Gerhard J. Woeginger
Faculteit Wiskunde en informatica
Technische Universiteit Eindhoven
Faculteit Elektrotechniek, Wiskunde en Informatica
Universiteit Twente, Postbus 217, 7500 AE Enschede
g.j.woeginger@math.utwente.nl

Computational problems

Problemen uit de discrete wiskunde lijken op het eerste gezicht vaak erg simpel. Ze kunnen meestal gemakkelijk en zonder gebruik te maken van wiskundige begrippen worden geformuleerd. Toch komt het vaak voor dat zo'n ogenschijnlijk eenvoudig probleem nog open is of dat er, zoals bij het handelsreizigersprobleem, wel een oplossing gegeven kan worden, maar alleen een die onbruikbaar is omdat de rekentijd bij grotere getallen te snel groeit. In dit artikel, gebaseerd op zijn voordracht op het NMC 2002, kijkt Gerhard Woeginger naar de tegenovergestelde situatie. Hij introduceert allerlei discrete problemen die onoplosbaar lijken, maar waarvoor er een simpele oplossing bestaat.

Computational Mathematics and Computer Science are full of innocent looking algorithmic problems that are difficult to solve. For these problems, the solution of moderately sized instances may take many many years, even if we use smart programs and run them on the fastest available computers. For instance:

- consider the product of two 300-digit prime numbers p and q, and compute p and q from this product;
- consider a 100×100 matrix with integer entries. Can one permute the rows and the columns of this matrix in such a way that in the resulting matrix all non-zero entries lie on the 20 diagonals above and below the main diagonal;
- consider a logical formula with 200 Boolean variables. Can one find a truth-setting for the variables such that the formula is satisfied? Or for instance:
- consider a list with 100 positive integers, each of them 100 digits long. Are there two distinct subsets whose elements add up to the same sum?

Of course the problems as stated are just toy problems. However, closely related combinatorial problems lie at the foundations of cryptography, artificial intelligence, mathematical economics, telecommunication, software analysis, VLSI design, and many other areas. Many of these problems are computationally intractable, and thirty years of research did not result in a satisfying computational way of attacking them. All known solution approaches boil down to enumerating zillions of subcases, and it seems impossible to control the resulting combinatorial explosion for big instances. For more information on these issues we refer the reader to the book by Garey & Johnson [1979] on NP-hardness, and to the book by Papadimitriou [1994] on computational complexity.

In this article we will not discuss such difficult problems. Instead, we will concentrate on easy problems that only look difficult at first sight. We will discuss the problem of arranging data records in a linear storage array so as to minimize the average access time, we will analyze the travelling salesman problem under specially structured distances, and we will consider a balancing problem for the blades on a hydraulic turbine wheel. We will argue that these three problems are absolutely trivial, and that they can be solved without any computation. It will turn out that there is *one* mathematical theorem in the background that explains why they are trivial.

Average access time in a linear storage array

In our first optimization problem, we consider an information retrieval system with a set R of n data records $R_1, \ldots, R_n$ that are referenced and accessed repeatedly. Data record R_i is referenced with probability p_i ($0 \leq p_i \leq 1$), and different references are mutually stochastically independent. Without loss of generality we assume $p_1 \leq p_2 \leq \cdots \leq p_n$. These n data records are to be stored in a linear array of storage cells $C_1, \ldots, C_n$, like a magnetic tape. Then they will repeatedly be accessed by a read head that moves along this storage array. When moving from storage cell C_i to cell C_j, the read head travels a distance $d_{i,j}$. See Figure 1 for an illustration.

What is the *best possible* ordering of the records in the array such that the read head's expected moving time between consec-

without computation

utively referenced records is minimized? Consider an ordering $\phi \in S_n$ of the data records that assigns record $R_{\phi(i)}$ to the storage cell C_i. The probability that record $R_{\phi(i)}$ is accessed and that right after it record $R_{\phi(j)}$ is accessed equals $p_{\phi(i)}p_{\phi(j)}$, and the distance between the cells is $d_{i,j}$. Hence, we wish to find a permutation $\phi \in S_n$ that minimizes the expression

$$\sum_{i=1}^{n}\sum_{j=1}^{n} p_{\phi(i)}p_{\phi(j)}\, d_{i,j}. \tag{1}$$

Intuitively one expects that the records with high access probabilities should be stored around the center of the storage array, whereas the records with low access probabilities should go to the far ends of the array. In a natural special case of this optimization problem the storage cells are laid out on the integers $1, \ldots, n$ and hence their distances are $d_{i,j} = |i - j|$. This special case was studied by Timofeev & Litvinov [1969], and thirty years later rediscovered and reproduced (independently) by Vickson & Lu [1998]. These authors proved that the optimal solution is always given by the same fixed permutation ϕ^*. In other words, there exists a *universal optimal ordering* ϕ^* of the data records for all possible instances with $d_{i,j} = |i - j|$. This universal ordering only depends on the ranking of the probabilities; it does not depend on their actual values!

Now, what does this universal solution ϕ^* look like? If we write $\phi = \langle \phi(1), \phi(2), \ldots, \phi(n)\rangle$ to specify a permutation ϕ, then

$$\phi^* = \langle 1, 3, 5, 7, 9, 11, 13, \ldots 14, 12, 10, 8, 6, 4, 2\rangle. \tag{2}$$

The permutation ϕ^* starts with all the odd numbers in increasing order, followed by all the even numbers in decreasing order. If we want to solve an instance of the data arrangement problem with distances $d_{i,j} = |i - j|$, then we do not need to check thousands of cases and subcases, we do not need to write smart computer programs, and we do not need to use a computer. In fact, no computation is necessary at all: We simply use the permutation ϕ^*.

What about other, less restricted distances $d_{i,j}$? Burkov, Rubinshtein & Sokolov [1969] considered the case where the distances satisfy $d_{i,j} = f(|i - j|)$ for some non-decreasing and convex function f. For $f(x) = x$ we again get the special case considered by Timofeev & Litvinov. For $f(x) = x^2$ we get the special case $d_{i,j} = (i - j)^2$ where travel times increase quadratically in $|i - j|$. Surprisingly, it turns out that this more general version is also universally solved by the permutation ϕ^*. Again, no computation is needed to handle instances of this type! The proof of Burkov, Rubinshtein & Sokolov starts with an arbitrary permutation, and then restructures, modifies, and rebuilds it, bringing it closer and closer to permutation ϕ^*. By exploiting the convexity of f, they show that with every such step of restructuring, modifying, and rebuilding, the objective value does not go up. In the end they arrive at permutation ϕ^* without increasing the objective value. Consequently, permutation ϕ^* must also yield an optimal solution. This proof is heavily based on the convexity of f. Is the result itself only true for convex functions f? No!

Metelski [1972] and Pratt [1972] realized that convexity is not necessary for the result. Metelski and Pratt also realized that the above results are all contained in a much older result due to

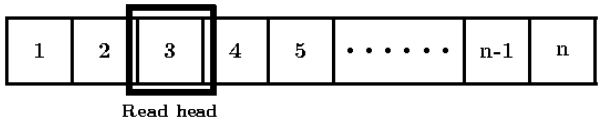


Figure 1 Data records in a linear storage array. The read head is accessing the data in cell 3.

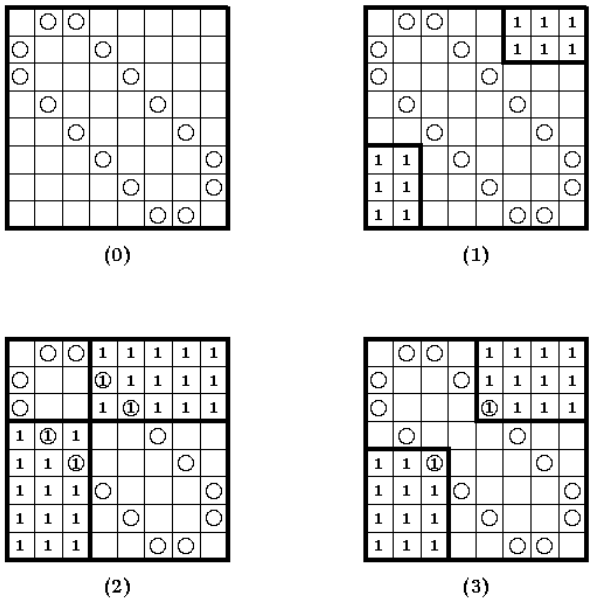


Figure 2 Illustrations for the proof of Supnick's result

Hardy, Littlewood & Pólya [1926]: Translated into the language of our article, this much older result states that the data arrangement problem is universally solved by the permutation ϕ^*, as long as the distances satisfy $d_{i,j} = f(|i-j|)$ with an arbitrary non-decreasing function f. This result can also be found in the well-known book *Inequalities* by Hardy, Littlewood & Pólya [1934]; its proof is discussed in Chapter 10 dealing with rearrangements of sequences. Let us summarize that there is a rich family of distance functions for which the data arrangement problem can be solved without computation.

The travelling salesman problem on Supnick matrices

Our second optimization problem is the *travelling salesman problem*: The goal is to find a shortest closed tour through n cities whose distances are specified by an $n \times n$ distance matrix $C = (c_{i,j})$. In other words, a travelling salesman starts from his hometown, visits all the other cities exactly once, in the end again returns to his hometown, and he does this with the minimum possible amount of gas. Mathematically, the salesman starts in city $\phi(1)$, then visits cities $\phi(2), \phi(3), \ldots, \phi(n)$, and then returns home. We wish to find a permutation $\phi \in S_n$ that minimizes the expression

$$\left(\sum_{i=1}^{n-1} c_{\phi(i),\phi(i+1)} \right) + c_{\phi(n),\phi(1)} \tag{3}$$

The travelling salesman problem is probably the most prominent and most studied problem in combinatorial optimization. It can be traced back to the work of the Irish mathematician Sir William Rowan Hamilton [1858]. The travelling salesman problem models situations that arise in robotics, production, scheduling, engineering, and many other areas. For more specific information on the travelling salesman problem and its applications, we refer the reader to the book by Lawler, Lenstra, Rinnooy Kan & Shmoys [1985].

In its general form the travelling salesman problem is an NP-hard problem; see Garey & Johnson [1979]. Hard problems that require lots of computation are not discussed in this article, so we will concentrate on the travelling salesman problem with a more restricted type of distance matrices, so-called *Supnick* matrices. An $n \times n$ matrix $C = (c_{i,j})$ is a Supnick matrix, if (i) it is symmetric and if (ii) it satisfies the Monge inequalities

$$c_{i,j} + c_{r,s} \;\leq\; c_{i,s} + c_{r,j} \tag{4}$$

for all $1 \leq i < r \leq n$ and $1 \leq j < s \leq n$. In words: In every 2×2 submatrix the sum of the two entries on the main diagonal is less than or equal to the sum of the two entries on the other diagonal. These inequalities go back to the eighteenth century, and to the work of the French mathematician Gaspard Monge [1781]. Burkard, Klinz & Rudolf [1996] survey the role of Monge structures in combinatorial optimization. Here are some examples of Supnick matrices:

- Sum matrices:
  Let $\alpha_1, \ldots, \alpha_n$ be real numbers. Then the sum matrix C with $c_{i,j} = \alpha_i + \alpha_j$ is a Supnick matrix. In fact, for a sum matrix the inequality signs in (4) can be replaced by equality signs
- Negative product matrices:
  Let $0 < \beta_1 \leq \beta_2 \leq \cdots \leq \beta_n$ be positive real numbers. Then the negative product matrix C with $c_{i,j} = -\beta_i \cdot \beta_j$ is a Supnick matrix. The inequalities in (4) are equivalent to $(\beta_r - \beta_i)(\beta_s - \beta_j) \geq 0$ with $r > i$ and $s > j$.
- LL-UR block matrices:
  Let $1 \leq x < y \leq n$ be integers, and consider the Lower-Left Upper-Right block matrix C with $c_{i,j} = 1$ if $i \leq x$ and $j \geq y$ or if $i \geq y$ and $j \leq x$, and with $c_{i,j} = 0$ in all other cases. This matrix C is a Supnick matrix. It has a rectangular block of 1-entries in the lower left corner (below the main diagonal), a symmetric block of 1-entries in the upper right corner (above the main diagonal), and it has 0-entries everywhere else. See Figure 3 for an illustration.

Note that the inequalities stated in (4) are *linear* inequalities. Hence, if we multiply a Supnick matrix by a positive real, or if we add two Supnick matrices, we will always end up with another Supnick matrix: The Supnick matrices form a cone. Rudolf & Woeginger [1995] investigated this cone and its extremal rays, and they came up with the following simple characterization of Supnick matrices: A matrix C is a Supnick matrix if and only if it

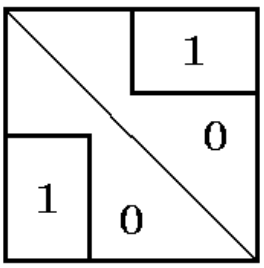


Figure 3 A Lower-Left Upper-Right block matrix.(LL-UR block matrix)

can be written as the sum of a sum matrix S and a non-negative linear combination of LL-UR block matrices.

Let us now return to the travelling salesman problem: Supnick [1957] proved via a fairly involved exchange argument that for the travelling salesman problem with Supnick distance matrices, the optimal tour is always given by choosing $\phi = \phi^*$ in Expression (3). Surprise! Again, the optimal solution ϕ^* can be written down without looking at the actual data and the actual city distances. Again, no computation is needed! And again, it is the permutation ϕ^* that constitutes the universal optimal solution! Supnick's result was rediscovered once by Rubinshtein [1971] and once by Michalski [1987]. In the following paragraphs we will present another, simple and straightforward argument for Supnick's result that is based on the above mentioned additive characterization of Supnick matrices in terms of sum matrices and LL-UR block matrices.

The travelling salesman problem with sum distance matrices C with $c_{i,j} = \alpha_i + \alpha_j$ is completely uninteresting: Every city i contributes the value $2\alpha_i$ to the total tour length, and thus every possible tour has length $2\sum_{i=1}^{n} \alpha_i$. Every permutation $\phi \in S_n$ minimizes the expression in (3), and, in particular, the permutation ϕ^* constitutes an optimal solution for the travelling salesman problem on sum matrices.

The travelling salesman problem on LL-UR block matrices is more interesting. For technical reasons, we will now *double* every travelling salesman problem tour and traverse it once in forward and once in backward direction. Since the distances are symmetric, this simply doubles the total tour length. Optimal solutions remain optimal, and non-optimal solutions remain non-optimal. Let us take a closer look at such a doubled tour corresponding to the permutation ϕ^*: In the forward direction, the doubled tour runs from city 1 to city 3, from city 3 to city 5, from 5 to 7 and so on. In the backward direction, it runs from city 2 to city 4, from 4 to 6, and so on. Hence, the doubled tour picks the entries $c_{i,i+2}$ and $c_{i+2,i}$ for $i = 1, \ldots, n-2$ together with the four entries $c_{1,2}$, $c_{2,1}$, $c_{n-1,n}$, $c_{n,n-1}$ out of the distance matrix, and it pays their total value. All the picked entries lie in the two diagonals above and in the two diagonals below the main diagonal of C. See the first matrix in Figure 2 for an illustration.

Let us now argue that the doubled tour for ϕ^* is the cheapest doubled tour for any LL-UR block matrix C. We distinguish three cases that depend on the size and position of the two rectangular blocks of 1-entries. We recall that the lower left corner of the upper right block is the matrix element $c_{x,y}$ with indices $1 \leq x < y \leq n$.

- In the first case $y - x \geq 3$, and the rectangular blocks in matrix C do not touch the doubled tour ϕ^*; see Figure 2, Matrix (1). Then the corresponding cost is 0, which clearly is optimal.
- In the second case $y - x = 1$. Then the rectangular blocks in C cover four of the entries picked by the doubled ϕ^*, and the corresponding cost is 4. See Figure 2, Matrix (2) for an illustration. In this case, the cities in $G_1 = \{1, \ldots, x\}$ are pairwise at distance 0, and the cities in $G_2 = \{x+1, \ldots, n\}$ are pairwise at distance 0. The distance between any city in G_1 and any city in G_2 equals 1. Any travelling salesman tour must go at least once from G_1 into G_2, and at least once back from G_2 into G_1. Hence, in this case, any tour has cost at least 2 and any doubled tour has cost at least 4. Again permutation ϕ^* yields an optimal solution.

Figure 4 Gaspard Monge (1746–1818) is well-known for his work on partial differential equations, calculus of variations and combinatorics. He contributed to various parts of science. As a great admirer of Napoleon, he joined Napoleon's expeditionary force into northern Africa.

- In the last case $y - x = 2$: see Figure 2, Matrix (3). In this case, the cost of the doubled tour ϕ^* equals 2. Every travelling salesman problem tour must contain some move from a city with number $\leq x$ to a city with number $\geq y$, or a move from a city $\geq y$ to a city $\leq x$. Therefore any tour has cost at least 1, any doubled tour has cost at least 2, and also in this case permutation ϕ^* yields an optimal solution.

Summarizing, we have shown that permutation ϕ^* yields the optimal travelling salesman problem tour for every distance matrix that is an LL-UR block matrix and for every distance matrix that is a sum matrix. Then ϕ^* also yields the optimal travelling salesman problem tour for any non-negative linear combination of such matrices, and by the result of Rudolf & Woeginger these combinations are exactly the Supnick matrices. The proof of Supnick's result is complete.

The problem of balancing hydraulic turbine runners

Our third optimization problem concerns the static balancing of turbine fans. A hydraulic turbine runner of the so-called Francis design consists of a cylinder around which a number of blades are welded at regular spacings. A blade may weigh up to 16 tons, but due to manufacturing imperfections the blades are not identical; their weights as well as the locations of their centers of gravity vary. Mosevich [1986] mentions differences as large as $\pm 5\%$ in the weights of the blades. Because of the differences in the weights, the center of gravity of the runner may not coincide with the axis of rotation, and this then leads to unbalance. Unbalance results in vibrations and in excess stresses on the supports, and thus shortens the life of the runner. Therefore, one of the main objectives during the welding process is to find a 'good' sequence of the blades around the cylinder which minimizes the deviation of the runner's center of mass from the runner axis. This balancing problem has been discussed by Bolotnikov [1978], by Stoyan, Sokolovskii & Yakovlev [1982], by Schlegel [1987], and by Laporte & Mercure [1988]. See Figure 6 for an illustration.

In a mathematical formulation of this problem (Laporte & Mercure [1988]), we are given n positive masses (=blades) $0 < m_1 \leq m_2 \leq \cdots \leq m_n$. These masses have to be assigned to the n vertices $v_i = (\sin(2i\phi/n), \cos(2i\phi/n))$, $1 \leq i \leq n$, of a regular polygon (=cylinder) whose center (=runner axis) lies in the origin; see Fig-

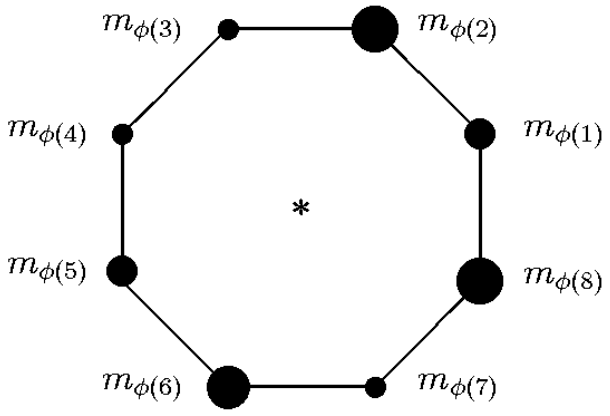


Figure 5 Eight masses assigned to the vertices of a regular octagon

ure 5. The goal is to assign the masses in such a way that the center of gravity of the resulting weighted polygon is as close to the origin as possible. That is, we want to find a permutation $\phi \in S_n$ that minimizes the Euclidean norm of the two-dimensional vector

$$\sum_{i=1}^{n} m_{\phi(i)} \begin{pmatrix} \sin \frac{2i\pi}{n} \\ \cos \frac{2i\pi}{n} \end{pmatrix}. \tag{5}$$

The square of the Euclidean norm of this vector equals

$$\begin{aligned} &(\sum_{i=1}^{n} m_{\phi(i)} \sin \tfrac{2i\pi}{n})^2 + (\textstyle\sum_{i=1}^{n} m_{\phi(i)} \cos \frac{2i\pi}{n})^2 = \\ &= \sum_{i=1}^{n} m_{\phi(i)}^2 + \\ &\quad \sum_{i=1}^{n} \sum_{j=1}^{n} 2 m_{\phi(i)} m_{\phi(j)} \, (\sin \tfrac{2i\pi}{n} \, \sin \tfrac{2j\pi}{n} + \cos \tfrac{2i\pi}{n} \, \cos \tfrac{2j\pi}{n}) \\ &= \sum_{i=1}^{n} m_{\phi(i)}^2 + 2 \sum_{i=1}^{n} \sum_{j=1}^{n} m_{\phi(i)} m_{\pi(j)} \, \cos \tfrac{2(i-j)\pi}{n}. \end{aligned}$$

Since the value of $\sum_{i=1}^{n} m_{\phi(i)}^2 = \sum_{i=1}^{n} m_i^2$ does not depend on the permutation ϕ, the minimization of the Euclidean norm of the vector in (5) is equivalent to the minimization of the expression

$$\sum_{i=1}^{n} \sum_{j=1}^{n} m_{\phi(i)} m_{\phi(j)} \, \cos \frac{2(i-j)\pi}{n}. \tag{6}$$

Unfortunately, Burkard, Çela, Rote & Woeginger [1998] proved that the just described minimization version of the turbine balancing problem is NP-hard. This means that the minimization version is very difficult to solve and needs exponential time unless the complexity classes P and NP coincide. This problem certainly does not belong to the optimization problems that can be solved without computation, and it is of no further interest for this article.

Let us instead consider the *maximization* version of the turbine balancing problem, where the goal is to get the center of gravity of the weighted polygon as far away from the origin as possible. That is, we want to find a permutation $\phi \in S_n$ that maximizes the expression in (6). Intuitively, all the heavy masses should go on one side of the polygon and all the light masses should go on the opposite side. Good! Maximization looks a lot easier than minimization! And in fact, Çela & Woeginger [1994] proved that the permutation ϕ^* yields a universal optimal solution for this maximization version of the turbine balancing problem. Once again we can get the optimal solution without any computation, and once again the permutation ϕ^* plays a central role.

A question

The permutation ϕ^* shows up as a universal optimal solution for all three optimization problems that we have touched so far: in the data arrangement problem where the distances satisfy $d_{i,j} = f(|i-j|)$ with an arbitrary non-decreasing function f; in the travelling salesman problem where the underlying distance matrix is a Supnick matrix; in the maximization version of the turbine balancing problem. Is there a mathematical explanation for this triple occurrence of ϕ^*? What are the common properties of these three problems that make them all solvable — without any further computation — by permutation ϕ^*?

Some more definitions

A matrix $A = (a_{i,j})$ with $1 \leq i, j \leq n$ is called *monotone* if $a_{i,j} \geq a_{i,j+1}$ and $a_{i,j} \geq a_{i+1,j}$ hold for all indices i, j. That is, the entries in every row and in every column are non-increasing. A matrix A is called a *Monge* matrix if it satisfies the Monge inequalities in (4). We stress that Monge matrices are not necessarily symmetric.

An $n \times n$ matrix $B = (b_{i,j})$ is called a symmetric *Toeplitz* matrix, if there exists a function $f\colon \{0, \ldots, n-1\} \rightarrow \mathbf{R}$ such that $b_{i,j} = f(|i-j|)$ for $1 \leq i, j \leq n$. The symmetric Toeplitz matrix B is said to be *generated* by this function f. A symmetric Toeplitz matrix is constant along every diagonal that is parallel to the main diagonal. Note that such a matrix is fully determined by the n entries in its first row.

The *quadratic assignment problem* in Koopmans-Beckmann form (Koopmans & Beckmann [1957]) takes as input two $n \times n$ matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ with real entries. The objective is to find a permutation $\phi \in S_n$ that minimizes the cost function

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{\phi(i),\phi(j)} \, b_{i,j}. \tag{7}$$

The quadratic assignment problem is a notoriously difficult NP-hard problem. Some quadratic assignment problem instances for $n = 30$ had remained unsolved for decades, and have only recently been solved by Anstreicher, Brixius, Goux & Linderoth [2002]. We refer the reader to the survey paper by Lawler [1963] and to the book by Çela [1998] for more information on the quadratic assignment problem. For us (and for this article) the quadratic assignment problem is relevant, since it provides a common generalization of our three optimization problems.

How to rewrite the three optimization problems

We will now bring our three optimization problems into the form of a quadratic assignment problem with a monotone Monge ma-

trix A and a symmetric Toeplitz matrix B. This quadratic assignment problem formulation is straightforward to find for the data arrangement problem where the distances satisfy $d_{i,j} = f(|i-j|)$ with an arbitrary non-decreasing function f: The minimization of the expression in (1) is equivalent to determining

$$\min_{\phi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} (-p_{\phi(i)} p_{\phi(j)})(-f(|i-j|)). \tag{8}$$

The matrix $A = (a_{i,j})$ with $a_{i,j} = -p_i p_j$ is a negative product matrix, and hence a Monge matrix. Since $0 \leq p_1 \leq p_2 \leq \cdots \leq p_n$, this matrix A is also monotone. The matrix $B = (b_{i,j})$ with $b_{i,j} = -f(|i-j|)$ is a symmetric Toeplitz matrix. Therefore, the optimization problem in (8) is a quadratic assignment problem with a monotone Monge matrix and a symmetric Toeplitz matrix.

Next, let us consider the travelling salesman problem on Supnick matrices. Every Supnick matrix $C = (c_{i,j})$ is by definition a Monge matrix, but it is not necessarily a monotone matrix. Here is a cheap way of making C monotone: Let $\gamma = \max_{i,j} 2|c_{i,j}|$, and define the sum matrix $S = (s_{i,j})$ with $s_{i,j} = -(i+j)\gamma$. In our above discussion of the travelling salesman problem, we observed that adding a sum matrix S to an arbitrary distance matrix C is irrelevant for the travelling salesman problem. A travelling salesman problem tour ϕ is optimal for the distance matrix C, if and only if it is optimal for the distance matrix $C' = C + S$. If the length of ϕ for matrix C equals L, then the length of ϕ for matrix C' equals $L + n(n+1)\gamma$. Moreover, by the definition of S the new distance matrix $C' = C + S$ is a monotone Monge matrix. This gives us a monotone Monge matrix for the quadratic assignment problem.

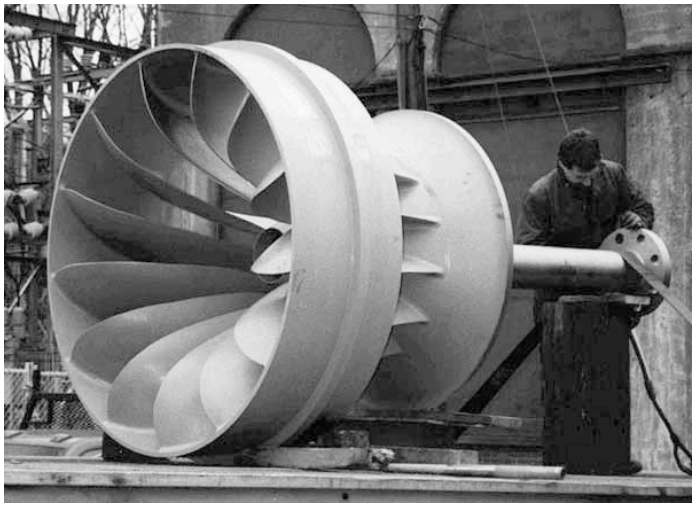
In order to formulate the travelling salesman problem as a quadratic assignment problem (7), we also need a second matrix. This second matrix $H = (h_{i,j})$ describes the structure of a so-called *Hamiltonian* cycle $\langle 1, 2, 3, 4, \ldots, n \rangle$ running through all cities. We set $h_{i,i+1} = 1$ for $i = 1, \ldots, n-1$ and we set $h_{n,1} = 1$. As in our proof of Supnick's result, we double this cycle and traverse it another time backwards. Hence, we set $h_{i+1,i} = 1$ for $i = 1, \ldots, n-1$ and $h_{1,n} = 1$. All remaining entries $h_{i,j}$ are 0. With this it is not hard to see that minimizing the expression in (3) is equivalent to finding

$$\min_{\phi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} c'_{\phi(i),\phi(j)} h_{i,j}. \tag{9}$$

This is a quadratic assignment problem where the first matrix C' is a monotone Monge matrix, and where the second matrix H is a symmetric Toeplitz matrix.

Finally, let us reformulate the maximization version of the turbine balancing problem. That really is simple. In the minimization version of the turbine balancing problem, we want to minimize the expression in (6). Hence in the maximization version, we want to minimize the expression in (6) multiplied by -1. That amounts to computing

$$\min_{\phi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} -m_{\phi(i)} m_{\phi(j)} \cos \frac{2(i-j)\pi}{n}. \tag{10}$$



www.norcanhydro.com

Figure 6 A hydraulic turbine runner

Analogously to the data arrangement problem, we observe that the negative product matrix $(-m_i m_j)$ is a monotone Monge matrix. Moreover, the symmetry $\cos(x) = \cos(-x)$ of the cosine function yields that the second matrix $B = (b_{i,j})$ with $b_{i,j} = \cos(2(i-j)\pi/n)$ is a symmetric Toeplitz matrix.

Where is the common structure?

We have seen that each of the three optimization problems can be written as a quadratic assignment problem with a monotone Monge matrix and a symmetric Toeplitz matrix B. Is this the common property that we are looking for? Is this the common property that automatically makes permutation ϕ^* the universal optimal solution of a quadratic assignment problem? This would be just too good to be true. And actually, we already know that it can not be true: The *minimization* version of the turbine balancing prob-

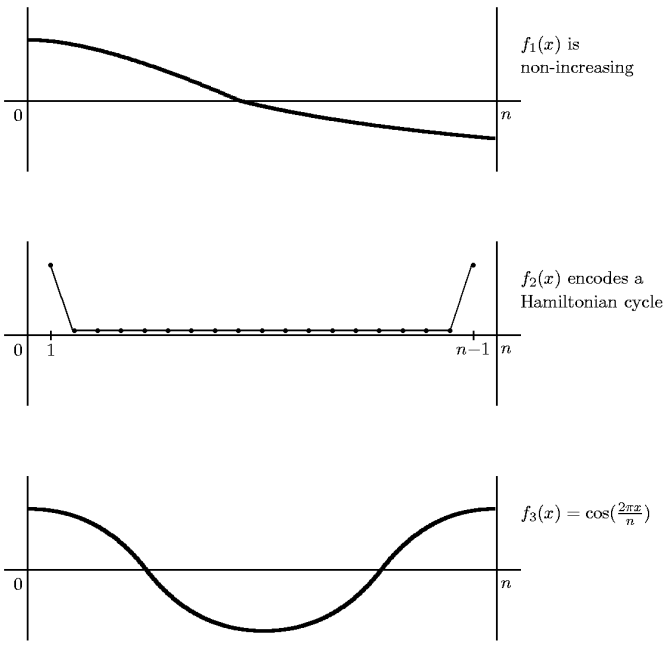


Figure 7 The generating functions for the three symmetric Toeplitz matrices

lem in (6) can be written as the quadratic assignment problem,

$$\min_{\phi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} (-m_{\phi(i)} m_{\phi(j)}) \, (-\cos \frac{2(i-j)\pi}{n}). \qquad (11)$$

Here the first matrix $(-m_i m_j)$ is a monotone Monge matrix, and the second matrix $B = (b_{i,j})$ with $b_{i,j} = -\cos(2(i-j)\pi/n)$ is a symmetric Toeplitz matrix. Now we know from our above discussion of the turbine problem that the minimization version is an NP-hard problem. It definitely can not be solved without computation, and it definitely can not have permutation ϕ^* as a universal optimal solution. So we must search for a stronger common property of the three optimization problems, and the right place to search are the generating functions for the three Toeplitz matrices.

- The generating function f_1 of the Toeplitz matrix in the data arrangement problem (8) is a non-increasing function.
- The generating function f_2 in the travelling salesman problem on Supnick matrices in (9) is given by $f_2(1) = f_2(n-1) = 1$, and $f_1(i) = 0$ otherwise.
- The generating function in the maximization version of the turbine problem in (10) is $f_3(x) = \cos \frac{2x\pi}{n}$.

Examples for the graphs of such generating functions f_1, f_2, f_3 are depicted in Figure 7. At first sight, these graphs do not have much in common. At second sight, the reader will notice that all three functions are non-increasing in the left half of the picture. Moreover, the graphs of f_2 and f_3 are symmetric with respect to the vertical line through $n/2$. The graph of f_1 is also non-increasing in the right half of the picture, but apart from this property it may behave quite chaotically. All three generating functions show the following behavior:

- On the domain $1, \ldots, \lfloor n/2 \rfloor$ the function f is non-increasing.
- On the domain $\lceil n/2 \rceil, \ldots, n-1$ the function f satisfies $f(x) \leq f(n-1-x)$. That is, in the right half of the picture the function graph never goes above the graph of its mirror image from the left half of the picture.

Let us say that a function $f\colon \{0, \ldots, n-1\} \to \mathbf{R}$ is *benevolent* if it satisfies these two crucial properties, and let us say that a symmetric Toeplitz matrix is *benevolent* if it is generated by a benevolent function. See Figure 8 for an example of a benevolent function.

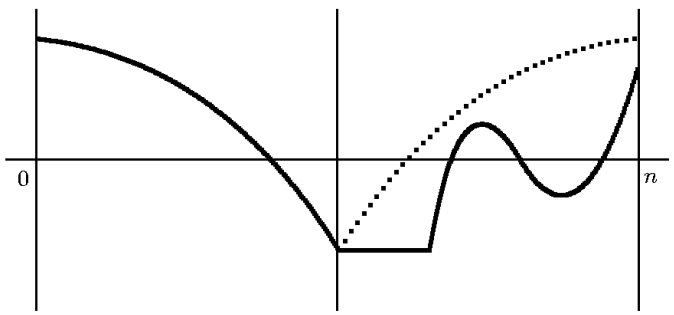


Figure 8 A function f is *benevolent* if on the domain $1, \ldots, \lfloor n/2 \rfloor$ the function f is non-increasing and on the domain $\lceil n/2 \rceil, \ldots, n-1$ the function f satisfies $f(x) \leq f(n-1-x)$. That is, in the right half of the picture the function graph never goes above the graph of its mirror image from the left half of the picture..

Now we have finally found the desired common structure: If matrix A is a monotone Monge matrix, and if matrix B is a benevolent Toeplitz matrix, then the corresponding quadratic assignment problem in (7) always has the permutation ϕ^* as an optimal solution. This result has been formulated and proved by Burkard, Çela, Rote & Woeginger [1998].

A glimpse of the proof

We will now sketch the idea for the proof that the permutation ϕ^* constitutes a universal optimal solution for any quadratic assignment problem of the type described above. The proof technique is essentially the same as in our argument for Supnick's travelling salesman problem result; it works with the extremal rays of the underlying matrix cones.

It is easy to see that the monotone Monge matrices form a cone. A trivial example for a monotone Monge matrix are the constant matrices $L = (\ell_{i,j})$ with $\ell_{i,j} = \lambda$ for all indices i, j. Here λ is an arbitrary real number. Another example for monotone Monge matrices are the *Lower-Right block* matrices $C = (c_{i,j})$. Such an LR block matrix has a rectangular block of 0-entries in the lower right corner, and it has 1-entries everywhere else. Formally, for fixed integers x and y an LR block matrix satisfies $c_{i,j} = 0$ if $i \geq x$ and $j \geq y$, and $c_{i,j} = 1$ otherwise. Rudolf & Woeginger [1995] have shown that every monotone Monge matrix can be written as the sum of a constant matrix and the non-negative linear combination of LR block matrices.

The benevolent Toeplitz matrices also form a cone. Let us consider two representative types of benevolent generating functions $f\colon \{0, \ldots, n-1\} \to \mathbf{R}$:

- The first type depends on a parameter α with $1 \leq \alpha \leq n/2$. $f(x) = 0$ if $\alpha \leq x \leq n - \alpha$ and $f(x) = 1$ otherwise.
- The second type depends on a parameter β with $n/2 \leq \beta \leq n-1$: $f(x) = -1$ if $x = \beta$ and $f(x) = 0$ otherwise.

Generating functions of the first type are symmetric with respect to the middle-axis $x = n/2$, and they are non-increasing in the left half of their domain. Generating functions of the second type are identically zero in the left half of their domain, and they are non-positive in the right half of their domain. If a Toeplitz matrix is generated by such a function of first (respectively, second) type, then we call it a benevolent Toeplitz matric of first (respectively, second) type. It is not difficult to see that every benevolent Toeplitz matrix can be written as the sum of some constant matrix, plus a non-negative linear combination of benevolent Toeplitz matrices of the first type, plus a non-negative linear combination of benevolent Toeplitz matrices of the second type.

The above observations set up the scene for the proof: We have a simple additive characterization of monotone Monge matrices, and we have a simple additive characterization of benevolent Toeplitz matrices. The constant matrices that show up in these characterizations are irrelevant for the quadratic assignment problem in (7). They contribute some fixed value to the objective function, and this fixed value is independent of the permutation ϕ. It remains to deal with LR block matrices on the one side, and with benevolent Toeplitz matrices of first and second type on the other side. Burkard, Çela, Rote & Woeginger [1998] show that the permutation ϕ^* constitutes a universal optimal solution for (i) the quadratic assignment problem with an LR block matrix A and a benevolent Toeplitz matrix B of the first type, and for (ii) the

quadratic assignment problem with an LR block matrix A and a benevolent Toeplitz matrix B of the second type. The exact arguments are simple, but somewhat tedious. Since all entries in the investigated matrices are 0 or 1, the purely combinatorial line of argumentation goes through.

By combining the results in (i) and (ii), we conclude that permutation ϕ^* is a universal optimal solution for any quadratic assignment problem with a monotone Monge matrix and a benevolent Toeplitz matrix.

References

1 K. Anstreicher, N. Brixius, J.P. Goux, and J. Linderoth [2002]. Solving large quadratic assignment problems on computational grids. *Mathematical Programming 91*, 563–588.

2 A.A. Bolotnikov [1978]. On the best balance of the disk with masses on its periphery. *Problemi Mashinostroenia 6*, 68–74, (in Russian).

3 R.E. Burkard, E. Çela, G. Rote, and G.J. Woeginger [1998]. The quadratic assignment problem with a monotone anti-Monge and a symmetric Toeplitz matrix: Easy and hard cases. *Mathematical Programming 82*, 125–158.

4 R.E. Burkard, B. Klinz, and R. Rudolf [1996]. Perspectives of Monge Properties in Optimization. *Discrete Applied Mathematics 70*, 95–161.

5 V.N. Burkov, M.I. Rubinshtein, and V.B. Sokolov [1969]. Some problems in optimal allocation of large-volume memories. *Avtomatika i Telemekhanika 9*, 83–91, (in Russian).

6 E. Çela [1994]. *The quadratic assignment problem*. Kluwer, Dordrecht.

7 E. Çela and G.J. Woeginger [1994]. A note on the maximum of a certain bilinear form. Technical Report SFB-8, TU Graz, Austria.

8 M.R. Garey and D.S. Johnson [1979]. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.

9 W.R. Hamilton [1858]. On a new system of roots of unity. *Proceedings of the Royal Irish Academy 6*, 415–416.

10 G.H. Hardy, J.E. Littlewood, and G. Pólya [1926]. The maximum of a certain bilinear form. *Proceedings of the London Mathematical Society 25*, 265–282.

11 G.H. Hardy, J.E. Littlewood, and G. Pólya [1934]. *Inequalities*, Cambridge University Press, Cambridge.

12 T.C. Koopmans and M.J. Beckmann [1957]. Assignment problems and the location of economic activities. *Econometrica 25*, 53–76.

13 G. Laporte and H. Mercure [1988]. Balancing hydraulic turbine runners: A quadratic assignment problem. *European Journal of Operational Research 35*, 378–382.

14 E.L. Lawler [1963]. The quadratic assignment problem. *Management Science 9*, 586–599.

15 E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys (eds.) [1985] *The travelling salesman problem*. John Wiley, Chichester.

16 N.N. Metelski [1972]. On extremal values of quadratic forms on symmetric groups. *Vesti Akad. Navuk BSSR Ser. Fiz.-Mat. Navuk 6*, 107–110, (in Russian).

17 M. Michalski [1987]. On a class of polynomially solvable travelling salesman problems. *Zastosowania Matematyki 19*, 531–539.

18 G. Monge [1781]. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Science (Année 1781, avec les Mémoires de Mathématique et de Physique, pour la même Année)*, 666–704.

19 J. Mosevich [1986]. Balancing hydraulic turbine runners — a discrete combinatorial optimization problem. *European Journal of Operational Research 26*, 202–204.

20 C.H. Papadimitriou [1994]. *Computational Complexity*. Addison-Wesley.

21 V.R. Pratt [1972]. An $N \log N$ algorithm to distribute N records optimally in a sequential access file. In: *Complexity of Computer Computations*, (R.E. Miller and J.W. Thatcher eds.), Plenum Press, New York, 111–118.

22 M.I. Rubinshtein [1971]. On the symmetric traveling salesman problem. *Automation and Remote Control 32*, 1453–1460.

23 R. Rudolf and G.J. Woeginger [1995]. The cone of Monge matrices: Extremal rays and applications. *ZOR – Mathematical Methods of Operations Research 42*, 161–168.

24 D. Schlegel [1987]. Die Unwucht-optimale Verteilung von Turbinenschaufeln als quadratisches Zuordnungsproblem. Ph. D. Thesis, ETH Zürich.

25 F. Supnick [1957]. Extreme Hamiltonian lines. *Annals of Mathematics 66*, 179–201.

26 Y.G. Stoyan, V.Z. Sokolovskii, and S.V. Yakovlev [1982]. A method for balancing discretely distributed masses under rotation. *Energomashinostroenia 2*, 4–5, (in Russian).

27 B.B. Timofeev and V.A. Litvinov [1969]. On the extremal value of a quadratic form. *Kibernetika 4*, 56–61, (in Russian).

28 R.G. Vickson and X. Lu [1998]. Optimal product and server locations in one-dimensional storage racks. *European Journal of Operational Research 105*, 18–28.