EB presented by Lemma 2. Notice that EB is constant and can be calculated beforehand.

$$x(n,m) \leq EB \qquad \forall m,n \in V \tag{11}$$

Obviously, solving our linear program does not give a DCMSH. A last step is needed to construct the optimal "non-elementary tree".

Fig. 7/c) illustrates the valued image of the solution in our example.

5.4 Reconstruction of the Optimal Hierarchy

The ILP returns the image of the solution in the auxiliary directed graph. The arcs in the image have values corresponding to the number of their occurrences in the solution: the variables $x(n,m)$ indicate, how many times the arc (n,m) is traversed from n to m. From this image a DCMSH can be built. First, a directed, labeled tree can be reconstructed. Several directed trees can correspond to the image, even if the number of occurrences / traverses of arcs are known. The properties discussed previously (*cf.* Section 4) permit the construction. Here, let us resume their application.

Each node occurrence has one and only one predecessor (except the source). If a node n has $i = \sum_{m \in \Gamma^-(n)} x(m,n)$ predecessors, then n must be duplicated i times in the tree (*i.e.* in the optimal hierarchy) To each occurrence of n there is one and only one incoming arc. For the arbitrary chosen source, one occurrence has no predecessor and the number of occurrences is $i+1$.

$o = \sum_{m \in \Gamma^+(n)} x(n,m)$ indicates the number of outgoing arcs in the tree for all occurrences of n taken together. To respect the degree constraints, each occurrence of n can have at most $D(n)$ outgoing arcs. Property 3 indicates that for a node n, there exists an optimal hierarchy in which all occurrences of n except one have the maximal degree $D(n)$. Following this property, the outgoing arc occurrences can be distributed between the occurrences of n s.t. $D(n)-1$ outgoing arc occurrences are connected to the first $o-1$ occurrences of n.

1. At first, the successors of the nodes is computed. Let $suc[]$ be the vector of successors s.t. $suc[n]$ corresponds to the set of successors of the node n. For each node n this set $suc([]$ is computed based on $\Gamma^+(n)$ and on the values $x(n,m)$. For each successor node $m \in \Gamma^+(n)$ $x(n,m)$ occurrences of m should be added to $suc[n]$.

2. The construction of the tree starts at the source node s and follows a Depth First Search (DFS) like construction.
   (a) For the source node s, the first occurrence has no predecessor. If the number of successors is greater than $D(s)$, the first occurrence of the source have $D(s)$ successors and outgoing arcs which are added to this first occurrence. The remaining arcs are associated with the following occurrences in the same way as for the other nodes.) The computation continues with the first successor of the source (current node).
   (b) If the current node n is a leaf without successors, the algorithm returns to the predecessor. Otherwise, the set $Suc(n)$ of successors is processed as follows:

       The outgoing arc occurrences can be distributed between the occurrences of n. To respect the degree constraints, each occurrence of n can have at most $D(n)$ neighbor arcs. Property 3 indicates that for a node n, there exists an optimal hierarchy in which all occurrences of n except one have the maximal degree $D(n)$. Following this property, the outgoing arc occurrences can be distributed between the occurrences of n s.t. $D(n)-1$ outgoing arc occurrences are connected to the first $|\Gamma^+(n)|-1$ occurrences of n. The remaining arcs are associated with the last occurrence of n.

The procedures are presented by Algorithm 1 and 2.

Omitting the direction of arcs, a tree T_u is obtained. Since the tree T_d returned by Algorithm 1 (and consequently T_u) refers to the nodes in G, $H = (T_u, h, G)$ is a hierarchy. The mapping h associates a node of G to each node of T_u corresponding to the labels in T_u.

Algorithm 1 Computation of a directed tree labeled by the node occurrences of G_A and following the optimal flows

Require: The graph $G_A = (V, A, D)$, with degree $D(n)$ associated to $n \in V$, the source s and the matrix X with values $x(n, m)$ computed by the ILP
Ensure: A labeled directed tree T_d
global variables
T_d **tree**
$suc[|V|]$ **vector of sets of node occurrences**
end of global variables
begin
$T_d \leftarrow \emptyset$ {initialize an empty tree}
for all $n \in V$ **do**
$suc(n) \leftarrow \emptyset$ {set of successor node occurrences}
for all $m \in \Gamma^+(n)$ **do**
for $i = 1, \ldots, x(n, m)$ **do**
$suc[n] \leftarrow suc[n] \cup \{m^i\}$
end for
end for
end for
Construct $(s, D(s))$ {start from the first occurrence of the source}
return T_d
end

Algorithm 2 Performs a DFS construction of a sub-tree rooted at n

Require: A node n and the number DS of its successors
Ensure: A labeled directed sub-tree in T_d
procedure **Construct**(n, DS)
local variables
$curr$ **node**
DS **integer**
end of local variables
begin
for $i = 1, \ldots, DS$ **do**
$curr \leftarrow suc[n].first()$
ADD $suc[n].first()$ to n in T_d {Add the first successor of n to the tree}
$suc[n] \leftarrow suc[n] \setminus suc[n].first()$
$DS \leftarrow \min(|suc[curr]|, D(curr))$
Construct $(curr, DS)$
end for
end

Theorem 3. *The result H is a DCMSH.*

Proof. Each node in V is associated with at least one node of T_u, the hierarchy H spans the node set V. By construction, the nodes in the tree T_u respect the degree constraints (a node occurrence of the node $n \in V$ has at most $D(n)$ neighbors). So, H is a degree constrained hierarchy spanning V. It is with minimum cost. Its cost is the sum of the cost of edge occurrences. An edge $\{m, n\}$ is present $x(m, n) + x(n, m)$ times in H (in the tree T_u).

$$c(H) = \sum_{\{m,n\} \in E} c(m, n) \cdot (x(m, n) + x(n, m))$$

This sum is minimized by the ILP. ■

6 Evaluation of the Solutions

In this section we compare the different values (mainly the costs) of four solutions (if they exist): the MSTSL, the DCMSH, the DCMST and the result of the heuristic [12].

For this comparison and analysis, random graphs based on the model of Albert-Barabási [1] were generated following the usual two steps:

- Creation of a connected small graph of m_0 nodes (a tree for example)
- Randomly add m_f nodes s.t. each node is connected to at most m old: the probability of an edge from the newly added node to an old one is proportional with the degree of this latter

$$P_i = \frac{k_i}{\sum_j k_j}$$

The implementation used the Leda library in C++ [9].

The computation of the MSTSLs and the heuristic were implemented in C++ in the generated graphs and the exact solutions DCMSHs and DCMSTs were computed using the CPLEX Ilog server [4] to implement the corresponding ILPs. The heuristic and the computation of the MSTSL are in polynomial time, but the exact solutions need expensive computations. Fortunately, in the tested random graphs the computation of the exact solutions asked from some seconds to some hours and were realizable.

The simulation settings and results are presented in the following.

6.1 Simulation Settings

In the experimentation, some parameters were variable: the number of nodes in the random graphs, the distribution of the degree bounds on the nodes and the costs of the edges.

We used medium size topology: started from a small tree of 5 nodes and using the BA algorithm, the final random graphs contained 80 - 170 nodes. Notice that the number of edges in the graphs were not deterministic, since the edges were generated randomly. In each experimentation presented here after, we indicate the average number of edges in the different graphs.

The degree bounds were set randomly. In each case, an interval $[D_{min}, D_{max}]$ were given, and the integer values in this interval were associated with the nodes using an equi-probable (uniform) distribution. Often, the interval corresponded to $[1, D_{max}]$. Consequently the bounds $1, 2, ..., D_max$ were randomly but uniformly associated with the nodes.

The cost of edges were also randomly and uniformly associated with edges from the interval $[1, Cmax]$.

The detailed parameter settings of the experiments can be found in the corresponding sub-sections.

6.2 Runs and Results

Effect of the maximal degree bound In a first experimentation, we propose the analysis of the maximal degree bound and consequently the distribution of the applied bounds. In each run 100 random graphs with 100 nodes were created. The edge costs were randomly generated with $C_{max} = 5$. The following table resume the average properties of the graphs and also the number of cases in which trees and hierarchies corresponding to the constraints exist.

D_{max}	$\|V\|$	$\|V_1\|$	conditions	$\|VSG\|$	hierarchies	trees
3	463.36	34.06	86	613.06	86	50
4	463.75	25.14	96	689.67	96	96
5	464.29	19.29	100	751.69	100	100
6	462.97	17.1	100	768.76	100	100
7	463.4	14.01	99	797.2	99	99
8	463.07	12.9	100	807.07	100	100
9	463.77	11.54	100	822.09	100	100
10	463.54	9.92	99	835.36	99	99
11	463.48	9.12	100	841.31	100	100
12	463.6	8.32	100	849.74	100	100

Figure 8 shows the evaluation of the costs of the four examined solutions. The cost of the optimal hierarchies is always better or equal to the cost of the optimal tree (if these solutions exist). Since the degree bounds are equi-probably associated with nodes, by increasing the value of D_{max}, the probability of degree bounds one (and also of the other low degree bounds imposing real constraints on the degrees) decreases. Consequently, the difference between trees and hierarchies decreases and they costs tend to the cost of the MSTSL. Notice also that the heuristic offers also good costs.

The solutions do not exist in an important percent of the cases when D_{max} is low. For instance, with $D_{max} = 3$, only 50 / 100 trees exist.

To examine the effect of the cardinality of V_1 on the number of failed solution, the following scenario was realized. The probability of the nodes in V_1 varied from

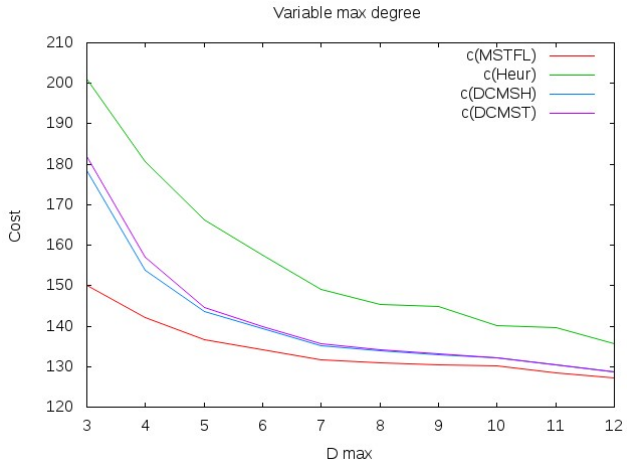

Fig. 8: The cost according to the degree bounds

5% to 50%. The following table resumes the results. With low number of leaves in V_1, trees and hierarchies exist. When there are at most 20% of nodes with degree bound one, the conditions for the existence of the solution are hardly fulfilled. There is no significant difference for trees and hierarchies (probably in the examined special random graphs).

$\|V_1\|/\|V\|$ %	$\|E\|$	$\|V_1\|$	conditions	$\|E_{SG}\|$	$nb(h)$	$nb(t)$
5	462.92	4.89	100	881.39	100	100
10	462.81	10.02	100	833.73	100	100
15	464.02	14.19	100	796.76	100	100
20	464.13	19.45	99	748.05	99	99
25	463.65	24.6	98	698.36	98	98
30	463.46	30.08	93	648.23	93	93
35	463.1	34.62	84	604.95	84	84
40	463.43	40.19	71	552.8	71	71
45	463.87	43.86	63	518.62	63	62
50	463.15	49.77	37	466.09	37	32

Effect of the edge costs In this case, the number of nodes was fixed to 100 and D_{max} was equal to 3. The edge costs were randomly generated from the interval $[1, C_{max}]$ and the value of C_{max} varied as it is indicated in the following table.

C_{max}	$\lvert E_{G'_A}\rvert$	dupl	$nb(h)$	$nb(t)$	c(lb)	c(Heur)	c(h)	c(t)
1	100.725	1.24176	91	57	99	144.176	101.143	99
2	102.349	1.54217	83	57	104.675	141.265	111.458	109.088
3	105.556	2.04444	90	63	116.5	154.589	132.644	132.746
4	107.536	2.36905	84	53	132.417	174.774	155.643	158.585
5	108.837	2.5	92	55	147.804	194.859	176.652	181.709
6	109.209	2.81319	91	61	165.934	220.824	197.593	206.541
7	108.965	3.01176	85	63	180.365	240.624	215.153	228.381
8	110.022	3.1978	91	61	199.56	264.835	240.495	259.279
9	109.683	3.59756	82	54	219.646	290.707	263.268	283.13

Independently from the edge costs, the obtained typologies are relatively stable; $|V_1|$ and $|E_{SG}|$ show a small variation. DCMST does not exist in $50-60\%$ of cases and DCMSH does not exist in 10% of cases. The first lines of the results need explanation. Let us analyze the first line. If the DCMST exist (in 57 cases of 100 runs) the cost is equal to 99, which is the cost of the 99 edges needed to cover the 100 nodes. The average cost of DCMSHs is greater, but it is the average of 91 cases which contain hierarchies returning several time to some nodes (cases not covered by trees). The average of the maximal number of usage of an edge (the number of duplications) is indicated in the column *dupl*.

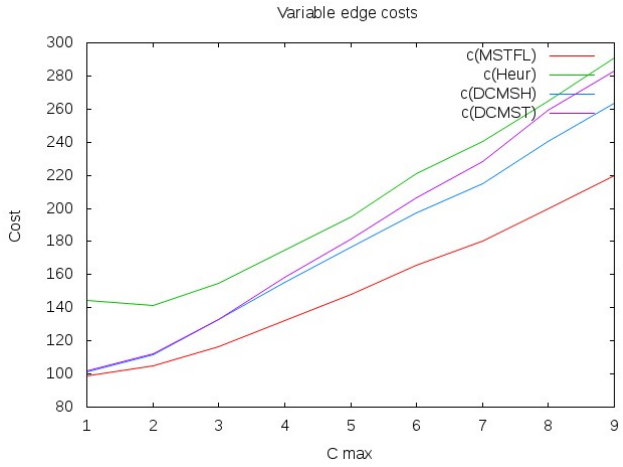

Fig. 9: The overall cost according to the distribution of edge costs

DCMST does not exist in $50-60\%$ of cases and DCMSH does not exist in 10% of cases. The first lines of the results need explanation. Let us analyze the first line. If the DCMST exist (in 57 cases of 100 runs) the cost is equal to 99, which is the cost of the 99 edges needed to cover the 100 nodes. The average cost of DCMSHs is greater, but it is the average of 91 cases which contain hierarchies returning several time to some nodes (cases not covered by trees).

Varying the size of the graphs

A third series permits to see the effect of the size of Albert-Barabási random graphs on the minimum spanning structures. The experimentation is summarized in the following table. The number of nodes $M = m_0 + m_f$ varied from 80 to 170 nodes. D_{max} was equal to 3 and C_{max} was 5.

M	$\|E\|$	$\|V_1\|$	$\|E_{G'_A}\|$	dupl	$nb(h)$	$nb(t)$	c(lb)	c(Heur)	c(h)	c(t)
80	364.5	20.14	543.69	1.7732	97	97	113.381	146.34	122.124	124.351
90	414.71	22.16	625.29	1.81633	98	98	125.673	159.673	135.408	137.969
100	463.67	24.98	694.83	1.86598	97	96	139.897	177.093	150.351	152.802
110	513.32	27.62	768.61	1.77551	98	98	153.735	195.276	165.847	168.888
120	562.41	29.21	849.71	1.82653	98	98	167.571	211.327	180.347	183.306
130	612.35	33.19	912.57	1.90722	97	97	182.34	231.216	196.505	200.412
140	661.74	35.55	990.49	2.03158	95	95	194.853	248.642	210.789	215.632
150	711.2	37.71	1063.44	1.97872	94	94	211.351	265.606	227.915	232.457
160	760.62	40.5	1139.45	1.89796	98	97	223.867	283.52	241.643	246.649
170	810.81	42.95	1212.8	2.08333	96	96	236.906	304.094	257.229	262.969

It can be seen that the observed values increase quasy linearly with the size of the graphs. For instance, let the lines with 80 and 160 nodes be compared. The average number of edges, the number of desired leaves in V_1, the cost are almost double. The number of DCMSHs and DCMSTs are mainly the same. Note that trees are 5 - 10 % more expensive than hierarchies.

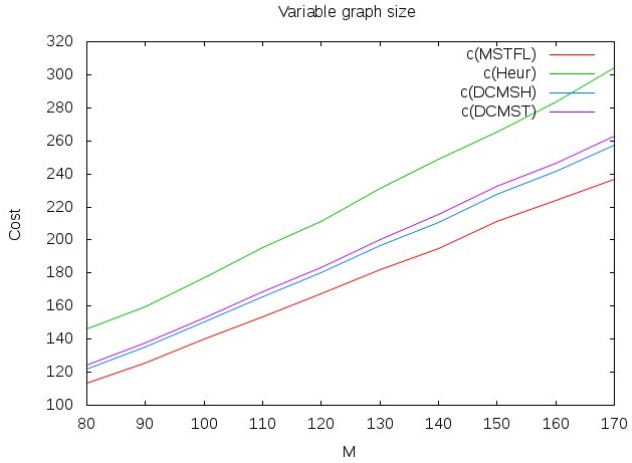

Fig. 10: The cost according to the graph size

7 Conclusios and Perspectives

Our analysis shows that

- DCMSHs exists even if DCMSTs do not exist,
- DCMSHs have less costs than DCMSTs,
- the heuristic performs very well,
- the execution time is raisonnable in Albert-Barabási random graphs.

The set of specified leaves with degree bound 1 is crucial for the properties of the solutions. If this set is large, the solution may not exist and the approximation within a constant is not trivial (we consider it as an open question). Notice that the heuristic provides results relatively close of the optimum.

In future works, the approximation of spanning hierarchy problems with different degree constraints in various graphs should be analyzed. We conjecture that in some cases, depending on positions of leaves in V_1, approximations can be found even if $|V_1| > 3$. Another important perspective is the analysis of partial spanning problems like degree constrained Steiner problems. We believe that a good part of the results presented in this paper can be applied in partial spanning problems. Applications of the degree constrained spanning hierarchies can be found, for instance, in optical broadcast / multicast routing. In these applications, additional constraints (*e.g.,* the uniqueness of the used wavelength in the fibers, limited length of paths, etc;) exist, and various constraints must be fully satisfied. Analyzing these issues promises further interesting challenges....

References

1. Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74:47–97, Jan 2002.
2. F. Bauer and A. Varma. Degree-constrained multicasting in point-to-point networks. In *INFOCOM '95: Proceedings of the Fourteenth Annual Joint Conference of the IEEE Computer and Communication Societies (Vol. 1)-Volume*, page 369, Washington, DC, USA, 1995. IEEE Computer Society.
3. Bruce Boldon, Narsingh Deo, and Nishit Kumar. Minimum-Weight Degree-Constrained Spanning Tree Problem: Heuristics and Implementation on an SIMD Parallel Machine. *Parallel Computing*, 22(3):369–382, 1996.
4. IBM ILOG Cplex. V12. 1: Users manual for cplex. *International Business Machines Corporation*, 46(53):157, 2009.
5. N. Deo and S.L. Hakimi. The shortest generalized Hamiltonian tree. In *Sixth Annual Allerton Conference*, pages 879–888, 1968.
6. Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.
7. J. Könemann and R. Ravi. A Matter of Degree: Improved Approximation Algorithms for Degree-Bounded Minimum Spanning Trees. *SIAM J. Comput.*, 31(6):1783–1793, 2002.
8. J. Könemann and R. Ravi. Primal-Dual Meets Local Search: Approximating MSTs With Nonuniform Degree Bounds. *SIAM J. Comput.*, 34(3):763–773, 2005.

9. Kurt Mehlhorn and Stefan Näher. *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, 1999.
10. Massinissa Merabet, Miklós Molnár, and Sylvain Durand. ILP formulation of the degree-constrained minimum spanning hierarchy problem. *Journal of Combinatorial Optimization*, 36(3):789–811, October 2018.
11. Miklós Molnár. Hierarchies to Solve Constrained Connected Spanning Problems. Technical Report 11029, LIRMM, July 2011.
12. Miklós Molnár. On the Approximation of Degree Constrained Spanning Problems in Graphs under Non Uniform Capacity Constraints. Research report, LIRMM (UM, CNRS), September 2021.
13. Miklós Molnár, Sylvain Durand, and Massinissa Merabet. Approximation of the Degree-Constrained Minimum Spanning Hierarchies. In *SIROCCO*, pages 96–107, 2014.
14. Takao Nishizeki, Takao Asano, and Takahiro Watanabe. An approximation algorithm for the Hamiltonian walk problem on maximal planar graphs. *Discrete Applied Mathematics*, 5(2):211 – 222, 1983.
15. R. Ravi. Matching Based Augmentations for Approximating Connectivity Problems. In *Proc. of the 7th Latin American Symposium on Theoretical Informatics (LATIN'06)*, pages 13–24, Valdivia, Chile, 2006.
16. R. Ravi, Madhav V. Marathe, S. S. Ravi, Daniel J. Rosenkrantz, and Harry B. Hunt Iii. Approximation algorithms for degree-constrained minimum-cost network-design problems. *Algorithmica*, 31(1):58–78, 2001.
17. R. Ravi and Mohit Singh. Delegate and Conquer: An LP-Based Approximation Algorithm for Minimum Degree MSTs. In *ICALP (1)*, pages 169–180, 2006.
18. Mohit Singh and Lap Chi Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 661–670, New York, NY, USA, 2007. ACM.