Optimal design of hierarchical networks with free main path extremes

Carlos Obreque a,b, Vladimir Marianov c,*, Miguel Ríos c

a Graduate Program, Department of Systems Engineering, Pontificia Universidad Católica de Chile, Chile
b Department of Industrial Engineering, Universidad del Bío-Bío, Concepción, Chile
c Department of Electrical Engineering, Pontificia Universidad Católica de Chile, Santiago, Chile

Received 28 December 2006; accepted 15 October 2007
Available online 29 January 2008

Abstract

We propose an optimal, two-stage procedure for the optimal design of minimum cost hierarchical spanning networks, consisting of a main path and secondary trees. The optimal location of the origin and destination nodes of the path is also found. We test our procedure and compare it with a known method.

Keywords: Hierarchical networks; Integer programming; Heuristics

1. Introduction

The hierarchical network design problem (HNDP) introduced by Current et al. [4] consists of finding the least cost two-level network that connects all the nodes in a given set. The higher level part of the network is a path composed of primary arcs, which visits some of the nodes of the network (primary nodes). The lower level portion of the network is composed of one or more trees whose arcs, termed secondary, are less expensive to build than the primary arcs. For all of the nodes that are not on the main path (secondary nodes), those trees provide a connection to one of the nodes on the path. Applications include telecommunications and transportation networks, as well as power line systems.

The HNDP is NP-hard, even if primary costs \( c_{ij} \) and secondary costs \( d_{ij} \) are proportional, i.e. if there is a constant \( \alpha > 0 \) such that \( c_{ij} = \alpha d_{ij} \). The HNDP has been studied and solved using different approaches [4,7,9,16,18]. It has been regarded as a particular case of more general problems [1–3,8,11,15,17]. It has been extended to include multiple main paths [19] and to consider fixed costs over the main path [5,6].

Current et al. [4] formulate the HNDP as an integer programming model, with a set of constraints that avoid the formation of tours, whose number increases exponentially with the number of nodes of the network. Pirkul et al. [16] propose a heuristic based on a Lagrangian relaxation of a new formulation of the problem. In Current [5] and Current and Pirkul [6], the nodes on the main path have a fixed cost representing the cost of locating transshipment facilities. Pirkul and Nagarajan [17] design a centralized computer network with concentrators in the nodes of the main path. Duin and Volgenant [7] introduce some methods for identifying optimal arcs and for eliminating some variables from the HNDP. Sancho [18] proposes a procedure based on dynamic programming that finds a sub-optimal solution of the HNDP. Sancho [19] extends this procedure to the case of multiple primary paths.

The so called two-level network design problem (TLNDP) is a generalization of the HNDP. In the TLNDP, the higher level part of the network is a tree. The TLNDP and its extension to multiple hierarchies has been studied extensively [1–3,8–11,13–15].

Some authors approach the HNDP by reducing its size before formulating the optimization models. The reduction consists of eliminating arcs and nodes that cannot possibly enter into the solution, or of detecting nodes and arcs that must be contained in the solution, as Koch and Martin [12] do...
with the Steiner problem. Duin and Volgenant [7] show that in the optimal solution of the HNDP, all secondary arcs must belong to the minimum spanning tree (MST) computed using secondary costs. Dual methods have been used that exploit this property [1,15]. Unfortunately, this property does not change the NP-hard status of the problem.

In the literature, the extreme nodes of the main path are known beforehand, except for a formulation [17] that relaxes the location of the origin node. In some applications, letting the model find the location of these nodes leads to better solutions. We propose a two-stage procedure for finding an optimal solution that solves a version of the HNDP in which both extremes are unknown a priori, and they must belong to sets of origin node candidates and destination node candidates, respectively.

The remainder of the paper is organized as follows. In Section 2, we present the multicommodity flow model for the HNDP, when the origin and destination nodes of the main path are preset. Section 3 describes the two-stage procedure for solving the HNDP. In Section 4, we introduce the model for the case in which the extreme nodes of the main path are to be found as part of the solution to the problem. Section 5 improves the method and shows extensive computational experience with the procedure.

2. The HNDP with fixed extreme nodes

For clarity of presentation, we first describe the model for the HNDP with known origin and destination nodes.

It is important to have in mind that the HNDP represents an undirected network that must be physically built. In this undirected network, each edge has a building cost that depends on the edge being primary or secondary— with a higher cost in the former case. This is the case, for example, in transportation networks or telecommunications networks. However, in order to find the optimal solution we use a procedure that assumes a directed network. Each edge is replaced by two directed arcs pointing in opposite directions, with the same cost, i.e., the cost matrix is symmetric. In the solution, at most one of these two arcs is built, and its cost is the building cost of the corresponding edge.

We use a modified version of the model used by Chopra and Tsai [3], for the TLNDP. Tours or cycles are avoided in the solution by sending a unit of flow from the origin to each other node. Given a directed graph $G = (N,A)$, each arc $(i,j)$ can become a primary or a secondary arc. The cost of a primary arc is $c_{ij}$, and the cost of a secondary arc is $d_{ij}$. Let $A$ be the set of candidate primary and secondary arcs. We define variables $x_{ij}$ equal to 1 if arc $(i,j)$ belongs to the primary path, and 0 otherwise; $y_{ij}$ equal to 1 if arc $(i,j)$ belongs to a secondary tree, and 0 otherwise; and $f_{ij}^d$ equal to 1 if there is flow with destination $k$ through arc $(i,j)$, and 0 otherwise. Node $o$ is the origin node and $d$ the destination node, both known beforehand in this case. For each node $k \in N \setminus \{o\}$, a flow is sent from $o$ to $k$, which precludes cycles. The model follows:

**HNDP:** Min $Z = \sum_{(i,j) \in A} c_{ij}x_{ij} + \sum_{(i,j) \in A} d_{ij}y_{ij}$

s.t. $\sum_{j : (o,j) \in A} f^k_{oj} = 1 \quad k \in N \setminus \{o\}$

(2)

$\sum_{i : (i,k) \in A} f^k_{ih} = 1 \quad k \in N \setminus \{o\}$

(3)

$\sum_{i : (i,j) \in A, j \neq k} f^k_{ij} = \sum_{h : (j,h) \in A, h \neq o} f^k_{jh} \quad k \in N \setminus \{o\}, j \in N \setminus \{k, o\}$

(4)

$f^o_{ij} \leq x_{ij} \quad (i,j) \in A : i \neq d, j \neq o$

(5)

$f^k_{ij} \leq x_{ij} + y_{ij} \quad k \in N \setminus \{o,d\}, (i,j) \in A : i \neq k, j \neq o$

(6)

$x_{ij} \in \{0,1\} \quad (i,j) \in A : i \neq d, j \neq o$

$y_{ij} \in \{0,1\} \quad (i,j) \in A$

The objective (1) minimizes the total cost. For each node except for the origin, constraints (2) and (3) force a flow path starting from the origin and ending at node $k$. Constraints (4) require flow conservation for each node $j$, except for the origin and the node $k$ to which the flow is being sent. By virtue of constraints (5), only primary arcs can carry flow oriented to the destination node $d$. Constraints (6) state that there is no flow where there is no arc. Remaining constraints are integrality and non-negativity constraints. Note that the flow variables form a tree composed by paths running from the origin to every other node. This formulation uses $O(n^3)$ variables, and $O(n^3)$ constraints.

3. The two-stage procedure for solving the HNDP

Duin and Volgenant [7] found the property (the “DV property”) stating that all secondary edges in the solution of the HNDP must belong to one of the minimum spanning trees of the original network, constructed using secondary costs. Then, we define variables $y_{ij}$ and $y_{ji}$ only for arcs that coincide with edges $(i,j)$ belonging to one of the MST’s. Note that, since all the MST’s have the same number of edges, the choice of MST is unimportant from the point of view of the DV property, as long as all the edges belong to the same MST. In our case, by virtue of the DV property, since the resulting optimal HNDP can only have secondary edges that belong to an MST, the flow destined to any node $k$ can only pass through arcs that coincide with edges belonging either to that MST or primary arcs belonging to the main path. Thus, there is no need to force flow to all nodes, but only to leaves of the MST. By sending flows only to the leaves, all the nodes in $N$ will necessarily be on some flow path to a leaf. Otherwise, if some node is not on a flow path, at least one leaf will not receive its flow, because of the uniqueness of the paths on the MST. This is always valid, except for the following case. Let $k$ be a node on an MST branch whose leaf is node $\ell$. If node $k$ belongs to the primary path, the flow being sent to $\ell$, instead of using the full branch, reaches $k$ through the primary path. Thus, predecessors of node $k$ on the branch remain “disconnected”. Hence, we can proceed by adding constraints as did Current et al. [4]. In our case, the constraints will be aimed at sending flow to the disconnected nodes. The leads to the following two-stage strategy:
First stage: Find an MST using secondary costs. Identify the
leaves of the MST.

Second stage:
- a. Solve the integer programming model. To avoid cycles, send
  flow to the leaves of the MST.
- b. If there are “disconnected” nodes, add constraints sending
  flows to these nodes and solve again.

Recall that each of the MST’s of the network will produce an
optimal solution. However, there could be some differences in
solution time depending on what MST is used, because the
number of variables of the model depends on the number of
leaves of the chosen tree.

4. The HNDP over the reduced network with free origin
and destination (R-HNDPFE)

In some of the applications of the HNDP, setting a priori
the origin and destination nodes could lead to sub-optimal
solutions. Instead, we assume that the origin and destination
nodes of the main path belong to the origin and destination
candidate node sets \( V_1 \subseteq N \) and \( V_2 \subseteq N \), respectively.
We define fictitious origin and destination nodes \( O \) and \( D \)
respectively, not in the original network. \( O \) is connected
to each node in \( V_1 \) through costless, directed arcs. These arcs can
carry flow from \( O \) to a node in \( V_1 \), but not in the opposite
sense. Likewise, each node in \( V_2 \) is connected through costless,
directed arcs to \( D \). Flow on these arcs can only be directed
towards \( D \).

Now, we formulate the HNDP over the reduced network
with free extremes, R-HNDPFE. Note that since the HNDP is a
particular case of the R-HNDPFE, this last problem is also NP-
hard. Let \( B \) be the set or arcs belonging to an MST, and \( T \)
the set of leaves of the MST. Variables \( y_{ij} \) are defined only for arcs
\((i, j) \in B \). There is no need to use flow variables for the main
path. We build this path using the shortest path formulation.

R-HNDPFE: Min \( Z = \sum_{(i, j) \in A} c_{ij} x_{ij} + \sum_{(i, j) \in A} d_{ij} y_{ij} \)
s.t. \( \sum_{j \in V_1} x_{Oj} = 1 \) \( \tag{7} \)
\( \sum_{i \in V_2} x_{iD} = 1 \) \( \tag{8} \)
\( \sum_{(i, j) \in A} x_{ij} - \sum_{(j, h) \in A} x_{jh} \)
\( \geq \begin{cases} -x_{Oj} & j \in V_1 \setminus V_2 \\ x_{jD} & j \in V_2 \setminus V_1 \\ x_{jD} - x_{Oj} & j \in V_1 \cap V_2 \\ 0 & j \in N \setminus (V_1 \cup V_2) \end{cases} \) \( \tag{9} \)
\( \sum_{(i, k) \in A} f_{ik} = \begin{cases} 1 - x_{Ok} & k \in T \cap V_1 \\ 1 & k \in T \setminus V_1 \end{cases} \) \( \tag{10} \)
\( \sum_{(i, j) \in A \setminus \{(i, k)\}} f_{ij} - \sum_{(j, h) \in A} f_{jh} \)
\( = \begin{cases} -x_{Oj} & k \in T, j \in V_1 : j \neq k \\ 0 & k \in T, j \in N \setminus V_1 : j \neq k \end{cases} \) \( \tag{11} \)
\( f_{ij}^{k} \leq x_{ij} + y_{ij} & k \in T, (i, j) \in B : i \neq k \) \( \tag{12} \)
\( f_{ij}^{k} \leq x_{ij} & k \in T, (i, j) \in A \setminus B : i \neq k \) \( \tag{13} \)
\( \sum_{(i, j) \in A} x_{ij} + \sum_{(h, j) \in B} y_{hj} \)
\( = \begin{cases} 1 & j \in V_1 \\ 1 & j \in N \setminus V_1 \end{cases} \) \( \tag{14} \)
\( x_{ij} + x_{ji} + y_{ij} + y_{ji} \leq 1 & (i, j) \in B : i < j \) \( \tag{15} \)
\( x_{ij}, x_{Op}, x_{qD} \in \{0, 1\} & (i, j) \in A, p \in V_1, q \in V_2 \)
\( y_{ij} \in \{0, 1\} & (i, j) \in B \)
\( f_{ij}^{k} \geq 0 & k \in T, (i, j) \in A : i \neq k. \)

Constraints (7)–(9) require a path connecting \( O \) to \( D \)
through nodes in the sets \( V_1, V_2 \). Constraints (10) and (11)
require flow conservation between the origin and each leaf of
the MST. Constraints (12) and (13) allow flow only through
<table>
<thead>
<tr>
<th>O set</th>
<th>D set</th>
<th>α</th>
<th>Opt</th>
<th>Main path</th>
<th>Current</th>
<th>OM</th>
</tr>
</thead>
<tbody>
<tr>
<td>12, 13, 21, 22, 23</td>
<td>66, 67, 68, 69</td>
<td>1.05</td>
<td>338.6</td>
<td>23-24-35-32-7-31-47-50-65-66</td>
<td>49, 11</td>
<td>1.17</td>
</tr>
<tr>
<td>1.50</td>
<td>344.0</td>
<td>23-24-35-32-7-31-47-50-65-66</td>
<td>24, 20</td>
<td>1.13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>362.0</td>
<td>23-24-35-32-7-31-47-50-65-66</td>
<td>20, 25</td>
<td>0.92</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>410.0</td>
<td>23-24-35-32-7-31-47-50-65-66</td>
<td>17, 95</td>
<td>1.16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38, 39, 57, 58</td>
<td>64, 65, 66, 67, 68, 69</td>
<td>1.05</td>
<td>338.6</td>
<td>58-77-78-88-84-82-70-65-66</td>
<td>49, 11</td>
<td>1.17</td>
</tr>
<tr>
<td>1.50</td>
<td>343.5</td>
<td>58-77-78-88-84-82-70-65-66</td>
<td>28, 44</td>
<td>1.98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>357.0</td>
<td>58-77-78-88-84-82-70-65-66</td>
<td>21, 97</td>
<td>2.09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>393.0</td>
<td>58-77-78-88-84-82-70-65-66</td>
<td>19, 78</td>
<td>1.69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38, 39, 57, 58</td>
<td>67, 68, 69</td>
<td>1.05</td>
<td>338.5</td>
<td>58-77-78-90-88-87-84-82-70-66</td>
<td>49, 11</td>
<td>1.17</td>
</tr>
<tr>
<td>1.50</td>
<td>342.5</td>
<td>58-77-78-90-88-87-84-82-70-66</td>
<td>25, 52</td>
<td>1.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>356.0</td>
<td>58-77-78-90-88-87-84-82-70-66</td>
<td>15, 53</td>
<td>1.28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>392.0</td>
<td>58-77-78-90-88-87-84-82-70-66</td>
<td>17, 33</td>
<td>1.23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>89</td>
<td>1.05</td>
<td>338.4</td>
<td>58-77-78-90-88-87-84-83</td>
<td>46, 30</td>
<td>0.92</td>
</tr>
<tr>
<td>1.50</td>
<td>341.5</td>
<td>58-77-78-90-88-87-84-83</td>
<td>22, 97</td>
<td>1.03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>352.0</td>
<td>58-77-78-90-88-87-84-83</td>
<td>16, 77</td>
<td>1.16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>380.0</td>
<td>58-77-78-90-88-87-84-83</td>
<td>16, 20</td>
<td>0.97</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38, 39, 57, 58</td>
<td>67, 68, 69, 83</td>
<td>1.05</td>
<td>338.4</td>
<td>58-77-78-90-88-87-84-83</td>
<td>46, 30</td>
<td>0.92</td>
</tr>
<tr>
<td>1.50</td>
<td>341.5</td>
<td>58-77-78-90-88-87-84-83</td>
<td>22, 97</td>
<td>1.03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>352.0</td>
<td>58-77-78-90-88-87-84-83</td>
<td>16, 77</td>
<td>1.16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>380.0</td>
<td>58-77-78-90-88-87-84-83</td>
<td>16, 20</td>
<td>0.97</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64, 65, 66</td>
<td>76, 77, 78</td>
<td>1.05</td>
<td>338.6</td>
<td>64-51-50-65-70-78-88-87-90-78-77-76</td>
<td>43, 97</td>
<td>1.45</td>
</tr>
<tr>
<td>1.50</td>
<td>343.5</td>
<td>64-71-86-88-90-78-77-76</td>
<td>21, 53</td>
<td>2.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>357.0</td>
<td>64-71-86-88-90-78-77-76</td>
<td>16, 63</td>
<td>1.80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>393.0</td>
<td>64-71-86-88-90-78-77-76</td>
<td>16, 91</td>
<td>2.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>90</td>
<td>1.05</td>
<td>338.2</td>
<td>76-77-78</td>
<td>57, 61</td>
<td>0.67</td>
</tr>
<tr>
<td>1.50</td>
<td>339.0</td>
<td>76-77-78</td>
<td>22, 64</td>
<td>0.69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>342.0</td>
<td>76-77-78</td>
<td>17, 05</td>
<td>0.72</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>350.0</td>
<td>76-77-78</td>
<td>15, 83</td>
<td>0.73</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29, 30, 48, 49</td>
<td>1, 10, 76, 100</td>
<td>1.05</td>
<td>338.6</td>
<td>48-50-65-70-82-84-87-88-90-78-77-76</td>
<td>146, 56</td>
<td>1.88</td>
</tr>
<tr>
<td>1.50</td>
<td>343.5</td>
<td>48-50-65-70-82-84-87-88-90-78-77-76</td>
<td>22, 06</td>
<td>0.94</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>360.0</td>
<td>48-50-65-70-82-84-87-88-90-78-77-76</td>
<td>16, 98</td>
<td>1.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>404.0</td>
<td>48-50-65-70-82-84-87-88-90-78-77-76</td>
<td>17, 16</td>
<td>1.34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29, 30, 48</td>
<td>91</td>
<td>1.05</td>
<td>338.4</td>
<td>48-50-65-70-82-84-87-89</td>
<td>44, 17</td>
<td>1.03</td>
</tr>
<tr>
<td>1.50</td>
<td>342.0</td>
<td>48-50-65-70-82-84-87-89</td>
<td>26, 56</td>
<td>0.81</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>354.0</td>
<td>48-50-65-70-82-84-87-89</td>
<td>17, 47</td>
<td>0.64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.00</td>
<td>386.0</td>
<td>48-50-65-70-82-84-87-89</td>
<td>17, 25</td>
<td>0.70</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. Final procedure and computational experience

We compared the results of our procedure with the method proposed by Current et al. [4]. Recall that Current’s procedure consists of solving a simple integer programming model, checking whether sub-tours appear in the solution, and if they appear, adding a sub-tour-breaking constraint and solving again. The number of required sub-tour-breaking constraints is potentially in \( O(2^n) \), where \( n \) is the number of nodes of the instance of the problem. Thus, this method can iterate for a very long time in some instances. The main difference from our procedure is that we solve at once the multicommodity flow integer model in Section 4, which has a polynomial number of constraints and precludes the appearance of sub-tours in the solution. Duin and Volgenant [7], on the other hand, use the DV property to reduce the size of the problem, while we not
only reduce the size of the problem (number of variables and constraints), but also use this property to realize that there is no need to send flow to all nodes, but only to the leaves of the MST, or their predecessors. Thus, we obtain an extra reduction in the number of flow variables and a faster solution procedure.

For the comparison between Current’s procedure and ours, we used the 100-node, 516-arc network of Fig. 1. The results are shown in Table 1. The columns show the origin and destination sets, the value of $\alpha = c_{ij}/d_{ij}$, the optimal solution, the main path, and the CPU times for Current’s procedure, and the proposed (OM) procedure that sends flow to the predecessors of the leaves. The average run time of Current’s procedure was 28.97 s, while the OM procedure took on average 1.14 s (a reduction of 96%). Note that the run times of the OM procedure have a very small variance. In contrast, Current’s procedure has considerable differences in run times for different values of $\alpha$.

The MST was found greedily, and it took negligible time as compared to the time it takes to solve the integer programming formulations. So did finding the leaves of the tree. For the second stage we used CPLEX 9.0.0 with AMPL Version 20021031 (Win32) on a Pentium 4 CPU 3.000 GHz with 448 MB RAM.

We never found disconnected nodes requiring additional flow constraints. When $\alpha = 1$, the solution is the MST. As $\alpha$ increases, the main path approaches the shortest path between the pair of nodes in the sets $V_1$ and $V_2$ that are closest. The length of the main path is bounded below by the shortest path between the closest nodes of the sets $V_1$ and $V_2$, and above by the path over the MST joining any two nodes belonging to these sets. For some instances, the location of the extreme points coincides with their location obtained by solving the shortest path problem, between the sets $V_1$ and $V_2$, but this is not always so, which confirms that leaving the extremes free provides better solutions.

Finally we remark that, although Current’s procedure is very efficient in most networks, there are cases in which it takes very long times, as shown in Table 2 for some particular origin–destination pairs. In these cases, we had to stop Current’s procedure after 10,000 s, without finding the optimal value. Our procedure is much faster (on average, the run time is at least $10^{-4}$ times shorter!). These differences are due to the fact that, if cheap solutions with sub-tours are available for the integer programming model in Current’s procedure, these cheap solutions will keep appearing at each iteration. In the case of the 100-node test network, there is a whole section of it that has small costs on the arcs (see the right-hand side of the network), which favors sub-tours. Our procedure, in contrast, precludes sub-tours directly in the model.

Acknowledgements

The authors gratefully acknowledge the support provided by FONDECYT, Projects Nos. 1040606 and 1070741, and by Instituto Científico Milenio “Complex Engineering Systems”.

References


Table 2
Cases in which Current’s procedure takes very long times

<table>
<thead>
<tr>
<th>O</th>
<th>D</th>
<th>$\alpha$</th>
<th>Opt</th>
<th>Main path</th>
<th>OM</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>95</td>
<td>1.05</td>
<td>341.4</td>
<td>1-10-13-9-8-7-6-94-95</td>
<td>10 000</td>
<td>0.91</td>
</tr>
<tr>
<td>1</td>
<td>1.50</td>
<td>368.0</td>
<td></td>
<td>1-2-9-8-7-6-94-95</td>
<td>10 000</td>
<td>0.86</td>
</tr>
<tr>
<td>3.00</td>
<td>452.0</td>
<td></td>
<td></td>
<td>1-2-9-8-7-6-94-95</td>
<td>10 000</td>
<td>0.66</td>
</tr>
<tr>
<td>7.00</td>
<td>676.0</td>
<td></td>
<td></td>
<td>1-2-9-8-7-6-94-95</td>
<td>10 000</td>
<td>0.55</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>1.05</td>
<td>340.9</td>
<td>2-9-8-7-6-94-95-96-100</td>
<td>10 000</td>
<td>0.91</td>
</tr>
<tr>
<td>1.50</td>
<td>367.5</td>
<td></td>
<td></td>
<td>2-9-8-7-6-94-95-96-100</td>
<td>10 000</td>
<td>0.69</td>
</tr>
<tr>
<td>3.00</td>
<td>456.0</td>
<td></td>
<td></td>
<td>2-9-8-7-6-94-95-96-100</td>
<td>10 000</td>
<td>0.64</td>
</tr>
<tr>
<td>7.00</td>
<td>692.0</td>
<td></td>
<td></td>
<td>2-9-8-7-6-94-95-96-100</td>
<td>10 000</td>
<td>0.59</td>
</tr>
<tr>
<td>10</td>
<td>92</td>
<td>1.05</td>
<td>340.7</td>
<td>10-13-9-8-7-6-92</td>
<td>10 000</td>
<td>0.95</td>
</tr>
<tr>
<td>1.50</td>
<td>365.0</td>
<td></td>
<td></td>
<td>10-13-9-8-7-6-92</td>
<td>10 000</td>
<td>0.95</td>
</tr>
<tr>
<td>3.00</td>
<td>446.0</td>
<td></td>
<td></td>
<td>10-13-9-8-7-6-92</td>
<td>10 000</td>
<td>0.81</td>
</tr>
<tr>
<td>7.00</td>
<td>662.0</td>
<td></td>
<td></td>
<td>10-13-9-8-7-6-92</td>
<td>10 000</td>
<td>0.81</td>
</tr>
</tbody>
</table>


