The maximum node clustering problem

G. Carello, Federico Della Croce, Andrea Grosso, M. Locatelli

The maximum node clustering problem

G. Carello[*], F. Della Croce[†], A. Grosso[‡], M. Locatelli[‡]

Résumé

Dans cet article, nous introduisons un problème de graphes, appelé Maximum Node Clustering (MNC). Nous prouvons que ce problème est fortement NP-complet et montrons qu'il peut être approché en temps polynomial avec un rapport arbitrairement proche de 2. Pour le cas particulier où le graphe est un arbre, nous prouvons que le problème est NP-complet au sens faible puisqu'il généralise le problème du sac-à-dos et qu'on peut le résoudre en temps pseudopolynomial par une approche de programmation dynamique. Nous présentons également un FPTAS pour le cas des arbres.

Mots-clefs : Maximum Node Clustering, Sac-à-dos, Complexité, Approximation

Abstract

In this paper we introduce a graph problem, called Maximum Node Clustering (MNC). We prove that the problem is strongly NP-complete and show that it can be approximated in polynomial time within a ratio arbitrarily close to 2. For the special case where the graph is a tree, we prove that the problem is weakly NP-complete as it generalizes the $0/1$ Knapsack problem and is solvable in pseudopolynomial time by a dynamic programming approach. For this latter case an FPTAS is also presented.

Key words : Maximum Node Clustering, Knapsack, Complexity, Approximation

[*]Dipartimento di Elettronica, Politecnico di Milano. `carello@elet.polimi.it`
[†]Dipartimento di Automatica ed Informatica, Politecnico di Torino. `federico.dellacroce@polito.it`
[‡]Dipartimento di Informatica, Università di Torino, `{grosso, locatell}@di.unito.it`

1 Introduction

We consider the following problem: an undirected graph $G(V, E)$ is given, with nonnegative node profits π_i, $i \in V$, edge weights w_e, $e = \{i, j\} \in E$, and a given capacity $B > 0$; determine a subset $S^* \subseteq V$ such that the total profit $\pi(S^*) = \sum_{i \in S^*} \pi_i$ is maximum and

$$w(S^*) = \sum_{e \in \delta(S^*)} w_e \leq B \tag{1}$$

where $\delta(S^*)$ is the set of all edges having at least one endpoint in S^*. In what follows it is also assumed that $w(\{i\}) \leq B, \ \forall\, i \in V$, otherwise node i could not be part of any feasible solution and could be discarded in advance. We call such problem Maximum Node Clustering (MNC). This problem generalizes the $0/1$ Knapsack problem (see the reduction of Figure 1 in Section 3) and is strictly related to the Dense k-Subgraph problem [7, 1].

The problem models also a class of knapsack-like problems with quadratic capacity constraint, arising in some telecommunication network design problems (see [2], [3], [9] and [13]). In telecommunication networks with hierarchical architecture, terminal nodes —*terminals* in what follows — representing origins and destinations of traffic demands, are connected to hub nodes in charge of aggregating small flows into larger ones and route them on intra-hub backbone links. Usually hubs have a limit, B, on the amount of traffic they can handle — i.e. the amount of traffic they can aggregate/disaggregate and route. The amount of traffic to be faced by a given hub is the sum of the traffic related to terminals connected to it. Consider a set of terminals V, a subset $S \subseteq V$ connected to the considered hub and a traffic matrix t_{ij}, $(i, j) \in V \times V$. The amount of traffic to be faced by the hub is

$$\sum_{i \in S} \sum_{j \in V} t_{ij} + \sum_{i \in S} \sum_{j \in V \setminus S} t_{ji} = \sum_{i \in S} \left[\sum_{j \in V} (t_{ij} + t_{ji}) - \sum_{j \in S} t_{ji} \right].$$

In certain environments we can specify meaningful "profits" coming from connecting a terminal i to the hub; this leads to a knapsack-like quadratic model

$$\max \sum_{i \in V} \pi_i x_i \tag{2}$$

subject to

$$\sum_{i \in V} \left[\sum_{j \in V} (t_{ij} + t_{ji})\, x_i - \sum_{j \in V} t_{ji} x_i x_j \right] \leq B \tag{3}$$

$$x_i \in \{0, 1\}, \ \forall i \in V, \tag{4}$$

where $x_i = 1$ iff the terminal i is connected to the considered hub.

The profits assigned to the terminals may be different. With all equal profits we aim to maximize the number of connected terminals. In another situation, suppose that a terminal i^* is considered and that the goal is to minimize the total amount of traffic that it sends on the intra-hubs links — this helps in reducing network costs, since a flow depending cost is usually given between every pair of hubs. Thus, the aim is to minimize $\sum_{j \in V \setminus S} (t_{i^*j} + t_{ji^*})$. Since $\sum_{j \in V} (t_{i^*j} + t_{ji^*})$ is a constant, this is equivalent to maximize $\sum_{j \in S} (t_{i^*j} + t_{ji^*})$. This leads to a quadratic knapsack problem in which the profit of an item j is given by $t_{i^*j} + t_{ji^*}$. Finally,(2)–(4) is the pricing problem in a column generation approach for a class of Bin Packing Problems with quadratic formulation of the capacity constraint (see [2]). In this case, completely general profits are defined by the simplex multipliers at each iteration. Model (2)–(4) defines a special case of MNC, where terminals are mapped on graph nodes and weights $w_{ij} = t_{ij} + t_{ji}$ are associated to edges $\{i, j\}$: equation (1) enforces constraint (3).

In this work, we first show that the problem is strongly NP-complete (even for bipartite graphs) and that approximation ratios arbitrarily close to 2 are attainable in polynomial time. Then, we consider the weakly NP-complete special case where G is a tree and show that it is solvable in pseudopolynomial time by a dynamic programming approach. For this latter case an FPTAS is also provided.

2 Results for the general problem

We first recall the following problems which are instrumental for the results.

$0/1$ $KNAPSACK$ (KP). Given items $N = \{1, 2, \ldots, n\}$, item profits $p_1, \ldots, p_n$ and weights $W_1, \ldots, W_n$ and $b \geq 0$ determine $S \subseteq N$ such that $W(S) = \sum_{i \in S} W_i \leq b$ and $\sum_{i \in S} p_i$ is maximum (decision version: $\sum_{i \in S} p_i \geq K$ for a given K).

DENSE K-SUBGRAPH (DKS). Given a graph $G'(V', E')$, with edge weights W_{ij}, $k > 0$ and $Q > 0$, determine an $S' \subseteq V'$ such that $|S'| = k$ and

$$\sum_{\{i,j\} \in E(S')} W_{ij} \geq Q, \tag{5}$$

where $E(S') = \{\{i, j\} \in E' \colon i, j \in S\}$.

The KP problem is NP-complete in the ordinary sense [8], while the DKS problem is NP-complete in the strong sense even for unweighted bipartite graphs [4].

The decision version of MNC calls for finding an $S \subseteq V$ such that $w(S) \le B$ and $\pi(S) \ge R$ for a given R. The following proposition establishes the complexity of MNC even for the special case of bipartite graphs.

Proposition 1 *MNC is strongly NP-complete.*

Proof: Feasibility of any $S \subseteq V$ can be checked in $\mathcal{O}(|E|)$ time, hence $MNC \in NP$. A reduction from $DKS \propto MNC$ is immediate: set $V \equiv V'$, $E \equiv E'$, $W_{ij} \equiv w_{ij}$, $\pi_i = 1$ for all $i \in V$, $B = \sum_{\{i,j\} \in E} W_{ij} - Q$, $R = |V| - k$. An $S \subseteq V$ exists such that $\pi(S) = |S| = R$ and $w(S) \le B$ iff $S' = V \setminus S$ is a k-node subset satisfying (5). □

We now discuss approximate algorithms for MNC on arbitrary graphs. From now on we denote by S_A an approximate (heuristic) solution for MNC.

Proposition 2 *An approximate solution S_A such that*

$$\frac{\pi(S^*)}{\pi(S_A)} \le 3 + \varepsilon$$

is computable in polynomial time for each $\varepsilon > 0$, and this bound is tight.

Proof: Let I be an instance of MNC with a graph $G(V, E)$, profits π_i, $i \in V$, weights w_{ij}, $(i, j) \in E$, and capacity B. We define an instance I_1 of KP as follows:

$$N = \{1, 2, \dots, n\} = V, \tag{6}$$

$$p_i = \pi_i, \quad i \in V, \tag{7}$$

$$W_i = \sum_{(i,j) \in E} w_{ij}, \quad i \in V, \tag{8}$$

$$b = B. \tag{9}$$

As already remarked in the Introduction, we assume $W_i \le B$ for all i (if not, node i cannot be part of the solution and can be eliminated updating accordingly the other nodes). Any $S \subseteq N$ feasible for I_1 is also a feasible solution for I, since $W(S) = \sum_{i \in S} W_i \ge w(S)$. Let S_1^* be an optimal solution of I_1; we now prove that

$$\frac{\pi(S^*)}{\pi(S_1^*)} \le 3. \tag{10}$$

First note that, for S^* being feasible for I, we must have

$$W(S^*) = \sum_{i \in S} W_i \le 2B; \tag{11}$$

we introduce a second, restricted instance of KP defined by

$$\begin{aligned} N &= \{i \colon i \in S^*\}, \\ p_i &= \pi_i, \quad i \in N, \\ W_i &= \sum_{(i,j) \in E} w_{ij}, \quad i \in N, \\ b &= B. \end{aligned}$$

That is, I_2 is I_1 with the item set restricted to the (unknown) items in S^*; clearly,

$$\pi(S_1^*) \geq \pi(S_2^*),$$

where S_2^* is an optimal solution for I_2; for any $t \in S^* \setminus S_2^*$ we have

$$W(S_2^*) + W_t > B$$

(for optimality of S_2^*). We combine this with (11) and get

$$W(S^* \setminus (S_2^* \cup \{t\})) \leq B. \tag{12}$$

Recalling $W_t \leq B$, then both $\{t\}$ and $S^* \setminus (S_2^* \cup \{t\})$ are feasible for I_2; hence, for the optimality of S_2^*:

$$\pi_t \leq \pi(S_2^*), \tag{13}$$

$$\pi[S^* \setminus (S_2^* \cup \{t\})] \leq \pi(S_2^*). \tag{14}$$

Now, we have

$$\pi(S^*) = \pi(S_2^*) + \pi_t + \pi\left[S^* \setminus (S_2^* \cup \{t\})\right] \leq 3\pi(S_2^*) \leq 3\pi(S_1^*),$$

which proves (10).

Using (10), we define an approximation algorithm for I as follows: first, we construct I_1 in linear time; then we compute an approximate solution for I_1 with relative error at most $\varepsilon/3$. Since KP admits an FPTAS, the latter can be done in time $\mathcal{O}\left(\frac{n^3}{\varepsilon}\right)$. Finally, we set our S_A to the approximate solution for I_1. The guaranteed approximation ratio is then

$$\frac{\pi(S^*)}{\pi(S_A)} = \frac{\pi(S^*)}{\pi(S_1^*)} \cdot \frac{\pi(S_1^*)}{\pi(S_A)} \leq 3\left(1 + \frac{\varepsilon}{3}\right) = 3 + \varepsilon.$$

For the bound tightness, consider any instance of MNC where $V = \{1,2,3\}$, $E = \{(1,2),(1,3),(2,3)\}$, $\pi_1 = \pi_2 = \pi_3 = 1$, $w_{12} = w_{13} = w_{23} = \frac{B}{3}$; then in I_1 $W_1 = W_2 = W_3 = \frac{2}{3}B$, and $\pi(S_1^*) = 1$, while $\pi(S^*) = \pi(\{1,2,3\}) = 3$. $\square$

The approximation ratio of Proposition 2 can be further improved by solving n instances of KP instead of one. Given an instance I of MNC, for each node $k \in V$ define a reduced instance $I[k]$ where node k and all its incident arcs are removed from G, and the capacity is reduced to

$$B - \sum_{j:\ (k,j)\in E} w_{kj}.$$

Approximating all these instances delivers better approximate solutions for I.

Proposition 3 *An approximate solution S_A such that*

$$\frac{\pi(S^*)}{\pi(S_A)} \leq 2 + \varepsilon$$

is computable in polynomial time for each $\varepsilon > 0$, and this bound is tight.

Proof: The improved approximation algorithm works as follows. Given I, we define instances $I[k]$, $k \in V$ as above. For each $I[k]$, a KP instance $I_1[k]$ is created as in (6)–(9) — always stripping off node k and its incident arcs. Then we generate approximate solutions $S_1[k]$ for the $I_1[k]$'s with relative error bounded from above by $\varepsilon/2$; each $S_1[k]$ is also feasible for $I[k]$, and $S_1[k] \cup \{k\}$ is feasible for I. Finally, we set S_A to be the set $S_1[k] \cup \{k\}$ with the largest profit.

To establish the result, it is sufficient to prove

$$\pi(S^*) \leq 2 \max_{k\in V} \{\pi_k + \pi(S_1^*[k])\}\,. \tag{15}$$

Let $S^*[k]$ and $S_1^*[k]$ be optimal solutions for $I[k]$, $I_1[k]$, respectively. Note that, for all $k \in S^*$,

$$S^*[k] = S^*. \tag{16}$$

Let $\bar{k} \in S^*$ be such that

$$\pi_{\bar{k}} \geq \pi_i \qquad \text{for all } i \in S^*, \tag{17}$$

and consider instance $I_1[\bar{k}]$. We proceed similarly to the proof of Proposition 2; an instance $I_2[\bar{k}]$ of KP corresponding to $I_1[\bar{k}]$ restricted to the items in $S^*[\bar{k}] = S^*$ is defined. Let S_2^* be the optimal solution for $I_2[\bar{k}]$. Clearly, $\pi(S_2^*) \leq \pi(S_1^*[\bar{k}])$. For some $t \in S^* \setminus (S_2^* \cup \{\bar{k}\})$ we have

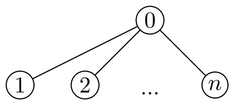
$$\pi(S^*) = \pi_{\bar{k}} + \pi(S_2^*) + \pi_t + \pi(S^* \setminus (S_2^* \cup \{t, \bar{k}\}));$$

also, by the same arguments used for equation (12), $S^* \setminus (S_2^* \cup \{t, \bar{k}\})$ is feasible for $I_2[\bar{k}]$; this implies

$$S^* \setminus (S_2^* \cup \{t, \bar{k}\}) \leq \pi(S_2^*).$$

Recalling (16), (17) and $t, \bar{k} \in S^*$, we have

$$\pi(S^*) = \pi_{\bar{k}} + \pi_t + \pi(S^* \setminus (S_2^* \cup \{t, \bar{k}\}) + \pi(S_2^*) \leq 2\,[\pi_{\bar{k}} + \pi(S_2^*)]\,,$$



$$\begin{array}{lcl}
N=\{1,2,\ldots,n\} & & V=\{0,1,2,\ldots,n\}\,,\ E=\{(0,i)\colon i=1,2,\ldots,n\} \\
W_1,\ldots,W_n & & w_{0i}=W_i,\ i=1,2,\ldots,n \\
p_1,\ldots,p_n & \Longrightarrow & \pi_0=0,\ \pi_i=p_i,\ i=1,2,\ldots,n \\
b & & B=b \\
K & & R=K
\end{array}$$

Figure 1: $KP \propto TMNC$

hence

$$\pi(S^*) \leq 2\left[\pi_{\bar{k}} + \pi(S_2^*)\right] \leq 2\left[\pi_{\bar{k}} + \pi(S_1^*[\bar{k}])\right]$$

which implies (15).

If all $\pi_k + \pi(S_1^*[k])$ are approximated within a maximum relative error $\varepsilon/2$,

$$\frac{\pi(S^*)}{\pi(S_A)} = \frac{\pi(S^*)}{\max_k \{\pi_k + \pi(S_1^*[k])\}} \cdot \frac{\max_k \{\pi_k + \pi(S_1^*[k])\}}{\pi(S_A)} \leq 2+\varepsilon.$$

For the bound tightness, consider an instance I of MNC on a complete graph where $V=\{1,2,3,4\}$, $E=\{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$, $\pi_i=1$ for all $i\in V$, $w_{ij}=\frac{1}{6}B$ for all $(i,j)\in E$. Each $I_1[k]$ has capacity $\frac{1}{2}B$, and optimal value 1, hence we get a heuristic solution with value 2, while $S^*=\{1,2,3,4\}$. $\pi(S^*)=4$. $\square$

3 The tree case

In the following we consider instances of MNC where the underlying graph is a tree $T(V,E)$ (Tree-MNC, $TMNC$). $TMNC$ is easily seen to be **NP**-complete: Figure 1 sketches a simple reduction $KP \propto TMNC$ where G is a star.

Let $V=\{1,2,\ldots,n\}$; without loss of generality, we assume that nodes are numbered so that each node $k\in V$ is connected to "children" nodes $\Gamma_k=\{k_1,k_2,\ldots,k_m>k\}$ (m depending on k) and, if $k\neq 1$, to a unique "parent" $k'<k$; node 1 is the root of the tree. Also, let $U\subset V$ be the set of leaves of T.

3.1 Dynamic programming

We develop a dynamic programming recursion for $TMNC$, which is in the spirit of those used for KP. Particularly, we remark that the recursion parameter could be (i) the maximum capacity allowed for a partial solution, or (ii) the total profit associated with the partial solution. We analyze in detail approach (ii).

Note that, for a node k to be part of a feasible solution $S \subseteq V$ a capacity

$$a_k = w(\Gamma_k) \text{ or } b_k = w_{k'k} + w(\Gamma_k)$$

must be allotted to k, depending whether $k' \in S$ or $k' \notin S$.

Solutions $S \subseteq V$ are build recursively, in stages; the recursion alternates *major stages* and *minor stages*. Each major stage is associated with a node $k \in V$, and decides whether k must be part of a solution or not; then a sequence of minor stages is dedicated to build the partial solution for the forest of subtrees $T_{k_1}, T_{k_2}, \ldots, T_{k_m}$ rooted at nodes $\{k_1, k_2, \ldots, k_m\} = \Gamma_k$. Each minor stage composes solutions for T_{k_j} and the forest $\{T_{k_{j+1}}, \ldots, T_{k_m}\}$.

A decision taken in a major stage for node k has consequences up to the major stages for $k_1, \ldots, k_m$ only; specifically, if k is to be brought into the solution, each k_j will need only a_{k_j} units of capacity to be allotted in order to join the solution; otherwise, it will need b_{k_j} units.

We state the recursion for $TMNC$ by using four functions.

- $\Phi_k(s)$ is defined as the minimum capacity to be allotted to an S including only nodes from the subtree rooted at k, and such that $\pi(S) = s$, assuming a_k units of capacity are required for k;

- $\phi_{k_j}(s)$ is defined as the minimum capacity to be allotted to an S including only nodes from the forest made up of the subtrees rooted at $k_j, \ldots, k_m$, and such that $\pi(S) = s$, assuming a_{k_i} units are required for each $k_i \in \{k_j \ldots k_m\}$;

- $\Psi_k(s)$ is defined as the minimum capacity to be allotted to an S including only nodes from the subtree rooted at k, and such that $\pi(S) = s$, assuming b_k units of capacity are required for node k;

- $\psi_{k_j}(s)$ is defined as the minimum capacity to be allotted to an S including only nodes from the forest made up of the subtrees rooted at $k_j, \ldots, k_m$, and such that $\pi(S) = s$, assuming b_{k_i} capacity of units are required for each $k_i \in k_j \ldots k_m$.

Note that, at the major stage corresponding to $\Phi_k(s)$ the decision is whether to bring or not k into the solution; the optimal decision is the one which gives minimum weight between $\phi_{k_1}(s - \pi_k) + a_k$ — note that each k_j will then require a_{k_j} units of capacity in this part of the recursion — and $\psi_{k_1}(s)$; in the latter case each k_j will require b_{k_j} units in this part of the recursion. Similar observations hold for $\Psi_k(s)$.

Concerining the minor stages, the minimum $\phi_{k_j}(s)$ is a composition of solutions for T_{k_j} and $\{T_{k_{j+1}}, \ldots, T_{k_m}\}$; the total profit has to be partitioned, with u units to the T_{k_j} solution and $s - u$ to the residual forest, hence we minimize over all the values $u = 0, \ldots, s$. Proceed similarly for $\psi_{k_j}(s)$.

Given the above definitions, the problem of clustering nodes with profit s and minimum weight writes out as

$$\Phi_k(s) = \min\{\phi_{k_1}(s - \pi_k) + a_k, \psi_{k_1}(s)\} \qquad \text{for } k \in V \setminus U, \tag{18}$$

$$\phi_{k_j}(s) = \min_{0 \le u \le s} \left\{\Phi_{k_j}(u) + \phi_{k_{j+1}}(s - u)\right\} \qquad \text{for } j = 1, \ldots, m-1,\ k_j \in \Gamma_k, \tag{19}$$

$$\Psi_k(s) = \min\{\phi_{k_1}(s - \pi_k) + b_k, \psi_{k_1}(s)\} \qquad \text{for } k \in V \setminus U, k \neq 1, \tag{20}$$

$$\psi_{k_j}(s) = \min_{0 \le u \le s} \left\{\Psi_{k_j}(u) + \psi_{k_{j+1}}(s - u)\right\} \qquad \text{for } j = 1, \ldots, m-1,\ k_j \in \Gamma_k, \tag{21}$$

where for each $s \in [0, \Pi],\ \Pi = \sum_{i \in V} \pi_i$

$$\phi_{k_m}(s) = \Phi_{k_m}(s)$$
$$\psi_{k_m}(s) = \Psi_{k_m}(s)$$

and with initial conditions on leaves $k \in U$:

$$\Phi_k(s) = \begin{cases} a_k = 0 & \text{if } s \in \{0, \pi_k\}, \\ \infty & \text{otherwise,} \end{cases} \tag{22}$$

$$\Psi_k(s) = \begin{cases} b_k = w_{k'k} & \text{if } s = \pi_k, \\ 0 & \text{if } s = 0, \\ \infty & \text{otherwise.} \end{cases} \tag{23}$$

Equations (18) and (20) correspond to major stages, while (19) and (21) correspond to minor stages.

This set of functions can be computed, for $0 \leq s \leq \sum_{i \in V} \pi_i$ from the leaves up to the root; the maximum profit for the instance at hand is given by

$$\pi(S^*) = \max\{s\colon \Phi_1(s) \leq B\};$$

the optimal solution S^* can be recovered by backtracking.

Proposition 4 *Recursion* (18)–(23) *optimally solves $TMNC$ in time $\mathcal{O}(n^3\pi^2_{\max})$.*

Proof: Optimality follows from the discusssion above (one could proceed, more formally, by induction from the leaves to the root). For the time bound: for each s, there are no more than n major stages ($\Phi_k(s)$ and $\Psi_k(s)$ to compute), each computed in constant time by (18) or (20); for each major stage k there are exactly $|\Gamma_k|$ minor stages ($\phi_{k_j}(s)$ and $\psi_{k_j}(s)$ to compute), each requiring time $\mathcal{O}(s)$. Hence for a fixed s a time proportional to $n+s\cdot\sum_{k\in V}|\Gamma_k| = \mathcal{O}(n+ns) = \mathcal{O}(ns)$ is required. There are $\sum_{i\in V}\pi_i$ values of interest for s, hence the total time is proportional to

$$n\left(\sum_{i\in V}\pi_i\right)^2 = \mathcal{O}\big(n^3\pi^2_{\max}\big). \quad \square$$

Remark 1: One could design the recursion with the capacity as parameter; then $\Phi_k(s)$ would be the maximum profit attainable with a subset S of nodes from the subtree rooted in k and such that $w(S) \leq s$, assuming a_k units of space are required for taking node k in S; the other definitions should be modified similarly. Along the same lines as above, one can prove that such recursion solves MNC on trees in time $\mathcal{O}\big(nB^2\big)$.

Remark 2: If nodes profits and/or edge weights are unitary ($\pi_{\max} = 1$ and/or $B < \frac{n(n-1)}{2}$), the complexity becomes polynomial. This induces, as a byproduct, that DKS on trees is polynomially solvable. We show this for DkS with unitary weights (the so-called *Densest* k-Subgraph problem in [1]), but through dicothomic search also the case with arbitrary weights ((the so-called *Heaviest* k-Subgraph problem in [1]) is polynomially solvable. Indeed, given an instance on a tree T of DkS, it is sufficient to optimally solve the corresponding $TMNC$ for all values of B (that is to solve $TMNC(B)$ for $1 \leq B \leq |E|$) and check the corresponding solution values. This can be done in polynomial time as $|E| \leq \frac{n(n-1)}{2}$. The value B^* such that $OPT(TMNC(B^*)) = n-k$ and $OPT(TMNC(B^*-1)) = n-k-1$, gives the optimal solution value of DkS.

3.2 A Fully Polynomial Time Approximation Scheme

Given the dynamic programming recursion (18)–(23), an FPTAS for MNC on trees is defined as follows (note that the existence of an FPTAS can also be derived following [11]). Let I be an instance of $TMNC$ with a tree $T(V, E)$, profits π_i, $i \in V$, w_e, $e \in E$ and B; let ε be the desired bound on the relative error, $\pi_{\max} = \max\{\pi_i \colon i \in V\}$, and $q = \lceil 1/\varepsilon \rceil$, $\bar{M} = \frac{\pi_{\max}}{n(1+q)}$.; we construct a scaled instance I' with the same tree, capacity $B' = B$ and weights $w'_e = w_e$, and scaled profits $\pi'_i = \lfloor \pi_i/\bar{M} \rfloor$. Each solution feasible for I' is also feasible for I. Then I' is solved by the $\mathcal{O}\left(n^3\pi^2_{\max}\right)$ dynamic programming recursion, and the optimal solution S' for I' is returned as heuristic solution for I.

Proposition 5 $\dfrac{\pi(S^*)}{\pi(S')} \leq 1 + \varepsilon$, *and S' can be computed in time* $\mathcal{O}\left(\frac{n^5}{\varepsilon^2}\right)$.

Proof: The proof exactly mimics the one given for KP (see for example [8]). For S^* and S', we have

$$\sum_{i\in S'} \pi_i \geq \sum_{i\in S'} \bar{M}\pi'_i \geq \sum_{i\in S^*} \bar{M}\pi'_i \geq \sum_{i\in S^*} (\pi_i - \bar{M}) \geq \sum_{i\in S^*} \pi_i - n\bar{M},$$

hence

$$\pi(S') \geq \pi(S^*) - n\bar{M}.$$

Then,

$$\begin{aligned}\frac{\pi(S^*)}{\pi(S')} &\leq 1 + \frac{n\bar{M}}{\pi(S')} \leq 1 + \frac{n\bar{M}}{\pi(S^*) - n\bar{M}} \leq 1 + \frac{n\bar{M}}{\pi_{\max} - n\bar{M}} = 1 + \frac{1}{q} \\ &\leq 1 + \varepsilon.\end{aligned}$$

The time bound follows from $\pi'_{\max} = \lfloor \pi_{\max}/\bar{M} \rfloor = \lfloor n(1+q) \rfloor$, hence the recursion takes time

$$\mathcal{O}\left(n^3(\pi'_{\max})^2\right) = \mathcal{O}(n^5q^2) = \mathcal{O}\left(\frac{n^5}{\varepsilon^2}\right). \quad \square$$

3.3 Conclusions

We have introduced in this paper the Maximum Node Clustering showing that the problem is strongly NP-complete and can be polynomially approximated within a ratio arbitrarily close to 2. We have then considered the NP-complete special case when the graph is a tree and proposed a pseudopolynomial time exact dynamic programming algorithm and a fully polynomial time approximation scheme. Several related issues are worthy of future research with particular reference to

- The approximability of the MNC problem on bipartite graphs: it is easy to prove that, for this special case, a faster and simpler way (with respect to Proposition 3) to get an approximation ratio arbitrarily close to 2 is the following: solve two instances I_1 of KP where in the first (second) case all nodes of the first (second) partition are excluded a priori from S^* and take the best solution. This result is essentially due to the bipartite structure of the problem. Is it possible to better exploit this structure to improve upon this ratio?

- The Worst-case complexity (see [12] and its relevant notation). It is quite straightforward to show that the MNC problem on bipartite graphs can be solved with complexity $O^*(2^{\frac{3}{4}n})$. Indeed, if for all nodes of the smallest partition (with size $\alpha \le \frac{n}{2}$) it is known whether they belong to S^*, the remaining problem corresponds to a KP problem with $\beta = n - \alpha \ge \frac{n}{2}$ variables. We know from [10] that a KP problem with β variables is exactly solvable with complexity $O^*(2^{\frac{\beta}{2}})$. But then, by running the corresponding KP problem for all the 2^α different cases as far as the smallest partition is concerned, we get a complexity $O^*(2^\alpha * 2^{\frac{\beta}{2}})$ where the worst case occurs for $\alpha = \beta = \frac{n}{2}$ that is $O^*(2^{\frac{3}{4}n})$. It is worthy to see if this result can be improved to close the gap with respect to the $O^*(2^{\frac{n}{2}})$ bound provided by the KP problem and whether it is possible to get anything better than the trivial $O^*(2^n)$ for the general problem.

References

[1] A. Billionnet, F. Roupin, "A Deterministic Approximation Algorithm for the Densest k-Subgraph Problem", (2005), *4OR*, forthcoming,

[2] G. Carello, "Hub Location Problems in Telecommunication Networks", Ph.D. Thesis, Politecnico di Torino (2004), available at http://www.orgroup.polito.it/members/carello/phdthesis.pdf.

[3] G.Carello, H. Yaman, "Solving the Hub Location Problem with Modular Link Capacities", (2004), *Computers and Operations Research*, forthcoming, available at Science Direct - Articles in Press.

[4] D.G. Corneil, Y. Perl, "Clustering and domination in perfect graphs", *Discrete Applied Mathematics*, 9(1) (1984), 7-39.

[5] M. Demange, V.Th. Paschós, "On an approximation measure founded on the links between optimization and polynomial approximation theory", *Theoretical Computer Science*, 156 (1996), 117-141.

[6] U. Feige, M. Langberg, "Approximation Algorithms for Maximization Problems arising in Graph Partitioning", *Journal of Algorithms*, 41 (2001), 174-211.

[7] U.Feige, G. Kortsarz, D. Peleg, "The Dense k-Subgraph problem", *Algorithmica*, 29 (2001), 410-421.

[8] M. Garey, D. S. Johnson, "Computers and Intractability: a guide to the theory of NP-completeness", 1977.

[9] E. Gourdin, M. Labbé, H. Yaman, "A branch and cut algorithm for hub location problems with single assignment", *Mathematical Programming*, forthcoming, available at http://smg.ulb.ac.be/.

[10] E. Horowitz, S. Sahni, "Computing partitions with applications to the knapsack problem", *Journal of the ACM*, 21 (1974), 277-292.

[11] G.J. Woeginger, "When does a dynamic programming formulation guarantee the existence of a fully polynomial time approximation scheme (FPTAS)?", *INFORMS Journal on Computing*, 12 (2000), 57-75.

[12] G.J. Woeginger, "Exact algorithms for NP-hard problems: a survey", in: *Combinatorial Optimization - Eureka! You shrink!*, M. Juenger, G. Reinelt and G. Rinaldi (eds.), LNCS 2570, Springer (2003), 185-207.

[13] H. Yaman, "Concentrator Location in Telecommunication Networks". Ph.D. Thesis, Université Libre de Bruxelles (2002).