Approximation algorithms and relaxations for a service provision problem on a telecommunication network

Citation for published version (APA):
Memorandum COSOR 98-04

Approximation algorithms and relaxations for a service provision problem on a telecommunication network

S. Dye, L. Stougie and A. Tomasgard

Eindhoven, March 1998
The Netherlands
Approximation algorithms and relaxations for a service provision problem on a telecommunication network *

Shane Dye, Leen Stougie, Asgeir Tomasgard

Abstract

Modern distributed networks have widely extended the possibilities of the telecommunication industry for offering a wide variety of services directly or indirectly by facilitating them for other service providers. These new opportunities require a shift of focus in research from traditional transportation of information to processing of information. This paper considers the problem of allocating applications for services (in the sense of software) to the computing nodes of the distributed network, so as to provide as much of the demand for the services as possible. The service provision problem is formulated as an integer linear programming model and is shown to be NP-hard.

We exploit similarities to the well-known (multiple) knapsack problem in devising approximation algorithms and analysing their performance from a worst case point of view. Among others a fully polynomial time approximation scheme is presented for the case with one computing node. The other main results of the paper concern the derivation of lower bounds on the optimal solution via LP and Lagrangian relaxations.

Keywords: distributed networks, telecommunication, service provision, internet, approximation algorithms, worst-case analysis, LP relaxation, Lagrangian relaxation, knapsack problem.

1 Introduction

This paper considers a service provision problem in a distributed telecommunication network. Typical services in such a network may be different transportation services, teleconferencing, televoting, video on demand and so on. In the near future there will be an explosion of new telecommunication applications as many of the differences between the telecommunication industry and the computer industry disappear. Many of the new services emerging from this will be more processing oriented than traditional telecommunication services are.

Technically, we use the term service to encompass a set of software applications running on computing nodes in the network. Clients demand services and this paper is concerned with the problem of allocating a limited resource to the services in order to maximize the profit from meeting this demand. We focus on allocating resources that are present in the computing nodes, rather than in the transportation network. We will start by explaining the background of the problem and the motivation for the model we study.

Traditionally, transportation capacity has been considered as the limited resource in the provision of (telecommunication) services on telecommunication networks. Recently, practitioners in the telecommunication industry are foreseeing that the growth in service provision will come from services that require a substantial amount of computing resources. It is their opinion that resources like processing capacity or memory space of the computers in the network will become limiting factors. In fact, this paper is a result of a research project in cooperation with and partly sponsored by the Norwegian telecommunication company Telenor.

This change of focus from transportation to the computing nodes is an impact of several technological developments. ATM technology and fibre optics lead to more flexible and lower priced...
transportation of high capacity (see for example [4, 15]). At the same time digital technology reduces the gap between the telecommunication and the computer industry. Standardization attempts make it easier to offer services over a wide variety of hardware platforms and thereby help remove differences between telecommunication and information services.

Several standardization initiatives for telecommunication networks with distributed processing capabilities exist. One of the most prominent achievements in this direction is the Telecommunications Information Networking Architecture Consortium (TINA-C), see for example [3, 17]. Distributed processing and standardization increases flexibility in service provision and resource allocation.

An example of the increased flexibility induced by distributed processing is the approach of [18, 19, 20], where a distributed network is modelled based on TINA-C. Typically the distributed network can be described in three different planes, as in Figure 1. In the upper plane we have a set of software applications. The computing nodes, where the applications reside, can be seen as the "glue" which binds the distributed application architecture and the underlying network architecture together. Examples of interaction between applications are shown by the solid lines between them. This interaction usually creates information flows through the computers and the underlying networks. The solid lines in the computing nodes plane describe virtual links between pairs of computers. We use the term virtual to indicate that there does not necessarily exist a corresponding direct physical link between the two nodes but that it is possible to transport information between them. The real transportation of information happens in physical links in the networks described in the network (lower) plane. In this plane the nodes describe physical networks and the solid lines indicate which networks are connected to each other (without saying how they are connected).

In this paper we consider the problem that arises when computing resources are the limiting factor in providing services. One main implication of TINA-C is the existence of a standardized software platform where all nodes look the same to the applications. In theory any service can be put on any node. In [20] it is also argued that the functionality of the TINA-C standard un-
under specific assumptions of the quality of transportation networks, separates the service provision problem from the underlying transportation problem. In short the requirement is that: within regions of the global network, the transportation networks are designed to keep the time delay acceptable on transportation of generated information and to ensure the correct quality of service (QoS), independent of the allocation of requests for services to nodes. Notice that we are investigating resource allocation at an operational level. It is unlikely that in considering investments the same separation between the transportation and the computing phase of service execution is possible.

Note that the distributed network model described above is a useful model for the Internet. The upper plane corresponding to the Web; the middle plane representing the servers; and the bottom plane being the international structure of telephone and data networks. Internet as of today does not possess all the properties of a distributed telecommunication network, for example when it comes to real-time applications, transportation capabilities and distributed processing functionality. However, Internet, if development proceeds, will probably not be far away from the conceptual scheme described above. In addition Internet is the best existing example of the convergence of telecommunication industry and the computer industry. It illustrates perfectly a network which traditionally was used for transportation of information, while many of today’s services has its main focus on processing information. Throughout this paper we will follow the general conceptual description above, and even with the lack of functionality in today’s Internet, we will see that the problem discussed has a lot of relevance for Internet’s future.

We now discuss the service provision procedure and the resulting in more detail. As mentioned before services are constituted by a set of applications. We assume that the applications which together constitute a service may be distributed on several computing nodes in the network. The validity of this assumption is supported by TINA-C. However, sets of applications that are always distributed together, will be considered as entities called subservices. A subservice can be used by several services at the same time without these being aware of each other. All services are built from subservices, as shown in Figure 2. Here service A is built from subservices “sound” and “video”. Service B is built from subservices “sound”, “data” and “video”. Also several instances of the same subservice may reside at different nodes, as for the subservice “sound” in the figure. The arcs in the lower half of Figure 2 represent the different instances.

In [20] it is argued that under some market assumptions it makes sense to model demand as the amount of the limiting resource requested by the different subservices, thereby only implicitly including the complete services in the model. This is done in the remaining sections of the paper.

The important characteristics of a service come from the subservices it uses. A subservice
requires node resources from the moment it is installed, even without serving any customers. There
is a fixed amount of resource use induced by service set-up, administration and the subservices’
ability to immediately serve incoming requests. From now on we call this subservice dependent
amount its installation requirement. A request for a subservice also induces a requirement for node
resources. We will call this utilization requirement. Thus, any request for a service, implies requests
for subservices, and in this way determines the needs this generates from the node resources. When
a subservice is installed at a node, we assume there is no limit on the number of requests that can
be served by it, except those by the invoked availability of the node resources.

Clearly, installing all subservices in the network implies the ability to meet all types of demand.
However, the total installation requirement induced by this may use so much of the nodes’ resources
that too little is left to actually meet all demand. Therefore, it may be profitable not to install
some of the subservices. On the other hand, not installing subservices implies an inability to
provide some services, and hence the inevitable loss of requests for those services. The resulting
trade-off makes the service provision problem a hard combinatorial problem. In fact it has many
similarities with the well-known (multiple) knapsack problem [12].

There are two different types of “actors” in the telecom markets for which the service pro­
vision problem is in particular interesting to examine: the service providers offering services to
customers and the network providers offering processing and transportation functionality to the
service providers.

Let us first assume that a service provider can rent space on a number of the network providers’
computers. As a consequence of the separation of the node problem and the transportation
problem, demand can be aggregated over all customers and all geographical locations for any
subservice. What is left is then a set of computers with given capacity and a set of subservices
with their demands. The service provider is free to utilize the capacity he has hired in any way
he likes in order to meet this demand in a profit maximizing way. We denote this problem the
multiple node service provision problem.

A second possibility is that the service provider hires a total amount of capacity from a network
provider. In this case he does not necessarily know if the capacity is on one or several computers
and he does not care. The problem he faces is the same as earlier, with the difference that he
now has only one node. We denote this problem the single node service provision problem. In
this case the network providers problem of how to replicate subservices over his nodes to meet all
service providers demand is now exactly the multiple node service provision problem. Note that
in the first situation the service provider gets the disadvantages of bearing the replication costs for
services (in terms of the installment requirements). In the second situation these disadvantages
are carried over to the service provider. The service provider sees only one “virtual” node and
hence installs each subservice at most once. The advantage the network provider gains, is of course
that he is free to utilize the same subservice instance for several service providers.

In [5] it is shown that deciding feasibility of the multiple node service provision in the sense
that no request needs to be rejected is NP-complete. In this paper we study optimization of
both the single node and the multiple node problem. A (mixed) integer linear programming
model for the service provision problem is given in Section 2 together with some results on its LP
relaxation. Worst-case performance results for greedy algorithms are derived in Section 3. Then,
in Section 4, for the single node problem we device an exact pseudo-polynomial algorithm and a
fully polynomial approximation scheme. In deriving these results we rely on the similarity of our
problems to the (multiple) knapsack problem [12]. The strength of Lagrangian relaxations of the
problems, especially in comparison to the LP relaxation is the subject of Section 5. Conclusions
and directions for future research form Section 6.

For more details on modelling of service provision problems see [18, 19, 20]. They contain
a set of problem models emerging from the scenario that computing capacity becomes an im­
portant issue for a telecommunication company. Next to service provision models, models for
strategical problems like investment in node (computer) and transportation capacity, and loca­
tion of computing nodes are considered there. Apart from these papers the authors do not know
any of optimization approaches to model resource allocation at computing nodes in a distributed
processing environment.
Next to this there are some papers dealing with aspects of specific new services introduced into modern telecommunication networks, for example on video on demand [1, 2]. These papers focus on transportation issues of resource allocation, though, and not on the computing nodes. In [10] distribution of the workload in a network of computers is discussed, but here unsplitable jobs are processed on a single machine, hence the distributed processing aspects are not really fully explored. For an overview of optimization models considering transportation issues of telecommunication see for example [7, 8, 16].

2 The model

In this section we present the mathematical programming formulations of the multiple node and single node service provision problem. Afterwards we comment on relationship of our problems with other problems known from the literature. The section is finished with an exploration of the Linear Programming (LP) relaxations of the problems, which is a preliminary to the worst-case performance analysis of approximation algorithms in the following section.

In the following we use $\mathbb{R}_+, \mathbb{Z}_+$ and $\mathbb{Q}_+$ to denote non-negative real numbers, integers and rational numbers, respectively.

2.1 Mathematical formulation

The formulations given here can also be found in [20]. We first consider the multiple node problem. Suppose there are $m$ computing nodes and $n$ subservices. Let $s_i \in \mathbb{Z}_+^m$ be the capacity of node $i$, $i = 1, ..., m$. For each subservice $j = 1, ..., n$ we define the parameters: $d_j \in \mathbb{Z}_+$, the demand for subservice $j$; $r_j \in \mathbb{Z}_+$, its installment requirement; and $q_j \in \mathbb{Z}_+$ the profit from serving one unit of demand for subservice $j$. We assume that each unit of demand for any subservice requires unit capacity on the node that meets it. Notice that we assume all parameter values to be integral.

We introduce two types of decision variables. The 0-1 variables $z_{ij}$ are to decide if subservice $j$ is installed on node $i$, in which case $z_{ij} = 1$, or not, reflected by $z_{ij} = 0$, $i = 1, ..., m$, $j = 1, ..., n$. Then, the variables $x_{ij}$ give the amount of demand for subservice $j$ met by computing node $i$, $i = 1, ..., m$, $j = 1, ..., n$.

The objective is to maximize total profit subject to the following restrictions. The first set of constraints that we call the capacity constraints say that the capacity of each node is not exceeded, i.e., the sum of the installment requirements and the total demand met at a node is not greater than its capacity. The demand constraints make that total demand cannot be exceeded for each subservice. The third set, the installation constraints make that demand for a subservice can be met at a node only if the subservice is installed at that node.

The resulting mathematical programming formulation is then given by

$$\begin{align*}
\max & \quad \sum_{i=1}^{m} \sum_{j=1}^{n} q_j x_{ij} \\
\text{s.t.} & \quad \sum_{j=1}^{n} r_j z_{ij} + \sum_{j=1}^{n} x_{ij} \leq s_i, \quad i = 1, ..., m \\
& \quad \sum_{i=1}^{m} x_{ij} \leq d_j, \quad j = 1, ..., n \\
& \quad M_{ij} z_{ij} - x_{ij} \geq 0, \quad i = 1, ..., m, \quad j = 1, ..., n \\
& \quad z_{ij} \in \{0, 1\}, \quad x_{ij} \geq 0, \quad i = 1, ..., m, \quad j = 1, ..., n
\end{align*}$$

The installation constraints are to require that each $x_{ij}$ can be positive only if $z_{ij} = 1$. We have to choose the constant $M_{ij}$ such that it is an upper bound on $x_{ij}$, $i = 1, ..., m$, $j = 1, ..., n$. A tight choice is $M_{ij} = \min\{d_j, \max\{s_i - r_j, 0\}\}$, $i = 1, ..., m, j = 1, ..., n$.

Moreover, we introduce the following assumptions to circumvent trivialities in the feasible solutions:
A1 \[ d_j \leq \sum_{i=1}^{m} \max\{0, s_i - r_j\}, \quad j = 1, \ldots, n, \]

To avoid the situation where all solutions must reject some demand for subservice \( j \).

A2 \[ r_j < \max_{1 \leq i \leq m} s_i, \quad j = 1, \ldots, n, \]

To avoid the case where there is no node with sufficient capacity to install subservice \( j \).

A3 \[ s_i > \min_{1 \leq j \leq n} r_j, \quad i = 1, \ldots, m, \]

To avoid having a node whose capacity does not suffice to install any subservice.

The single node version has just one computing node \( (m = 1) \). Its mathematical programming formulation is obtained by dropping the index \( i \) in the above formulation. The demand constraints become upper bounds on the decision variables. We write out the formulation for easy reference.

\[
\begin{align*}
\min & \quad \sum_{j=1}^{n} q_j x_j \\
\text{s.t.} & \quad \sum_{j=1}^{n} r_j z_j + \sum_{j=1}^{n} x_j \leq s \\
& \quad M_j z_j - x_j \geq 0, \quad j = 1, \ldots, n \\
& \quad z_j \in \{0,1\}, \quad 0 \leq x_j \leq d_j, \quad j = 1, \ldots, n.
\end{align*}
\]

In this model we use use \( M_j = d_j \) (cf. Assumption A1).

One might argue that the variables \( x_{ij} \) should be integer valued. However, in any basic optimal solution they will automatically be integral given that the problem coefficients are integers.

2.2 Relationship to other problems and complexity

The multiple node service provision problem is exactly a transportation network with so-called supply eating arcs. Dye et al. [5] discuss the question of determining whether a solution exists for which all demand is met in the multiple node problem. They also provide a brief discussion of the similarities of this problem with other classical mixed integer programs.

In the following sections, we utilize similarity between the multiple node service provision problem and the multiple knapsack problem. Imposing the additional restriction that \( x_{ij} = d_j \) if and only if \( z_{ij} = 1 \) transforms (1) into a multiple knapsack problem with weights \( w_j = d_j + r_j \) and profits \( p_j = q_j d_j, \quad j = 1, \ldots, n \), and container capacities \( c_i = s_i, \quad i = 1, \ldots, m \). The multiple node service provision problem is, therefore, a relaxation of multiple knapsack. Furthermore, in the proof of the following proposition we show that the multiple knapsack problem with \( m \) containers is polynomially reducible to the \( m \)-node service provision problem.

Proposition 1 The single node service provision problem is NP-hard and the multiple node service provision problem is NP-hard in the strong sense.

Proof: Consider the following reduction from the integer multiple knapsack problem with \( m \) containers having capacities \( c_i, \quad i = 1, \ldots, m \), and \( n \) items each having an associated profit, \( p_j \), and weight, \( w_j \), for \( j = 1, \ldots, n \). Transform this to an integral instance of the \( m \)-node service provision problem with \( n \) subservices by setting \( s_i = 2nc_i + n \) for \( i = 1, \ldots, m \), \( q_j = p_j \), \( r_j = 2nw_j \) and \( d_j = 1 \) for \( j = 1, \ldots, n \). A feasible solution, \( y^* \) say, of the multiple knapsack problem corresponds to a solution, \( (x^*, z^*) \) say, of the service provision problem, having the same objective value, defined by \( x_j^* = z_j^* = y_j^+, \quad j = 1, \ldots, n \). Any basic feasible solution of the service provision problem, \( (x^+, z^+) \), has \( x^+ \) integer valued and has a corresponding feasible solution to the multiple knapsack problem, \( y^+ \), with the same objective value defined by \( y^+ = z^+ \).
The single node problem admits a pseudo-polynomial algorithm and in Section 4.3 we present a fully polynomial approximation for this problem. In contrast, no pseudo-polynomial algorithm can exist for the multiple node service provision problem unless \( P = NP \).

In the following sections many of our results parallel those for the multiple knapsack problem. Notice, however, that in multiple knapsack \( m \leq n \) whereas this does not need to be the case for the multiple node service provision problem and \( m \) will often explicitly enter into the complexity. However, for our underlying application, in general \( m \) will be significantly less than \( n \).

2.3 The LP relaxation

Consider the LP relaxation of the single node problem, (2), obtained by replacing the 0-1 restriction on \( z_j \) by \( 0 \leq z_j \leq 1 \), \( j = 1, \ldots, n \). Since \( q \geq 0 \), the installation constraints will be binding at any optimal LP solution implying that we may replace \( z_j \) by \( d_j z_j \) (having chosen \( M_j = d_j \), \( j = 1, \ldots, n \)). This yields a continuous knapsack problem with weights \( r_j + d_j \) and profits \( q_j d_j \).

Let

\[
v_j = \frac{q_j d_j}{r_j + d_j},
\]

and assume that the numbering of the subservices is already according to non-increasing \( v \)-value. Then, the optimal solution of the LP relaxation is given by

\[
z_j = \begin{cases}
1 & j < \ell, \\
\frac{s*}{r_j + d_j} & j = \ell, \\
0 & j > \ell.
\end{cases}
\]

and \( s* = s - \sum_{j=1}^{\ell-1} (r_j + d_j) \) (4)

where \( \ell \) is the largest index \( k \) for which \( \sum_{j=1}^{k-1} (r_j + d_j) \leq s \), the so-called critical item. We note that when subservices are initially unordered, an optimal solution can be computed in \( O(n) \) time using the critical item partitioning routine from [12, Section 2.2.2].

That \( Z^{LP}/Z^{OPT} \leq 2 \) follows directly from the corresponding result for the knapsack problem [12, Section 2.2.1]. The upper bound on the ratio is shown to be asymptotically tight by a series of problems (adapted from [12]) with \( n = 2 \), \( q_1 = q_2 = k \), \( r_1 = r_2 = k \), \( d_1 = d_2 = 1 \) and \( s = 2k + 1 \), \( k = 1, 2, \ldots \). The ratio gets arbitrarily close to 2 choosing \( k \) sufficiently large.

For the LP relaxation of the multiple node problem, (1), consider the case in which \( M_{ij} = d_j \), \( i = 1, \ldots, m \), \( j = 1, \ldots, n \). Again, the installation constraints will be binding in an optimal solution, allowing us to replace \( x_{ij} \) by \( d_j z_{ij} \) to obtain an instance of the continuous multiple knapsack problem with weights \( r_j + d_j \) and profit \( q_j d_j \). As with multiple knapsack [12] this is equivalent to the LP relaxation of the surrogate relaxation formed by considering a single node of capacity \( \sum_{i=1}^{m} s_i \).

An optimal solution can be calculated in \( O(m + n) \) time following the reasoning in [12, Section 6.3]. Again assume that the items are ordered according to non-increasing ratios \( v_j \). The subservice demands to be met by each computing node are found sequentially by identifying the critical item relative to supply node \( i \), \( k_i \), for \( i = 1, \ldots, m \), where

\[
k_i = \min \left\{ k \mid \sum_{j=1}^{k} (r_j + d_j) > \sum_{t=1}^{i} s_t \right\}.
\]

(5)

\( k_m \) is the critical demand of the surrogate relaxation (assuming for simplicity that

\[
\sum_{j=1}^{n} (r_j + d_j) \geq \sum_{i=1}^{m} s_i.
\]

(6)
Also here the fact that $Z^{LP}/Z^{OPT} \leq m+1$ comes directly from the corresponding multiple knapsack result [12, Theorem 6.3]. Tightness is shown by the series of problems (adapted from [12]) with $n > 2m$, $s_1 = 2k (k \geq 2)$, $s_2 = \cdots = s_m = r_1 = \cdots = r_{m+1} = k$, $r_{m+2} = \cdots = r_n = k-1$, $d_1 = \cdots = d_n = 1$, $q_1 = \cdots = q_{m+1} = k$ and $q_{m+2} = \cdots = q_n = 1$. $Z^{OPT} = k + m - 1$ while $Z^{LP} = (m+1)k^2/(k+1)$ and the ratio gets arbitrarily close to $m+1$ for sufficiently large $k$.

When $M_{ij} < d_j$ the installation constraints keep on being binding so that we might replace $x_{ij}$ by $M_{ij}z_{ij}$. This will generally lead to a lower optimal value of the resulting LP relaxation. Furthermore, the LP relaxation is in this case not necessarily equivalent to the LP relaxation of the surrogate relaxation and therefore does not lend itself to such a simple solution structure.

3 Greedy heuristics

From Section 2.3 we see that the LP relaxation is solved by ranking demands by non-increasing $v$-value (3) and choosing to meet those with the highest value first. This leads naturally to a greedy heuristic for the multiple node service provision problem. This greedy heuristic parallels the greedy heuristic for multiple knapsack [12] with one important difference. In the service provision problem we may be able to meet demand of the critical items only partially. We give heuristics and bounds for the multiple node problem and show that the bounds are also tight for the single node problem.

Greedy uses the $v$-value ranking for the demands as introduced in the previous section, and orders the computing nodes arbitrarily. We denote the solution value of greedy by $Z^G$. The computing nodes are considered sequentially in arbitrary order. Computing node $i$ meets subservice demands in the prescribed order until it reaches a subservice, $g_i$, whose demand it cannot fully meet. Call $g_i$ the greedy critical item relative to $i$. Computing node $i$ meets as much demand of subservice $g_i$ as it can with its remaining capacity. If the remaining demand for subservice $g_j$ is positive, it is the first demand to be met by computing node $i$. The greedy critical items are found sequentially and are therefore dependent on the ordering of the computing nodes. They are not necessarily the same as those for the LP relaxation, (from (5) in Section 2.3), but $g_1 = k_1$ and $g_i \leq k_i$. $g_i$ is recursively defined by

$$g_i = \min \left\{ k \mid \sum_{j=1}^{k} (r_j + d_j) + \sum_{t=1}^{i-1} \min\{s^*_t, r_{g_t}\} > \sum_{t=1}^{i} s_t \right\},$$

where $s^*_t$ is the remaining capacity at node $i$, given by

$$s^*_t = \sum_{t=1}^{i} s_t - \sum_{j=1}^{g_i-1} (r_j + d_j) + \sum_{t=1}^{i-1} \min\{s^*_t, r_{g_t}\}.$$

This version of greedy can be implemented in $O(nm)$ time: using $O(n)$ time to find each critical item and reindex (m times), and $O(n + m)$ time to create the solution. We will show that $Z^G \geq Z^{OPT}(\bar{s} - r_{\text{max}})/\bar{s}$ with $\bar{s}$ mean node capacity and $r_{\text{max}}$ maximum installation requirement.

In order to obtain a bound that is independent of the problem data we device another greedy heuristic that we call this modification LP greedy and denote its solution value by $Z^{LG}$. LP greedy chooses the best solution of: that obtained by meeting all demands met by the LP relaxation solution defined in Section 2.3 except for the demands from the critical items $k_1, \ldots, k_m$, and the solutions obtained by meeting demand of each critical item $j = k_1, \ldots, k_m$ independently. LP greedy can be implemented such as to achieve a running time of $O(n + m)$ (see Section 2.3). We will show that

$$Z^G \geq \frac{1}{m+1} Z^{OPT}.$$
3.1 Worst case bounds

Proposition 2 \( Z^G \geq Z^{OPT}(\bar{s} - r_{max})/\bar{s} \) where \( \bar{s} = \frac{1}{m} \sum_{i=1}^{m} s_i \) and \( r_{max} = \max_{1 \leq j \leq n} r_j \).

Proof: Let \((z, x)\) be the greedy solution. For each node, \(i\), we have

\[
s_i = \sum_{j=g_{i-1}}^{g_i - 1} (x_{ij} + r_j) + x_{i,gi} + \min\{s_i^*, r_{gi}\},
\]

\[
\sum_{i=1}^{m} x_{ij} = d_j \quad \text{for } 1 \leq j < g_m \quad \text{and} \quad \sum_{i=1}^{m} x_{i,g_m} = d^* \leq d_{g_m}.
\]

This means

\[
\sum_{i=1}^{m} s_i = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} + \sum_{j=1}^{g_{i-1}} r_j + \sum_{i=1}^{m} \min\{s_i^*, r_{gi}\}
\]

\[
= \sum_{j=1}^{g_m - 1} (r_j + d_j) + \sum_{i=1}^{m} \min\{s_i^*, r_{gi}\} + d^*
\]

Now,

\[
Z^G = \sum_{j=1}^{g_m - 1} q_j d_j + q_{g_m} d^* = \sum_{j=1}^{g_m - 1} v_j (r_j + d_j) + q_{g_m} d^*
\]

and

\[
Z^{LP} = k_m - 1 \sum_{j=1}^{q_m} q_j d_j + q_{g_m} s^* = \sum_{j=1}^{g_m - 1} v_j (r_j + d_j) + q_{g_m} d^* + \sum_{j=g_m+1}^{k_m-1} v_j (r_j + d_j) + v_{k_m} s^*,
\]

where \( s^* \) comes from (4) with \( s \) replaced by \( \sum_{i=1}^{m} s_i \). Thus, \( s^* = \sum_{i=1}^{m} s_i - \sum_{j=1}^{g_m-1} (r_j + d_j) \).

Hence,

\[
\frac{Z^{OPT}}{Z^G} \leq \frac{Z^{LP}}{Z^G}
\]

\[
=\frac{v_{g_m} (d_{g_m} - d^*) (1 + \frac{r_{g_m} d_{g_m}}{d_{g_m}}) + v_{g_m} \sum_{j=g_m+1}^{k_m-1} (r_j + d_j) + v_{g_m} s^*}{1 + \frac{v_{g_m} \sum_{j=1}^{g_m-1} (r_j + d_j) + v_{g_m} d^* (1 + \frac{r_{g_m} d_{g_m}}{d_{g_m}})}{\sum_{i=1}^{m} s_i}}
\]

\[
=\frac{\sum_{i=1}^{m} (s_i - \min\{s_i^*, r_{gi}\}) + \frac{s_m d^*}{d_{g_m}}}{\frac{1}{m} \sum_{i=1}^{m} \max\{0, s_i - r_{gi}\}}
\]

The result follows.

To prove tightness of this bound consider the following set of instances with \( n \geq m + 2, m \geq 2, \)

\( s_1 = \cdots = s_{m-1} = 2u + r (u \geq 1), s_m = 2r + 2u, r_1 = u, r_2 = \cdots = r_n = r, d_1 = u, d_2 = 2(m-1)u, \)

\( d_3 = \cdots = d_{n-1} = 2u, d_n = 2u + r, q_1 = 4u(2u + r)(m - 1), q_2 = (2u + r)(2(m-1)u + r), q_3 = \)
\[ \cdots = q_{n-1} = (2u+r)^2(m-1) \text{ and } q_n = 2u(2u+2r)(m-1). \]
\[ Z^G = 2u(2u+r)(m-1)(2mu+r) \text{ and } Z^{OPT} = 2u(2u+r)(m-1)(2mu+r+mr) \text{ and } Z^{OPT}/Z^G = (2mu+r+mr)/(2mu+r) = \frac{s}{s-r_{max}}. \]

For the single node case consider the following set of examples with \( n = 3, s = r + u (u \geq 2), \)
\( r_1 = 1, r_2 = r_3 = r \geq 1, d_1 = u - 1, d_2 = 1, d_3 = u, q_1 = u^2, q_2 = (u-1)u(r+1) \) and
\( q_3 = (u+r)(u-1). \) From this \( Z^G = (u-1)u^2, Z^{OPT} = (u+r)(u-1)u \) and \( Z^{OPT}/Z^G = (u+r)/u. \)

For LP greedy we have the following bound.

**Proposition 3** \( Z^{LG} \geq \frac{1}{m+1} Z^{OPT}. \)

**Proof:**
From Section 2.3 we have
\[ Z^{LP} \leq Z^T + \sum_{i=1}^{m} q_i d_i, \]
where \( Z^T \) is the part of the objective value \( Z^{LP} \) coming from demands which are fully met by a single computing node. We therefore have \( Z^T \leq Z^{OPT}. \) For LP greedy we get
\[ Z^{LG} \leq Z^{LP} \leq Z^{OPT} + \sum_{i=1}^{m} Z^{OPT} = (m+1)Z^{OPT}, \]
since \( Z^{OPT} \geq q_j d_j, j = 1, \ldots, n \) by assumption A1.

To show that the \( m + 1 \) bound is tight consider the following example with \( n = 2m + 1 \)
\( (m \geq 2), s_1 = 2k (k > m), s_2 = \cdots = s_m = k, r_1 = k - 1, r_2 = k, r_3 = 1 \) and \( r_{j+1} = k - 1 \)
for \( j = 3, 5, 7, \ldots, n - 2, r_n = k - 1, d_1 = \cdots = d_n = 1, \) and \( q_j = r_j + d_j \) for \( j = 1, \ldots, n. \)
From this \( Z^{LG} \leq \max \{k(k + 2m - 2, k + 1)\} = k(k + 2m - 2), Z^{OPT} = km + k \) and \( Z^{OPT}/Z^{LG} = k(m+1)/(k+2m-2) \) is arbitrarily close to \( m + 1 \) for sufficiently large \( k. \)

For the single node case consider the following set of examples with \( n = 3, s = 2k (k \geq 1), \)
\( r_1 = r_2 = r_3 = k, d_1 = d_2 = 1, d_3 = k, q_1 = q_2 = (k + 1) \) and \( q_3 = 2. \) From this \( Z^{LG} = (k+1), \)
\( Z^{OPT} = 2k \) and \( Z^{OPT}/Z^{LG} = 2k/(k+1). \)

If we make the additional assumption that \( r_j + d_j \leq \min_i s_i \) for all demands \( j \) we may improve LP greedy by considering the solution where all of the critical demands are met by different supply nodes. In this case, \( Z^{OPT} \geq \sum_{i=1}^{m} q_i d_i, \) and \( Z^{LG}/Z^{OPT} \) is arbitrarily close to \( 1/2 \) for sufficiently large \( k. \)

### 3.2 Updating greedy

Consider greedy for the single node problem. When we come to meet demand for subservice \( k, \)
we use its \( v \)-value as if all of its demand could be met. One possible modification is to update the \( v \)-value of subservices as the heuristic progresses based on the amount of demand that could be met at the current iteration. We call this heuristic *updating greedy* and denote its objective value by \( Z^{UG}. \) Perhaps surprisingly, updated greedy leads to worse worst-case bounds than greedy.

The single node tightness example for greedy in Section 3.1 gives the same ratio, \( s/(s-r_{max}), \)
for updating greedy. But now, consider the following sequence of examples (they are rational to make them clearer) with \( n, s > 1 \in \mathbb{Q}, \)
\( r_1 = \cdots = r_{n-1} = \frac{r-1}{2n}, r_n = 1, d_1 = \cdots = d_{n-1} = \frac{r-1}{2n}, \)
\( d_n = s - 1, \)
\[ q_j = \frac{s-1}{s} - \frac{j-1}{n-j+1} + \frac{1}{n^2} \text{ for } j = 1, \ldots, n - 1, \]
and \( q_n = 1. \) Updated greedy meets demands 1 to \( n - 1 \) with
\[ Z^{UG} = \sum_{j=1}^{n-1} \left( \frac{s-1}{1 - \frac{j-1}{n-j+1} + \frac{1}{n^2}} \right) < (s-1) - \log s + \frac{1}{n}. \]
using the inequality $x > \log(1 + x)$ for $x > 0$ and log identities. Now $Z^{OPT} = 1 - s$, for this instance $Z^{UG}/Z^{OPT} < 1 - \log s/(s - 1) + 1/(n(1 - s))$)

$$\frac{Z^{OPT}}{Z^{UG}} > \frac{s - 1}{s - 1 - \log s - \frac{1}{n}} = \frac{(s - 1)^2}{s(s - 1 - \log s - \frac{1}{n})} = \frac{s}{s - 1}$$

which can be arbitrarily close to $2s/(s - 1) = 2s/(s - r_{\text{max}})$ for $s$ sufficiently close to 1 and sufficiently large $n$.

4 A fully polynomial-time approximation scheme

As a preliminary to a fully polynomial-time approximation scheme (fptas) for the service provision problem we present a dynamic programming formulation and determine the time complexity to solve the recursion. The first recursion presented in Subsection 4.1 is quite straightforward. As a basis for the fptas in subsection 4.3 we need to improve this recursion in Subsection 4.2. The results of this section is based on the work of Ibarra and Kim [9] and Lawler [11] for the knapsack problem.

4.1 A simple recursion

The single node service provision problem can be solved by dynamic programming, given that we made the assumption that all problem parameters are integers. Let the stages of the DP formulation correspond to the subservices numbered 1, ..., $n$. As a state of the recursion we take $\pi$, which is the profit to be achieved. We then define $f_j(\pi)$ as the minimum node capacity required to obtain a profit of at least $\pi$ using the subservices 1, ..., $j$, $j = 1, ..., n$. Starting from

$$f_0(\pi) = 0 \text{ for } \pi \leq 0, \quad f_j(\pi) = \infty, \text{ for } \pi > 0,$$

this leads to the following recursion for $j = 1, ..., n$

$$f_j(\pi) = \min_{0 \leq x_j \leq d_j} \{f_{j-1}(\pi - q_j x_j) + r_j + x_j, f_{j-1}(\pi)\}.$$ 

We see that the evaluation of each $f_j(\pi)$ requires $d_j + 1$ comparisons. For each stage $j$ the recursion has to be solved up to the point where we find $\pi$ for which $f_j(\pi) \leq S$ and $f_j(\pi + 1) \geq S + 1$. The optimal solution $\pi^{OPT}$ is found in the $n$-th stage. Going through the recursion by evaluating $f_j(\pi)$ for all $j = 1, ..., n$ before evaluating any $f_j(\pi + 1)$ the time complexity of the DP-method will be $O(n \pi^{OPT} \sum_{j=1}^{n} (d_j + 1))$. Storing all intermediate information requires space complexity $O(n \pi^{OPT})$.

The problem with the above recursion is that its complexity is an exponential function of two binary encoded problem parameters, $\log \pi^{OPT}$ and $\log(\max_j d_j)$. This makes it unfit as a basis for a fully polynomial-time approximation scheme. In the following subsection we will present an improvement over the recursion that allows a time complexity of $O(n \pi^{OPT})$.

4.2 A better recursion

An improvement over the recursion in the previous section can be obtained via the following lemma which holds for any capacity $s$.

**Lemma 1** Given any decision on the installment of subservices (fixing values of $z_j$, $j = 1, ..., n$ in (2)), optimization over the $x_j$-variables has a solution in which at most one $x_j$ has a value different from 0 or $d_j$.

**Proof:** Fixing the variables $z_j$ in (2) leads to an LP-problem with one constraint and bounded non-negative variables. Hence any basic optimal solution will have one basic variable. Each of
the other (non-basic) variables $x_j$ will have a value equal to either its lower bound 0 or its upper bound $d_j$.

Let $L = (z, x)$ be an optimal solution to the single node service provision problem with at most one subservice demand met fractionally.

**Lemma 2** When $j_f$ is the only fractional subservice in the optimal solution $L$, then for all $j$ for which $q_j < q_{j_f}$ we know that $x_j = 0$.

**Proof:** Assume $\exists i \in \{j | q_j < q_{j_f}\}$ for which $x_i > 0$. Then examine the solution $L' = (z', x')$ defined by: $x'_j = z_j, j = 1, \ldots, n$ and $x'_j = x_j, j \notin \{l, j_f\}, x'_j = x_j + \epsilon, x'_i = x_i - \epsilon$ where $0 < \epsilon < x_i$. Then the objective value of $L$, $\pi_L$, is smaller than the objective value of $L'$, $\pi_{L'}$: $\pi_L - \pi_{L'} = \epsilon(q_{j_f} - q_i) > 0$, contradicting the optimality of $L$.

We define two sets of solutions of (2).

**Definition 1** Let $Q$ be the solution set of all feasible solutions to (2) with at most one partially met demand.

According to Lemma 1 it is enough to search $Q$ to find an optimal solution.

**Definition 2** For any $L = (z, x) \in Q$, let $Q(L, l)$ be the set of solutions in $Q$ which have the same first $l - 1$ elements as $L$,

$$Q(L, l) = \{(z', x') \in Q | (z'_1, \ldots, z'_{l-1}) = (z_1, \ldots, z_{l-1})\}.$$  

Consider the partial solution of any $L \in Q$ for the subset of services ranging from 1, \ldots, $l-1$. It is enough to search $Q(L, l)$ to check if this partial solution can be included in an optimal solution.

**Lemma 3** Assume that the subservices are sorted so that $q_1 \geq q_2 \geq \ldots \geq q_n$. For any solution $L \in Q$ and any $l \in \{1, \ldots, n\}$ the following holds: if an optimal solution to (2) is contained in $Q(L, l)$, there exists an optimal solution in $Q(L, l)$ where

$$x_l \in \{0, \min(s - r_l - \sum_{j=1}^{l-1} (r_j z_j + x_j), d_l)\}$$

**Proof:** Define $s^* = s - r_l - \sum_{j=1}^{l-1} (r_j z_j + x_j)$. Let us first study the case that $s^* \geq d_l$. Suppose the lemma is not true, i.e., any optimal solution in $Q(L, l)$ has $0 < x_l < d_l$. Then from Lemma 1 we know that there must exist a solution $L^1 = (z^1, x^1)$ in which $l$ is the only fractionally met demand.

Since the capacity constraints are binding if a partially met demand exists in an optimal solution we have

$$x_l^1 = s^* - \sum_{j \in I^l} (r_j + d_j) \quad (7)$$

with $I^l = \{j \mid l + 1 \leq j \leq n, x_j^1 = d_j\} \neq \emptyset$. Obviously, if such a solution $L^1$ exists, there must exist such a solution in which $I^l$ has minimum cardinality. Let us assume $L^1$ is that solution. By Lemma 2 we know that $q_j = q_l$ for all $j \in I^l$.

Let $k$ be the largest $j \in I^l$. Then define a new solution $L^2 = (z^2, x^2)$ by $z_j^2 = z_j, \forall j \neq l, k$, and $x_k^2 = \min(x_k^1 + d_k, d_l)$, $x_j^2 = x_j^1, \forall j \neq l, k$, and $x_k^2 = \max(0, d_l - (d_l - x_k^1))$. Obviously, $L^2$ has the same objective value as $L^1$ since we only redistributed demand met between subservices $l$ and $k$.

In case the residual demand of subservice $k$ has become 0, we have found a solution with a set $I^l$ of smaller cardinality than that of $L^1$, which contradicts the supposed minimality. Otherwise, $x_k^2 = d_l$ and we have a solution that satisfies the hypothesis of the lemma in contradiction to what we supposed.

An analogous argument shows the hypothesis for the case $s^* < d_l$.

It is clear that utilizing Lemma 3 the single node service provision problem can be solved by any dynamic programming approach for knapsack problems which
Dynamic Programming

procedure DynamicProgramming(s, d, q, r, CAP, L)
begin
\[ \pi_{\max} = Z_{LP} \]
for \( i := 1 \) to \( \pi_{\max} \) step 1 do
\[ \text{CAP}(i) := \text{''undefined''} \]
end do
CAP(0) = 0
for \( j := 1 \) to \( n \) step 1 do
for \( i := \pi_{\max} \) downto 0 step -1 do
\# different from knapsack - start
\[ w_j = \min(d_j + r_j, s - \text{CAP}(i)) \]
\[ p_j = q_j \cdot \max(0, w_j - r_j) \]
\# different from knapsack - stop
if \( \text{CAP}(i) \neq \text{undefined} \) and \( p_j > 0 \) and \( \text{CAP}(i) + w_j \leq s \) then
if \( \text{CAP}(i + p_j) = \text{undefined} \) or
\( \text{CAP}(i) + w_j < \text{CAP}(i + p_j) \) then
\[ \text{CAP}(i + p_j) := \text{CAP}(i) + w_j \]
\[ L(i + p_j) := L(i) \cup \{ j \} \]
end if
end if
end do
end do
end

Figure 3: Knapsack DP modified to solve single node service provision problems.

- allows sorting in the order described in Lemma 3
- allows that (item size, price) is reduced from \((r_j + d_j, q_j d_j)\) to \((\bar{s}, q_j (\bar{s} - r_j))\) in all states with \( r_j < \bar{s} < d_j + r_j \), where \( \bar{s} \) is the free capacity of the state.

These criteria are satisfied by, among others, the approaches of Ibarra and Kim [9] and Lawler [11] that leads to fully polynomial approximations for knapsack. To demonstrate this a simple dynamic program is given in Figure 3, slightly modified where indicated from a typical knapsack algorithm, see for example [12].

The algorithm takes as input \( r, q, d, s \). The vectors are assumed sorted in order of non-increasing \( q_j \). It outputs two lists \( \text{CAP} \) and \( L \). \( \text{CAP}(i) \) is the minimum capacity needed to reach a profit of \( i \). \( L(i) \) is the solution vector used to reach profit \( i \). The number of states is at most \( 2^{\pi^{OPT}} \) where \( \pi^{OPT} \) is the optimal solution, and in each state at most \( n + 1 \) words of information is saved. So the overall space complexity is \( O(n \pi_0) \) and the computational complexity is \( O(n \pi_0) \). In addition we need \( O(n \log n) \) to order the subservices.

4.3 A fully polynomial-time approximation scheme

The fptas that we present is inspired by Ibarra and Kim [9] and Lawler [11]. First we have to reformulate the problem: Each variable \( x_j \) is scaled by its demand \( d_j \) so as to obtain a variable \( y_j = x_j/d_j \), that has values in the set \( \{0, 1/d_j, \ldots, (d_j - 1)/d_j, 1\} \), \( j = 1, \ldots, n \). In order to retain
the same optimal value we have to multiply each of the profit coefficients \( q_j \) by the corresponding demand \( d_j \): \( q_j' = q_j d_j \).

Starting from this formulation we scale the profit coefficients with a factor \( k > 1 \): \( q_j = \left\lfloor \frac{q_j}{k} \right\rfloor \), \( j = 1, \ldots, n \), and we solve the recursion from the previous section on the scaled problem. After having found the optimal solution we multiply the value obtained by \( k \) as an approximation \( \tilde{\pi} \) to the optimal value \( \pi^{OPT} \) of the original problem. Let \( y^{OPT}_j \), and \( \tilde{y}_j \), \( j = 1, \ldots, n \) be the values for the \( y \)-variables in an optimal solution for the original problem and the scaled problem, respectively.

From Lemma 1 we know that there exist optimal solutions such that there is at most one subservice \( j \) whose demand is not met completely or not at all. Let \( 0 < y^{OPT}_j < d_j \) and \( 0 < y_{h} < d_h \).

\[
\tilde{\pi} = k \left( \sum_{j=1}^{n} q_j' \tilde{y}_j \right) \geq k \left( \sum_{j=1}^{n} q_j^{OPT} \right) \geq \sum_{j=1}^{n} (q_j' - k) y^{OPT}_j \geq \pi^{OPT} - k \sum_{j=1}^{n} y^{OPT}_j.
\]

Take \( k = \epsilon \pi^G / n \). From the above it follows Therefore,

\[
\frac{\pi^{OPT} - \tilde{\pi}}{\pi^{OPT}} \leq \frac{\epsilon \pi^G \sum_{j=1}^{n} y^{OPT}_j}{n \pi^{OPT}} \leq \frac{\epsilon}{n} \pi^G \leq \epsilon.
\]

The time to solve the scaled recursion is \( O(n \pi^{OPT} / k) \). Observe that

\[
\pi^{OPT} / k = \frac{n \pi^{OPT}}{\epsilon \pi^G} \leq \frac{2n}{\epsilon},
\]

so that the time complexity is \( O(n^2 / \epsilon) \).

The time complexity can be changed to \( O(n^2 / \epsilon^2) \) by a technique similar to the one proposed in [11].

### 4.4 No fully polynomial time approximation scheme

We emphasize that the multiple node service provision problem is strongly NP-hard, so that a fptas for it is highly unlikely to exist. In what follows we will prove that even if the number of nodes if fixed an fptas can only exist if \( P=NP \). We show that this is already so for the problem with two computing nodes by a reduction from the bipartite partition problem.

In the bipartition problem \( n \) positive integers \( a_1, \ldots, a_n \) are given that add up to \( 2A \). The question is if there exists a subset of these \( n \) numbers that add up to \( A \). This problem is NP-hard [6].

Given an instance of the bipartition problem we define the following service provision problem with 2 nodes, that we state immediately in its integer programming formulation.

\[
\text{max} \quad \sum_{j=1}^{n} a_{1j} x_{1j}
\]

\[
\text{s.t.} \quad \sum_{j=1}^{n} (a_{1j} - 1) z_{1j} + \sum_{j=1}^{n} x_{1j} \leq A \\
\sum_{j=1}^{n} (a_{2j} - 1) z_{2j} + \sum_{j=1}^{n} x_{2j} \leq A \\
x_{1j} + x_{2j} \leq 1, \quad j = 1, \ldots, n \\
z_{ij} - x_{ij} \geq 0, \quad i = 1, 2, \quad j = 1, \ldots, n \\
z_{ij} \in \{0, 1\}, \quad x_{ij} \geq 0, \quad i = 1, 2, \quad j = 1, \ldots, n.
\]

If there is a bipartition then we can satisfy all demand and hence have an optimal solution of the service provision problem of size \( n \). Now suppose there exists a fptas for the service provision
problem with 2 nodes and we allow an error of $\varepsilon = 1/(n+1)$. Then if the answer to the bipartition problem is "yes" the fptas will output a solution with value $n$. On the other hand if there is no bipartition the fptas will arrive at a solution value of no more than $n - 1$. This implies that the bipartition problem could then be decided in polynomial time since the running time of the fptas is polynomial in $n$ and $1/\varepsilon$, and hence with a choice of $1/(n+1)$ for $\varepsilon$, polynomial in $n$. This would prove that P=NP.

5 Lagrangian relaxation

For the investigation of Lagrangian relaxations we take $M_{ij} = d_j$ in (1). Using a tighter value of $M_{ij}$ will generally provide better relaxations. In general we formulate the Lagrangian relaxation of the multiple node formulation, (1), referring to the single node formulation when there is a difference in the relaxations.

5.1 Capacity constraint

The Lagrangian relaxation of (1) with respect to the capacity constraints can be formulated as

$$L^R(\lambda) = \max \sum_{j=1}^{n} \sum_{i=1}^{m} (q_j - \lambda_i)x_{ij} - \sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i r_{ij}z_{ij} + \sum_{i=1}^{m} \lambda_i s_i$$

s.t. $\sum_{i=1}^{m} z_{ij} \leq d_j$ \quad $j = 1, \ldots, n$,

$d_jz_{ij} - x_{ij} \geq 0$ \quad $i = 1, \ldots, m$, \quad $j = 1, \ldots, n$,

$z_{ij} \in \{0,1\}, x_{ij} \geq 0$ \quad $i = 1, \ldots, m$, \quad $j = 1, \ldots, n$

**Proposition 4** All extreme solutions of the LP relaxation of (8) are integral.

**Proof:** The problem decomposes into $n$ subproblems (one for each subservice). Consider subproblem $j$, $j = 1, \ldots, n$. Now rewrite the $x$-variables as $x_{ij} = d_j x_{ij}/d_j$. Then for each $j$ the corresponding demand constraint and each of the installation constraints can be divided by $d_j$. After doing so the matrix of the left hand side coefficients of the resulting constraints in the variables $z_{ij}$ and $x_{ij}/d_j$ is totally unimodular. Therefore the extreme feasible points are all integer (and hence this also w.r.t. the problem in the original $x_{ij}$-variables). □

A known theorem states that in such a case as above, in which the Lagrangian relaxation of an integer programming problem has the property that its polytope is integral then the Lagrangian dual $Z^{LD} = \min_{\lambda \geq 0} Z^{LR}(\lambda)$ is equal to the optimal value of the LP relaxation of the problem $Z^{LP}$ (see Nemhauser and Wolsey [14, Section II.3, Corollary 6.6]).

5.2 Demand constraint

The Lagrangian dual of (1) with respect to the demand constraint can be formulated as

$$Z^{LD} = \min_{\lambda \geq 0} Z^{LR}(\lambda)$$

where

$$Z^{LR}(\lambda) = \max \sum_{i=1}^{m} \sum_{j=1}^{n} (q_j - \lambda_j)x_{ij} + \sum_{j=1}^{n} \lambda_j d_j$$

s.t. $\sum_{j=1}^{n} r_{ij}z_{ij} + \sum_{j=1}^{n} x_{ij} \leq s_i$ \quad $\forall i = 1, \ldots, m$,

$d_jz_{ij} - x_{ij} \geq 0$ \quad $\forall j = 1, \ldots, n$, \quad $i = 1, \ldots, m$,

$z \in \{0,1\}^{n \times m}$, \quad $x \geq 0$.

This decomposes into solving $m$ single node problems.
Note that this relaxation is only appropriate when \( m \geq 2 \) since when \( m = 1 \) the demand constraints appear as bounds on \( z \) (which are redundant due to the installation constraints).

For the case when \( m \geq 2 \) we illustrate that we can have \( Z^{LD} < Z^{LP} \), i.e., the Lagrangian relaxation with respect to the demand constraints may lead to better bounds than the LP relaxation. It is important to notice that while the original problem was strongly NP-hard, the Lagrangian relaxation here has a fully polynomial approximation scheme (see Section 4.3).

The example has \( n = m = 2 \) and is defined by the following data:

\[
s = [65, 90], \quad d = [84, 47], \quad r = [22, 20], \quad g = [10, 7].
\]

The LP relaxation is easily solved (see Section 2.3). It has optimal value \( Z^{LP} \approx 1080 \).

Now consider \( Z^{LR}(\lambda) \) with \( \lambda = [22/84, 0] \). Complete enumeration of \( z \) and solving the remaining problems for \( x \) yields Table 1 from which we read that \( Z^{LD} \leq Z^{LR}(\lambda) = 1037^{24/28} < Z^{LP} \).

<table>
<thead>
<tr>
<th>( z )</th>
<th>( s - rz )</th>
<th>( x^* )</th>
<th>Objective value</th>
</tr>
</thead>
<tbody>
<tr>
<td>([0, 0, 0, 0])</td>
<td>([65, 90])</td>
<td>([0, 0, 0, 0])</td>
<td>225</td>
</tr>
<tr>
<td>([0, 0, 0, 1])</td>
<td>([65, 70])</td>
<td>([0, 0, 0, 47])</td>
<td>554</td>
</tr>
<tr>
<td>([0, 0, 1, 0])</td>
<td>([65, 68])</td>
<td>([0, 0, 68, 0])</td>
<td>722^{24/28}</td>
</tr>
<tr>
<td>([0, 0, 1, 1])</td>
<td>([65, 48])</td>
<td>([0, 0, 48, 0])</td>
<td>576^{12/28}</td>
</tr>
<tr>
<td>([0, 1, 0, 0])</td>
<td>([45, 90])</td>
<td>([0, 45, 0, 0])</td>
<td>540</td>
</tr>
<tr>
<td>([0, 1, 0, 1])</td>
<td>([45, 70])</td>
<td>([0, 45, 0, 47])</td>
<td>869</td>
</tr>
<tr>
<td>([0, 1, 1, 0])</td>
<td>([45, 68])</td>
<td>([0, 45, 68, 0])</td>
<td>1037^{24/28}</td>
</tr>
<tr>
<td>([0, 1, 1, 1])</td>
<td>([45, 48])</td>
<td>([0, 45, 48, 0])</td>
<td>891^{12/28}</td>
</tr>
<tr>
<td>([1, 0, 0, 0])</td>
<td>([43, 90])</td>
<td>([43, 0, 0, 0])</td>
<td>539^{23/28}</td>
</tr>
<tr>
<td>([1, 0, 0, 1])</td>
<td>([43, 70])</td>
<td>([43, 0, 0, 47])</td>
<td>868^{23/28}</td>
</tr>
<tr>
<td>([1, 0, 1, 0])</td>
<td>([43, 68])</td>
<td>([43, 0, 68, 0])</td>
<td>1037^{12/28}</td>
</tr>
<tr>
<td>([1, 0, 1, 1])</td>
<td>([43, 48])</td>
<td>([43, 0, 48, 0])</td>
<td>891^{7/28}</td>
</tr>
<tr>
<td>([1, 1, 0, 0])</td>
<td>([23, 90])</td>
<td>([23, 0, 0, 0])</td>
<td>393^{11/28}</td>
</tr>
<tr>
<td>([1, 1, 0, 1])</td>
<td>([23, 70])</td>
<td>([23, 0, 0, 47])</td>
<td>722^{11/28}</td>
</tr>
<tr>
<td>([1, 1, 1, 0])</td>
<td>([23, 68])</td>
<td>([23, 0, 68, 0])</td>
<td>891^{7/28}</td>
</tr>
<tr>
<td>([1, 1, 1, 1])</td>
<td>([23, 48])</td>
<td>([23, 0, 48, 0])</td>
<td>744^{23/28}</td>
</tr>
</tbody>
</table>

### 5.3 Installation constraint

The Lagrangian dual of (1) with respect to the installation constraints can be formulated as

\[
Z^{LD} = \min_{\lambda \geq 0} Z^{LR}(\lambda)
\]
where

\[
Z^{LR}(\lambda) = \max \sum_{i=1}^{m} \sum_{j=1}^{n} (q_{ij} - \lambda_{ij})x_{ij} + \sum_{i=1}^{m} \sum_{j=1}^{n} \lambda_{ij}d_{ij}z_{ij}
\]

s.t. \[
\sum_{j=1}^{n} r_{ij}z_{ij} + \sum_{j=1}^{n} x_{ij} \leq s_i \quad i = 1, \ldots, m
\]

\[
\sum_{i=1}^{m} x_{ij} \leq d_j \quad j = 1, \ldots, n
\]

\[
z_{ij} \in \{0,1\}, \quad x_{ij} \geq 0 \quad i = 1, \ldots, m, \quad j = 1, \ldots, n
\]

(10)

For the multiple node case the same example from the previous subsection shows that it is possible that \(Z^{LD} < Z^{LP}\). However, we notice that the Lagrangian relaxation here is another strongly NP-hard problem (the same polynomial transformation as used in Proposition 1 suffices). Recall that for this example \(Z^{LP} = 1080\). For the single node problem the situation is different. We will show that here \(Z^{LD} = Z^{LP}\). Complete enumeration of \(z\) and solving the remaining problems for \(x\) yields Table 2 from which we read that \(Z^{LD} \leq Z^{LR}(\lambda) = 1078 < Z^{LP}\).

<table>
<thead>
<tr>
<th>(z)</th>
<th>(s - rz)</th>
<th>(x^*)</th>
<th>Objective value</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0,0,0,0]</td>
<td>[65,90]</td>
<td>[0,47,84,0]</td>
<td>977</td>
</tr>
<tr>
<td>[0,0,0,1]</td>
<td>[65,70]</td>
<td>[65, 0,19,47]</td>
<td>1068</td>
</tr>
<tr>
<td>[0,0,1,0]</td>
<td>[65,68]</td>
<td>[16,47,68,0]</td>
<td>1078</td>
</tr>
<tr>
<td>[0,0,1,1]</td>
<td>[65,48]</td>
<td>[65, 0,19,29]</td>
<td>1077(\frac{10}{47})</td>
</tr>
<tr>
<td>[0,1,0,0]</td>
<td>[45,90]</td>
<td>[0,45,84,2]</td>
<td>1068</td>
</tr>
<tr>
<td>[0,1,0,1]</td>
<td>[45,70]</td>
<td>[14,31,70,0]</td>
<td>1077(\frac{40}{47})</td>
</tr>
<tr>
<td>[0,1,1,0]</td>
<td>[45,68]</td>
<td>[16,29,68,0]</td>
<td>1077(\frac{40}{47})</td>
</tr>
<tr>
<td>[0,1,1,1]</td>
<td>[45,48]</td>
<td>[36, 9,48,0]</td>
<td>1067(\frac{7}{47})</td>
</tr>
<tr>
<td>[1,0,0,0]</td>
<td>[43,90]</td>
<td>[0,43,84,4]</td>
<td>1078</td>
</tr>
<tr>
<td>[1,0,0,1]</td>
<td>[43,70]</td>
<td>[43, 0,41,29]</td>
<td>1077(\frac{10}{47})</td>
</tr>
<tr>
<td>[1,0,1,0]</td>
<td>[43,68]</td>
<td>[16,27,68,0]</td>
<td>1077(\frac{34}{47})</td>
</tr>
<tr>
<td>[1,0,1,1]</td>
<td>[43,48]</td>
<td>[36, 7,48,0]</td>
<td>1067(\frac{5}{47})</td>
</tr>
<tr>
<td>[1,1,0,0]</td>
<td>[23,90]</td>
<td>[0,23,84,6]</td>
<td>1077(\frac{49}{47})</td>
</tr>
<tr>
<td>[1,1,0,1]</td>
<td>[23,70]</td>
<td>[14, 9,70,0]</td>
<td>1067(\frac{27}{47})</td>
</tr>
<tr>
<td>[1,1,1,0]</td>
<td>[23,68]</td>
<td>[16, 7,68,0]</td>
<td>1067(\frac{21}{47})</td>
</tr>
<tr>
<td>[1,1,1,1]</td>
<td>[23,48]</td>
<td>[23, 0,48,0]</td>
<td>1008(\frac{13}{84})</td>
</tr>
</tbody>
</table>

For the single node problem the situation is different. We will show that here \(Z^{LD} = Z^{LP}\).
Before proving this we need some preliminary results. Define the augmented single node service provision problem by introducing an additional subservice, \( j = n + 1 \), with \( q_{n+1} = 0 \), \( r_{n+1} = 0 \) and \( d_{n+1} = s \) and the node capacity constraint required to hold with equality. Subservice \( n + 1 \) is an explicit slack variable for the node capacity constraint.

**Lemma 4** Any feasible solution to the augmented problem, \((x^a, z^a)\), defines a feasible solution to the original by considering only services \( j = 1, \ldots, n \). This is also true for the LP relaxation, Lagrangian relaxation and Lagrangian dual. Additionally, for the mixed integer problem, the LP relaxation and the Lagrangian dual these solutions have the same objective value.

**Proof:** The result is immediate for the single node service provision problem and its LP relaxation. The feasibility result is also true for the Lagrangian relaxation. Now, for any \( \lambda \in \mathbb{R}^{n+1}_+ \) putting \( \lambda_j^a = \lambda_j \) for \( j = 1, \ldots, n \) and \( \lambda_{n+1}^a = 0 \) we obtain \( Z^{LR}(\lambda') \leq Z^{LR}(\lambda) \) for the augmented problem and the result follows for the Lagrangian dual. \( \square \)

In light of the above lemma, from now on we consider only the augmented problem. Notice that for the Lagrangian dual we may restrict the search over \( \lambda \in \mathbb{R}^{n+1}_+ \) to those for which \( 0 \leq \lambda \leq q \) since if \( \lambda_j > q_j \) for \( j = 1, \ldots, n \) and \( \lambda_{n+1} = 0 \), \( x_j = 0 \) at every optimal solution. We can formulate the Lagrangian dual as

\[
Z^{LD} = \min_{0 \leq \lambda \leq q} Z^{LR}(\lambda)
\]

where

\[
Z^{LR}(\lambda) = \max \sum_{j=1}^{n+1} (q_j - \lambda_j) x_j + \sum_{j=1}^{n+1} \lambda_j d_j z_j
\]

s.t.

\[
\sum_{j=1}^{n+1} r_j z_j + \sum_{j=1}^{n+1} x_j = s
\]

\[
0 \leq x_j \leq d_j \quad j = 1, \ldots, n + 1
\]

\[
z_j \in \{0, 1\}, \quad j = 1, \ldots, n + 1.
\]

Notice that (11) and its extreme solutions have a very special structure. We need the following observations.

**Lemma 5** Every extreme solution, \((x, z)\) of (11) has a subservice, \( \ell \), (the critical item of the extreme solution) with the following properties

\[
x_j = d_j \iff q_j - \lambda_j > q_\ell - \lambda_\ell,
\]

\[
x_j = 0 \iff q_j - \lambda_j < q_\ell - \lambda_\ell.
\]

**Proof:** Fixing \( z \) to its optimal values, the results follow from Section 2.3. \( \square \)

In light of this result we define the sets

\[
W(x, z) = \{1 \leq j \leq n + 1 \mid q_j - \lambda_j > q_\ell - \lambda_\ell\}
\]

and

\[
Y(x, z) = \{1 \leq j \leq n + 1 \mid q_j - \lambda_j < q_\ell - \lambda_\ell\}.
\]

at extreme solution \((x, z)\) for which \( \ell \) is the critical item. Note that \( W(x, z) \) is well defined even when the critical item is not unique. The following two results show that we need only consider \( \lambda \) values for which \( W(x, z) = \emptyset \) and all \( j \in Y(x, z) \) have \( \lambda_j = 0 \) where \((x, z)\) is the optimal solution to the Lagrangian relaxation.

**Lemma 6** Given any \( \lambda \) for which \( 0 \leq \lambda \leq q \), there exists \( \lambda' \) such that \( Z^{LR}(\lambda') \leq Z^{LR}(\lambda) \) and \( |W(x', z')| = 0 \) where \((x', z')\) is an optimal solution to (11) using \( \lambda' \).
Proof: Set \( v = 1 \) and \( \lambda^v = \lambda \). Let \((x^v, z^v)\) be an extreme optimal solution to (11) using \( \lambda^v \) with critical item \( \ell^v \) (if there are many possible solutions choose one which minimizes \(|W(x^v, z^v)|\)). If \(|W(x^v, z^v)| = 0\) we are done. Choose \( k^v \) to be a minimizer of \((q_j - \ell^v_j)\) over \( W(x^v, z^v) \). We consider parametrically increasing \( \lambda_{k^v} \). Let \( \eta(y) \in \mathbb{R}^{u+1} \) be defined by \( \eta_j = \lambda^v_j \) for \( j \neq k^v \) and \( \eta_{k^v} = y \) with \( y \) ranging over the interval \([\lambda^v_{k^v}, \lambda^v_{k^v} - q_{k^v} + q_{k^v}]\). This interval may be covered by finitely many, \( u \), non-degenerate subintervals, \( \{[a_t, b_t]\}_{t=1}^u \), each associated with a single extreme solution, \((x(t), z(t))\) which is optimal for (11) using \( \eta(y) \) with \( y \in [a_t, b_t] \) [13, Theorem 7]. If none of these optimal solutions has \( x(t) < \lambda_{k^v} \) put \( y^* = \lambda_{k^v} - q_{k^v} \) otherwise put \( y^* \) equal to the smallest \( a_t \) for which \( x(t) < \lambda_{k^v} \). Let \((x^*, z^*) = (x(t), z(t))\) if \( y^* = a_t \) and \((x^*, z^*) = (x(u), z(u))\) otherwise. It follows from Lemma 5 that \(|W(x^*, z^*)| = 0\) we are done. Otherwise repeat the above with \( \lambda^{v+1} = y^* \) and \( v \) incremented by 1. Finiteness of termination is guaranteed by the since \(|W(x^v, z^v)|\) is finite and with each iteration we decrease the cardinality of \( W(x^v, z^v) \) by at least 1.

Lemma 7 Given any \( \lambda \) for which \( 0 \leq \lambda \leq q \), there exists \( \lambda' \) such that \( Z^{LR}(\lambda') \leq Z^{LR}(\lambda) \) and all \( j \in Y(x', z') \) have \( \lambda'_j = 0 \) where \((x', z')\) is an optimal solution to (11) using \( \lambda' \).

Proof: Similar to that of Lemma 6. □

Proposition 5 For the single node problem the Lagrangian dual of (1) with respect to the installation constraints has \( Z^{LD} = Z^{LP} \).

Proof of Proposition 5: We consider the Lagrangian relaxation of the augmented problem. By Lemma 6 and 7 we need only consider \( A \) for which \( \lambda_j = \max\{q_j - \gamma, 0\} \) for some \( \gamma \in \mathbb{R}_+ \) and for which there is an optimal solution to (11), \((x, z)\), for which all \( j \) with \( q_j < \gamma \) have \( x_j = 0 \). Define \( P_\gamma = j \mid 1 \leq j \leq n + 1, q_j \geq \gamma \), then \( Z^{LD} \) may be expressed as

\[
Z^{LD} = \min_{\gamma \leq \max q_j} f(\gamma)
\]

where

\[
f(\gamma) = \max \sum_{j \in P_\gamma} \gamma x_j + \sum_{j \in P_\gamma} (q_j - \gamma)d_j z_j
\]

s.t. \( \sum_{j \in P_\gamma} r_j z_j + \sum_{j \in P(\gamma)} x_j = s \)

\[
0 \leq x_j \leq d_j, \quad j = 1, \ldots, n + 1
\]

\[
z_j \in \{0, 1\}, \quad j = 1, \ldots, n + 1.
\]

Introducing a new variable \( y = \sum_{j \in P_\gamma} x_j \) we may write \( f(\gamma) \) equivalently as

\[
f(\gamma) = \max \gamma y + \sum_{j=1}^{n+1} \max\{q_j - \gamma, 0\} d_j z_j
\]

s.t. \( \sum_{j=1}^{n+1} r_j z_j + y = s \)

\[
y \geq 0
\]

\[
z_j \in \{0, 1\}, \quad j = 1, \ldots, n + 1.
\]
Notice that there are no upper bounds on $y$. Using the constraint we substitute $s - \sum_{j=1}^{n+1} r_j z_j$ for $y$ giving

$$f(\gamma) = \max \gamma s + \sum_{j=1}^{n+1} (\max(q_j - \gamma, 0)d_j - \gamma r_j)z_j$$

s.t. \[ \sum_{j=1}^{n+1} r_j z_j \leq s \]

\[ z_j \in \{0, 1\}, \quad j = 1, \ldots, n + 1. \]

where the new constraint comes from the non-negativity condition on $y$. If we relax the constraint, the optimal solution is

$$z^* = \begin{cases} 
1 & \text{if } \frac{q_j d_j}{r_j + d_j} > \gamma, \\
0 & \text{if } \frac{q_j d_j}{r_j + d_j} \leq \gamma.
\end{cases}$$

We see that the inner maximization is a piecewise linear function with breakpoints at $\gamma = \frac{q_j d_j}{r_j + d_j}$. The minimum over these breakpoints is at $\gamma^* = \frac{q_\ell d_\ell}{r_\ell + d_\ell}$, where $\ell$ is the critical item of the LP relaxation (see Section 2.3). With $\gamma = \gamma^*$ there is an optimal solution to the inside maximization for which the constraint holds and it is easy to see that $Z^LP = f(\gamma^*) = Z^LP$.

\[ \square \]

### 6 Concluding remarks

In this paper we have studied approximations of the optimal solution value of service provision problems on distributed telecommunication networks. Worst case performance ratios of approximation algorithms resemble those for the optimal solution of the (multiple) knapsack problem. For lower bounds we showed that some Lagrangian relaxations might give better lower bounds than the LP relaxation, although in one case it leads to another NP-hard problem.

In another paper, which is in progress, we study the problems from the perspective of polyhedral theory as a basis for a (branch-and-bound type of) exact method.

All the results in this paper concern deterministic problems, which are useful to study where it comes to the setting up of a configuration of the network to accommodate "normal demand" situations. Of course, provision of services is a highly dynamical process with inherent uncertainty about the times requests for services occur. Therefore, next to the deterministic version it is very natural to study a version of the problem in which this uncertainty is reflected, e.g., by specification of probability distributions on the demands for requests.

The telecommunication industry is especially interested in situations in which demand for one or a subset of the services peaks more or less unexpectedly. Such a situation might be modelled as a two-stage stochastic programming problem. In solving such models the approximation results derived here will play an important role for gathering estimates of second-stage costs or benefits. Work in this direction is in progress at this moment.

Another possible way to deal with the uncertainty is using the paradigm of on-line optimization. In this direction no attempts have been made. A considerable challenge is to devise for this problem a reasonable adversary model for providing the on-line input. Without any restrictions on the input that might be given it is easy to see that any algorithm might perform arbitrarily badly in comparison to an optimal off-line solution.

### Acknowledgements

The authors thank Gerhard Woeginger for a discussion that gave the insight for the proof of the non-existence of a fptas for the service provision problem with two nodes. Maarten van der Vlerk
is acknowledged for the discussions we had at the initialization of this research into approximation algorithms for the service provision problem.

References


