Gearing optimization

V.V. Lozin*†

Abstract

We consider an optimization problem that arises in machine-tool design. It deals with optimization of the structure of gearbox, which is normally represented by a graph. The edges of such a graph correspond to pairs of gear-wheels and the vertices stand for velocities. There is a designated input vertex and a set of output vertices. The problem is to create a graph with given number of output vertices while minimizing the total number of vertices. We present an integer programming formulation of this problem and propose an efficient solution in the special case of regular graphs.

Keywords: Gearbox optimization; Integer Programming; Geometric Graphs

1 Introduction

The origin of the problem studied in this paper lies in the area of machine-tool design. It deals with optimization of the structure of gearbox, which is normally represented by a graph. We use standard graph-theoretic terminology, which can be found, for instance, in [3]. In particular, a star is a complete bipartite graph, one of the parts of which contains exactly one vertex; this vertex is called *center* of the star. All vertices adjacent to the center have degree 1 and we call them *pendant* vertices.

Consider a gearbox with two rotating shafts. Velocity of the first shaft is constant and is defined by some external source. Let us represent this velocity by the central vertex of a star and call it the input vertex. By means of gear-wheels the rotation of the first shaft can be transmitted to the second one. We represent each pair of gear-wheels connecting the two shafts by an edge of the star. Different connections induce different velocities of the second shaft, and we associate each of these velocities with a pendant vertex of the star. In this simplest model, the pendant vertices of the star are called output vertices. In general, gearboxes and the respective graphs have more complicated structure consisting of several levels: each velocity of the second shaft can induce several velocities of the third one and so on. Figure 1 represents an example of a multilevel graph. We call such a graph *structural net*. In the multilevel model, the output vertices are those

*Mathematics Institute, University of Warwick, Coventry, CV4 7AL, United Kingdom. E-mail: V.Lozin@warwick.ac.uk

†The author gratefully acknowledges the support of DIMAP – the Center for Discrete Mathematics and its Applications at the University of Warwick

representing velocities of the last shaft (shown at the bottom of the net). In this paper we study the problem of creating a graph with given number of output vertices, while minimizing the total number of vertices. If there are no restrictions on the number of edges in each particular star, then the solution is trivial. However, in the real world situations such restrictions arise from some constructive restraints, which makes the problem more difficult.

In the present paper we first introduce a formal definition of the notion of structural net and describe a problem associated to it (Section 2), then propose an integer programming formulation of this problem (Section 3), and finally, present an efficient solution to the problem in the special case of regular nets (Section 4).

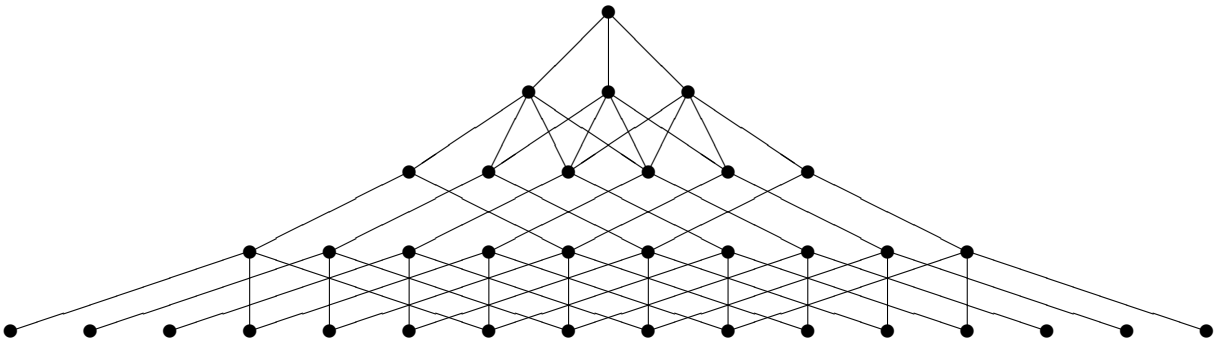

Figure 1: An example of a structural net

2 Definitions and the problem

Let $S_p(b)$ denote a star with central vertex b and p pendant vertices. We shall refer to p as the degree of the star. The set of pendant vertices of the star $S_p(b)$ will be denoted $N(b) = \{b_1, \ldots, b_p\}$.

A *proper realization* of a star $S_p(b)$ on the coordinate plane is a function $g(v) = (g_x(v), g_y(v))$, where v is a vertex of $S_p(b)$ and $(g_x(v), g_y(v))$ is a point on the plane with abscissa $g_x(v)$ and ordinate $g_y(v)$, enjoying the following properties:

- $g_y(b) - g_y(b_i) = 1$ for each $i = 1, \ldots, p$;
- $g_x(b_i) - g_x(b_{i-1}) = a > 0$ for each $i = 2, \ldots, p$;
- $g_x(b) = \frac{g_x(b_1)+g_x(b_p)}{2}$.

The value a, appeared in the definition, will be called the *density* of the realization. By $S_{p,a}(b)$ we shall denote a proper realization of the star $S_p(b)$ with density a. Throughout the paper by a star we mean a proper realization of the star on the coordinate plane.

A *structural net* $SN_h(b_0, B_h)$ of height h with input vertex b_0 and set of output vertices B_h can be defined by induction on h as follows:

(A) $SN_1(b_0, B_1)$ is defined to be $S_{p_1,a_1}(b_0)$. The set of output vertices of $SN_1(b_0, B_1)$ is $N(b_0)$;

(B) $SN_h(b_0, B_h)$ is obtained from $SN_{h-1}(b_0, B_{h-1})$ by adding to it a star $S_{p_h,a_h}(b)$ for each $b \in B_{h-1}$. The set of output vertices of $SN_h(b_0, B_h)$ is $\bigcup_{b \in B_{h-1}} N(b)$.

It is not difficult to see that a net is completely defined by two ordered sets of parameters $P = (p_1, p_2, \ldots, p_h)$ and $A = (a_1, a_2, \ldots, a_h)$. Our objective is to find a net such that the number of output vertices is equal to a given constant K, while its hight and total number of vertices is as small as possible. Clearly, a trivial solution to this problem is the star with K pendant vertices. However, usually the value of p_j $(j = 1, \ldots, h)$ is subject to a constraint. For instance, for $K = 15$ and $p_j \leq 2$, an optimal net is shown in Figure 2. It has height $h = 4$ and 30 vertices all together. An optimal net with the same amount of output vertices $K = 15$ but with a different constraint on p_j is shown in Figure 3. It has a smaller hight and smaller total number of vertices. Notice that in both examples p_j has the same value for each j (2 in Figure 2 and 3 in Figure 3). Such nets will be called *regular*. Further slackening the upper bound on p_j results in a better solution represented in Figure 4. This solution, however, is not regular anymore. Moreover, from the results of Section 4 it follows that there is no regular structural net of height $h = 2$ with $K = 15$.

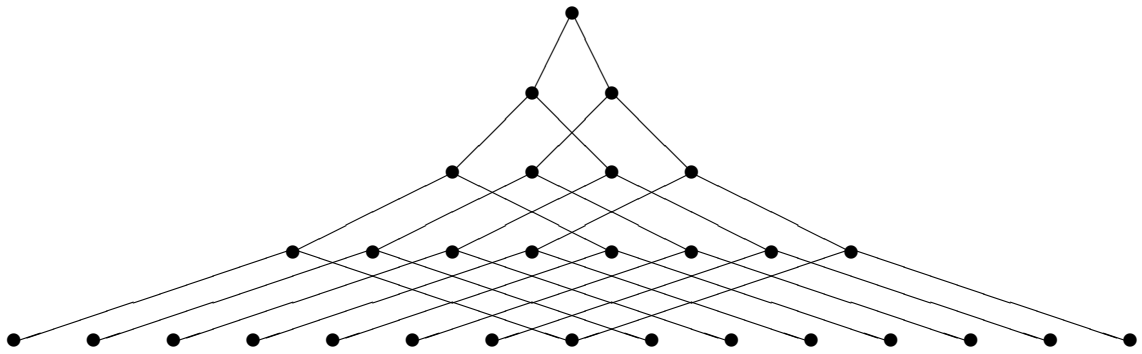

Figure 2: An optimal structural net with $K = 15$ and $p_j \leq 2$

In this paper we consider a simplified version of the problem, where vector $P = (p_1, p_2, \ldots, p_h)$ is given in advance as part of the input. With this simplification, the problem can be stated as follows:

Structural Net Optimization. Given a constant K and a vector $P = (p_1, p_2, \ldots, p_h)$ of positive integers, compute a vector $A = (a_1, a_2, \ldots, a_h)$ which minimizes the total number of vertices in the structural net defined by P and A under the constraint $|B_h| = K$.

We shall be looking at an integer solution to this problem, which is customary in applications. A net in which all components of vectors P and A are integers will be called *integral*. The next section proposes an integer programming formulation of the problem.

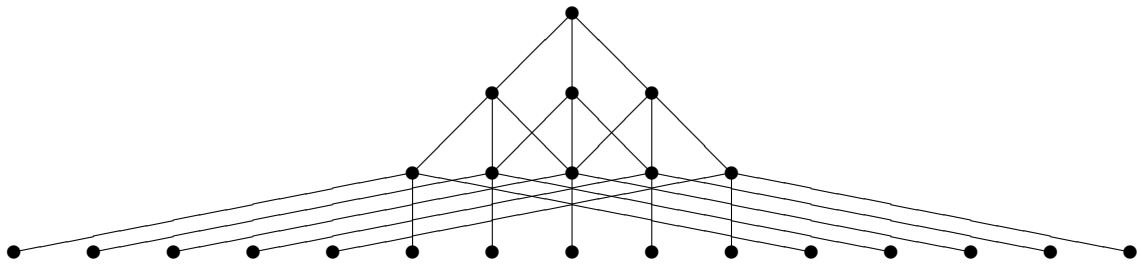

Figure 3: An optimal structural net with $K = 15$ and $p_j \leq 3$

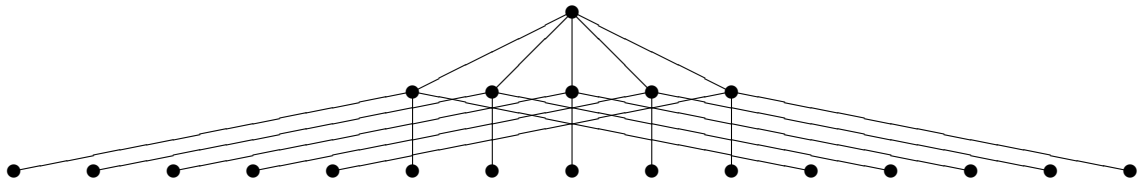

Figure 4: An optimal structural net with $K = 15$ and $p_j \leq 5$

3 Mathematical model

To unify notation, we shall denote $B_0 = \{b_0\}$. The set of all vertices of the net will be denoted $B = \bigcup\limits_{j \geq 0} B_j$. The vertices of B_j will be called the vertices of j-th level. The number of vertices of level j is denoted $K_j = |B_j|$. Obviously, for any net, $K_0 = 1$ and $K_1 = p_1$.

A star whose pendant vertices belong to B_j will be called the star of j-th level. For a subset of vertices $C \subseteq B_j$, the value

$$d(C) = \max_{c_i, c_j \in C} |g_x(c_i) - g_x(c_j)|$$

will be called *diameter* of C. The diameter of B_j will be denoted d_j, i.e. $d_j := d(B_j)$. The *width* of a star $S_{p_j, a_j}(b)$ is defined to be $d(N(b))$, i.e. the diameter of the set of its pendant vertices. It follows from definition that all stars of the same level have the same width, which will be denoted by A_j. It is not difficult to see that $d_j = d_{j-1} + A_j$ for any level j.

By definition of a star $S_{p_j,a_j}(b)$, the vertices in the set $N(b)$ are ordered according to their x-coordinates, and the i-th vertex in this set is denoted b_i. With this notation, we define $B_j(i) := \{b_i \in N(b) | b \in B_{j-1}\}$. In other words, $B_j(i)$ is the set of vertices consisting of the i-th vertices from all stars of level j. Notice that in general $B_j(1), \ldots, B_j(p_j)$ is not a partition of B_j.

As mentioned in the previous section, we shall consider only integral nets. The following lemma is simple, and hence we omit the proof.

Lemma 1 *In an integral net, the distance between any two vertices of the same level is integer.*

A trivial consequence from this lemma is

Corollary 1 *In an integral net, $d_j \geq K_j - 1$ for each $j = 1, \ldots, h$.*

We say that vertices u and v in B_j are *consecutive* if no other vertex of B_j is located between u and v.

Definition 1 *A set of vertices B_j will be called proper if any two consecutive vertices in the set are of distance 1. The set B_0 is defined to be a proper set.*

Directly from this definition and Corollary 1 we obtain

Lemma 2 *B_j is proper if and only if $d_j = K_j - 1$.*

Now we prove an important theorem which will be needed in the sequel.

Theorem 1 *If B_{j-1} is a proper set, then B_j is proper if and only if $a_j = \frac{K_j - K_{j-1}}{p_j - 1}$.*

Proof. To prove sufficiency, assume $a_j = \frac{K_j - K_{j-1}}{p_j - 1}$. Since B_{j-1} is proper, we have $d_{j-1} = K_{j-1} - 1$ by Lemma 2. In addition, we know that $A_j = d_j - d_{j-1}$ and $A_j = a_j(p_j - 1)$ (see definition of A_j). Therefore,

$$K_j = a_j(p_j - 1) + K_{j-1} = A_j + d_{j-1} + 1 = d_j + 1.$$

Thus, B_j also is proper by Lemma 2.

To prove necessity, assume that B_{j-1} and B_j both are proper. Then $d_{j-1} = K_{j-1} - 1$ and $d_j = K_j - 1$ by Lemma 2. From $d_j - d_{j-1} = A_j = a_j(p_j - 1)$ we obtain

$$a_j = \frac{d_j - d_{j-1}}{p_j - 1} = \frac{K_j - K_{j-1}}{p_j - 1}$$

as required. ■

Definition 2 *A structural net will be called proper if the set of vertices of each level is proper.*

From Theorem 1 and the fact that B_0 is proper by definition it follows that any structural net can be transformed into a proper one without changing the number of vertices in any level. The procedure of transforming a structural net into a proper one will be called *normalization*. We shall describe the normalization with a vector

$$\Delta a_1, \Delta a_2, \ldots, \Delta a_h,$$

where $\Delta a_j = a_j - a'_j$, and a'_j stands for the density of level j after the normalization, i.e. $a'_j = \frac{K_j - K_{j-1}}{p_j - 1}$.

To compute the number of vertices in each level j of a proper net, we shall use a simplified version of the inclusion-exclusion formula. To derive such a simplification, let us first simplify the notation as follows: denote $n := p_j$, $C := B_j$ and $C_i := B_j(i)$ for each $i = 1, \ldots, n$. The k-th element of the set C_i is denoted $C_i(k)$. For two elements a, b in C, we shall write $a < b$ to indicate that the x-coordinate of a is less than the x-coordinate of b. It is not hard to see that

(1) $C_{i-1}(1) < C_i(1)$ for each $i = 2, \ldots, n$,

(2) $C_{i-1}(k) > C_i(1)$ implies $C_{i-1}(k) \in C_{i-1} \cap C_i$.

The first property is obvious. To see the second one, observe that $C_{i-1}(k) < C_i(k)$ and every integer point between $C_i(1)$ and $C_i(k)$ belongs to C_i, as each level of the net is proper. Property (2) implies, in particular, that

(3) $(\bigcup_{l=1}^{i-1} C_l) \cap C_i = C_{i-1} \cap C_i$.

With these observations in mind we prove the following lemma.

Lemma 3 $|\bigcup_{i=1}^{n} C_i| = \sum_{i=1}^{n} |C_i| - \sum_{i=1}^{n-1} |C_i \cap C_{i+1}|$.

Proof. We use induction on n. For $n = 2$, the statement of the lemma coincides with the inclusion-exclusion formula (see e.g. [1]) and hence is true. Assume now that it is true for $n - 1$ sets, i.e. the following equality holds:

$$|\bigcup_{i=1}^{n-1} C_i| = \sum_{i=1}^{n-1} |C_i| - \sum_{i=1}^{n-2} |C_i \cap C_{i+1}|.$$

Consider n sets $C_1, \ldots, C_n$ and represent the set C as $C = C' \cup C_n$, where $C' = \bigcup_{i=1}^{n-1} C_i$. Applying the inclusion-exclusion formula for the two sets C' and C_n, we obtain:

$$|\bigcup_{i=1}^{n} C_i| = |C' \cup C_n| = |C'| + |C_n| - |C' \cap C_n| = |\bigcup_{i=1}^{n-1} C_i| + |C_n| - |(\bigcup_{i=1}^{n-1} C_i) \cap C_n|.$$

From this, taking into account the inductive hypothesis and properties (1), (2), (3), we conclude:

$$|\bigcup_{i=1}^{n} C_i| = \sum_{i=1}^{n-1} |C_i| - \sum_{i=1}^{n-2} |C_i \cap C_{i+1}| + |C_n| - |C_{n-1} \cap C_n| = \sum_{i=1}^{n} |C_i| - \sum_{i=1}^{n-1} |C_i \cap C_{i+1}|,$$

as required. ■

If $\prod_{i\geq 1} p_i < K$, then clearly there is no solution to our problem, since the product $\prod_{i\geq 1} p_i$ gives the maximum possible number of output vertices for the given input vector $P = (p_1, \ldots, p_h)$. In case when $\prod_{i\geq 1} p_i = K$ we call the net *complete*. In a complete net, the number of vertices in each level is given by

$$K_j^* = \prod_{i\leq j} p_i.$$

Therefore, by Theorem 1, a proper complete net is defined by

$$a_j^* = \prod_{i\leq j-1} p_i.$$

Assume now that $\prod_{i\geq 1} p_i > K$. In this case, we begin with a proper complete net, and for each $i > 1$ reduce the value of a_i^* by x_i in order to decrease the number of output vertices. Reduction of a_i by x_i will be called *compression* of level i, and x_i *compression value*. In what follows, we call a level j *complete* if $K_j = K_{j-1}p_j$.

Theorem 2 *Assume that levels $i, i+1, \ldots, h$ are proper and complete. If the value of a_i reduces by an integer $x_i < a_i$, then the number of vertices K_j reduces by ΔK_j , where*

$$\Delta K_i = x_i(p_i - 1),$$

$$\textit{for } j > i,\ \Delta K_j = \Delta K_{j-1}p_j.$$

Proof. In the set B_i let us consider two consecutive subsets $B_i(j)$ and $B_i(j+1)$. They are disjoint, since B_i is complete, and their union is a proper set, since B_i is proper. After reducing a_i by the integer x_i the intersection $B_i(j) \cap B_i(j+1)$ contains exactly x_i vertices, and the union $B_i(j) \cup B_i(j+1)$ remains proper. Denote by B_i' the set of vertices of level i after compression and let $K_i' = |B_i'|$. With Lemma 3 we can compute the number K_i' as follows:

$$K_i' = \sum_{j=1}^{p_i} |B_i(j)| - \sum_{j=1}^{p_i-1} |B_i(j) \cap B_i(j+1)| = K_i - x_i(p_i - 1).$$

Therefore, $\Delta K_i = K_i - K_i' = x_i(p_i - 1)$.

Since B_i' is a proper set, we have by Lemma 2 $d_i' = K_i' - 1$. Therefore, $\Delta d_i = d_i - d_i' = (K_i - 1) - (K_i' - 1) = \Delta K_i$. In order to determine Δd_j for $j > i$, observe that $d_j = d_{j-1} + A_j$ and A_j does not change under compression of level $i < j$. Therefore, we have

$$\Delta d_j = d_j - d_j' = (d_{j-1} + A_j) - (d_{j-1}' + A_j) = d_{j-1} - d_{j-1}' = \Delta d_{j-1}.$$

Thus, $\Delta d_j = \Delta K_i$ for each $j \geq i$.

By Theorem 1, $a_j = K_{j-1} = d_{j-1} + 1$. Therefore, $a_j > d_{j-1} > d_{j-1} - \Delta d_{j-1} = d_{j-1}'$ for $j > i$. This implies that every level $j > i$ remains complete after compression of level i. Therefore, for $j > i$, $K_j' = K_{j-1}'p_j$. Consequently, $\Delta K_j = K + j - K_j' = K_{j-1}p_j - K_{j-1}'p_j = \Delta K_{j-1}p_j$. ■

After compression, level i is no longer complete but remains proper. On the contrary, levels $i+1,\dots,h$ are still complete but they are not proper in general. To make them proper without changing the number of vertices, we apply the normalization.

Corollary 2 *In order to normalize the structural net obtained by compression of level i by x_i, one has to reduce values of a_j, for each $j > i$, by ΔK_{j-1}.*

Proof. It follows from the proof of Theorem 2 that level i remains proper after the compression. Therefore, by Theorem 1, to make levels $i+1,\dots,h$ proper we recalculate values of a_j for $j > i$ in such a way that the new value a'_j satisfies $a'_j = \frac{K'_j - K'_{j-1}}{p_j - 1}$, where $K'_j = K_j - \Delta K_j$ is the number of vertices of level j after the compression. Thus, for each $j > i$, we have

$$a_j - a'_j = a_j - \frac{K'_j - K'_{j-1}}{p_j - 1} = \frac{a_j(p_j - 1) - K'_j + K'_{j-1}}{p_j - 1} = \frac{K_j - K_{j-1} - K'_j + K'_{j-1}}{p_j - 1} =$$

$$\frac{(K_j - K'_j) - (K_{j-1} - K'_{j-1})}{p_j - 1} = \frac{\Delta K_j - \Delta K_{j-1}}{p_j - 1} = \frac{\Delta K_{j-1} p_j - \Delta K_{j-1}}{p_j - 1} = \Delta K_{j-1}.$$

■

Denote by $X = (x_1, x_2, \dots, x_h)$ the vector of all compression values. With each vector X we associate a vector $R = (r_1, r_2, \dots, r_h)$, where r_j is the total reduction of the number of vertices of level j calculated over all compressions. Clearly r_j depends only on $x_1, \dots, x_j$ and according to Theorem 2 can be expressed as follows:

$$r_j = \sum_{i=1}^{j} x_i(p_i - 1) \prod_{k=i+1}^{j} p_k.$$

Similarly, with vector X we associate a vector $N = (n_1, \dots, n_h)$, where n_j is the total reduction of the density of level j taken over all normalization operations. It is not hard to see that $n_1 = 0$ and $n_j = r_{j-1}$ for each $j > 1$.

Now let a_j and K_j denote the density and the number of vertices of level j in the structural net obtained from the proper complete net by compressing every level j by x_j followed by respective normalization. Then

$$a_j = a^*_j - x_j - n_j = \prod_{i=1}^{j-1} p_i - x_j - \sum_{i=1}^{j-1} x_i(p_i - 1) \prod_{k=i+1}^{j-1} p_k,$$

$$K_j = K^*_j - r_j = \prod_{i=1}^{j} p_i - \sum_{i=1}^{j} x_i(p_i - 1) \prod_{k=i+1}^{j} p_k.$$

Therefore, the structural net optimization problem can be formalized as follows.

$$\min \sum_{j=1}^{h} \Big(\prod_{i=1}^{j} p_i - \sum_{i=1}^{j} x_i(p_i - 1) \prod_{k=i+1}^{j} p_k\Big) \tag{0}$$

subject to

$$\prod_{i=1}^{h} p_i - \sum_{i=1}^{h} x_i(p_i - 1) \prod_{k=i+1}^{h} p_k = K, \tag{1}$$

$$x_j \geq 0, \text{ integer.} \tag{2}$$

In general, integer programming problems are NP-hard (see e.g. [2]). In the next section we propose an efficient solution to the problem (0)–(2) in the special case of regular nets.

4 Optimization of regular nets

Definition 3 *A structural net with $p_1 = p_2 = \ldots = p_h$ is called regular.*

For regular nets, the only input data are $P := p_1 = p_2 = \ldots = p_h$ and K. The height h, which in the general case also is part of the input (given implicitly), can now be defined as $h = \lceil \log_P K \rceil$. In other words, we shall assume that the following double inequality holds:

$$P^{h-1} < K \leq P^h.$$

Indeed, if $K > P^h$, then the problem has no solution, while the other inequality arises from the fact that we want to find a solution with minimum number of levels.

Rewriting problem (0)–(2) for regular nets, we obtain the following formulation.

$$\min \sum_{j=1}^{h-1} (P^j - \sum_{i=1}^{j} x_i(P-1)P^{j-i}) \tag{0'}$$

subject to

$$P^h - \sum_{i=1}^{h} x_i(P-1)P^{h-i} = K, \tag{1'}$$

$$x_j \geq 0, \text{ integer.} \tag{2'}$$

Theorem 3 *The problem (0')–(2') has a solution if and only if $P^h - K$ is a multiple of $P - 1$. Moreover, if a solution exists, it is unique and can be found in linear time.*

Proof. Let $P^h - K$ be a multiple of $P - 1$. Denote $I = (P^h - K)/(P - 1)$ and rewrite constraint (1') as

$$\sum_{i=1}^{h} x_i P^{h-i} = I. \tag{1''}$$

Assume there is a feasible vector $X = (x_1, x_2, \ldots, x_h)$ with $x_k \geq P$ for some index k. Then $k > 2$, since otherwise we have, on the one hand, $I \geq P^{h-1}$, and on the other hand, taking into account that $P^{h-1} < K$,

$$I = (P^h - K)/(P - 1) < (P^h - P^{h-1})/(P - 1) = P^{h-1}.$$

Now we show that if $x_k \geq P$ for $k > 2$, then X is not an optimal solution to the problem in question. To this end, consider vector $X' = (x'_1, x'_2, \ldots, x'_h)$ such that $x'_j = x_j$ for

$j \neq k, k-1$, and $x'_{k-1} = x_{k-1} + s$, $x'_k = t$, where $s > 0$ and $t < P$ are non-negative integers uniquely determined from the equality $x_k = sP + t$. Let us show that X' is a feasible solution, and the objective value for X' is strictly less than for X.

Since X and X' coincide everywhere except for two coordinates $k-1$ and k, it suffices to consider only the two terms of the sum $(1'')$ corresponding to those coordinates:

$$x'_{k-1}P^{h-(k-1)} + x'_kP^{h-k} = (x_{k-1} + s)P^{h-k+1} + tP^{h-k} =$$
$$x_{k-1}P^{h-k+1} + sPP^{h-k} + tP^{h-k} = x_{k-1}P^{h-(k-1)} + x_kP^{h-k}$$

Therefore, X' satisfies $(1'')$.

Now let us analyze the objective function value corresponding to X':

$$\sum_{j=1}^{h-1} P^j - \sum_{j=1}^{h-1}\sum_{i=1}^{j} x'_i(P-1)P^{j-i} = \sum_{j=1}^{h-1} P^j - (P-1)\sum_{i=1}^{h-1} x'_i \sum_{j=0}^{h-i-1} P^j.$$

As before, it is sufficient to consider only the two terms of the second sum of the last expression corresponding to x'_{k-1} and x'_k:

$$x'_{k-1}\sum_{j=0}^{h-k} P^j + x'_k\sum_{j=0}^{h-k-1} P^j = (x_{k-1}+s)\sum_{j=0}^{h-k} P^j + t\sum_{j=0}^{h-k-1} P^j =$$
$$x_{k-1}\sum_{j=0}^{h-k} P^j + s + s\sum_{j=1}^{h-k} P^j + t\sum_{j=0}^{h-k-1} P^j =$$
$$x_{k-1}\sum_{j=0}^{h-k} P^j + s + sP\sum_{j=0}^{h-k-1} P^j + t\sum_{j=0}^{h-k-1} P^j =$$
$$x_{k-1}\sum_{j=0}^{h-k} P^j + x_k\sum_{j=0}^{h-k-1} P^j + s.$$

This expression appears in the objective function negatively, so the objective value for X' is smaller than for X.

The above discussion leads to the conclusion that under constraints $(1'')$ and $(2')$ the optimal objective value is attained only on those vectors every component of which is strictly less than P. If such a vector exists, then, according to $(1'')$, its components are coefficients in representation of the number I in the number system with base P. The existence and uniqueness of such vector is due to the fact that any number can be uniquely represented in any number system. Clearly this vector can be computed in linear time. This completes the proof of the "if" part of the theorem.

Conversely, assume the problem has a solution. Then, according to $(1')$, $P^h - K$ is a multiple of $P-1$. This implies uniqueness of the solution and a linear-time algorithm to find it, as was shown before. ■

References

[1] M. Aigner, *Combinatorial Theory*, Springer-Verlag, New York/Berlin, 1979.

[2] M.R. Garey and D.S. Johnson, Computers and intractability. A guide to the theory of NP-completeness. A Series of Books in the Mathematical Sciences. W. H. Freeman and Co., San Francisco, Calif., 1979.

[3] F. Harrary, *Graph Theory*, Addison-Wesley, Reading, MA, 1969.