ON THE TURNPIKE PROBLEM

by

Tamara Dakić

B.Sc., University of Zagreb, 1991

M.Sc., University of Zagreb, 1993

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
August 2000

0-612-61635-5

Abstract

The turnpike problem, also known as the *partial digest problem*, is:

> Given a multiset of $\binom{n}{2}$ positive numbers ΔX, does there exist a set X such that ΔX is exactly the multiset of all positive pairwise differences of the elements of X.

The complexity of the problem is not known.

We write the turnpike problem as a $0-1$ quadratic program. In order to solve a quadratic program, we relax it to a semidefinite program, which can be solved in polynomial time. We give three different formulations of the turnpike problem as a $0-1$ quadratic program.

For the first $0-1$ quadratic program we introduce a sequence of semidefinite relaxations, similar to the sequence of semidefinite relaxations proposed by Lovász and Schrijver in their seminal paper "Cones of matrices and set-functions and $0-1$ optimization" (*SIAM Journal on Optimization* 1, pp 166-190, 1990). Although a powerful tool, this method has not been used except in their original paper to develop a polynomial time algorithm for finding stable sets in perfect graphs. We give some theoretical results on these relaxations and show how they can be used to solve the turnpike problem in polynomial time for some classes of instances. These classes include the class of instances constructed by Zhang in his paper "An exponential example for partial digest mapping algorithm" (*Tech Report, Computer Science Dept., Penn State University* 1993) and the class of instances that have a unique solution and all the numbers in ΔX are different and on which Skiena, Smith and Lemke's backtracking procedure, from their paper "Reconstructing sets from interpoint distances" (*Proc. Sixth ACM Symp. Computational Geometry*, pp 332 - 339, 1990) backtracks

only a constant number of steps. Previously it was not known how to solve the former in polynomial time.

We use our theoretical formulations to develop a polynomial time heuristic to solve general instances of the problem.

We perform extensive numerical testing of our methods. To date we do not have an instance of the turnpike problem for which our methods do not yield a solution.

The second $0-1$ quadratic program formulation of the turnpike problem will be too large for practical purposes. We use association schemes and some other methods to reduce its size and obtain the third $0-1$ quadratic program. We establish a connection between this relaxation and the first relaxation and show its limitations.

Contents

List of Figures

Chapter 1

Introduction

In this section we give basic definitions and background results needed in this thesis.

1.1 The Turnpike Problem

1.1.1 Problem definition

The turnpike problem, also known as the *partial digest problem*, is:

> Given a multiset of $\binom{n}{2}$ positive numbers ΔX, does there exist a set X such that ΔX is exactly the multiset of all positive pairwise differences of the elements of X. (P)

If the answer to the above question is positive, we call the multiset ΔX the *difference set* of X and the set X a *solution set.*

If the answer to the above question is negative, we say that for the multiset ΔX there are no solution sets.

The problem first appeared in the 1930s in experiments on X-ray crystallography [24],[25],[26]. According to Skiena, Smith and Lemke [31] it was also posed in 1977 by Shamos as a computational geometry problem [30].

The word *turnpike* refers to a toll road and the problem got its name from the problem of reconstructing the order of cities along the road from their pairwise distances.

Another name for the turnpike problem is the partial digest problem, which arises in molecular biology and in particular in restriction site analysis of DNA, [7]. A DNA molecule can be regarded as a string on the alphabet of nucleotides $\{A, C, G, T\}$, where A represents adenine, C cytosine, G guanine and T thymine. A *restriction enzyme* is a chemical that cuts a DNA molecule at places, called *restriction sites*, determined by certain sequences of nucleotides. The lengths of the cut fragments can be measured. The *restriction site analysis* is the method of using this information to determine where the restriction sites are on the molecule.

A few types of experiments can be performed. A *full* or *complete digest* is an experiment in which for a given DNA molecule and a restriction enzyme the chemical reaction is allowed to complete. If there is more than one restriction site any permutation of the obtained fragments is a possible interpretation of the molecule. To gain additional information about the molecule, more than one complete digest using different restriction enzymes that cut the molecule at different sites can be performed. For example, if two different enzymes are used, the experiment is called a *double digest*.

If there are many identical molecules and the chemical reaction is not allowed to complete, then all fragments between any two restriction sites are obtained. Such an experiment is called a *partial digest*. Therefore, in order to reconstruct the molecule we have to answer the question that is similar to the turnpike problem.

1.1.2 Known facts and algorithms

Two subsets X, Y of the set of real numbers $\mathbb{R}$ are said to be *congruent* if $X = Y - a = \{y - a | y \in Y\}$, for some $a \in \mathbb{R}$, or $X = -Y + a = \{-y + a | y \in Y\}$, for some $a \in \mathbb{R}$.

It is easy to see that if two sets X and Y are congruent, the multisets ΔX and ΔY are identical. Therefore, given a multiset ΔX, we can assume that both 0 and the largest element of ΔX are in the solution set X. Henceforth we always assume that $0 \in X$.

Two noncongruent sets X and Y are *homeometric* if the multisets ΔX and ΔY are identical.

For a given multiset $\{a_1, a_2, \ldots, a_l\}$ we define its generating function by

$$f(x) = \sum_{i=1}^{l} x^{a_i}.$$

Now, if $Q(x)$ is the generating function for a multiset $\Delta X \cup (-\Delta X)$ and $P(x)$ the generating function for a solution set X, and if n is the number of points in X, then

$$Q(x) + n = P(x)P(\frac{1}{x}).$$

In [28] Rosenblatt and Seymour work over rings of the form

$$K[\mathbb{R}^n] = \sum\{a_v x^v : a_v \in K, v \in \mathbb{R}^n\},$$

where K is either $\mathbb{Z}$, $\mathbb{R}$ or $\mathbb{C}$, and use factoring to reconstruct X from ΔX. However, their main results are not algorithmic, but rather give necessary and sufficient conditions for two sets to have the same difference set.

In case the multiset ΔX contains only integers, the polynomial $Q(x) + n$ can be factored over the ring of polynomials with integer coefficients in time polynomial in the largest exponent [17]. By combining this fact with the theoretical results from [28] Lemke and Werman obtain a reconstruction algorithm that runs in time polynomial in the largest difference in the multiset ΔX, [16].

In [31] Skiena, Smith and Lemke propose a backtracking algorithm to solve the turnpike problem. To visualize their algorithm, we observe that if a given multiset ΔX is a difference set of $X = \{0 < x_1 < \cdots < x_{n-1}\}$, the elements of ΔX can be organized in a pyramid that on one side has the elements of $X - \{0\}$, and on the other side elements of the set $(x_{n-1} - X) - \{0\}$. For example, if

$$\Delta X = \{1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 17, 18\}$$

and

$$X = \{0, 4, 10, 15, 17, 18\},$$

the pyramid is

				18				
			17		14			
		15		13		8		
	10		11		7		3	
4		6		5		2		1

In the bottom row of the pyramid we put the differences of two consecutive elements of X, i.e. the differences of the form $x_{i+1} - x_i$, for $i = 0, \dots, n-1$. The second row from the bottom contains differences of the form $x_{i+2} - x_i$, for $i = 0, \dots, n-2$, and in general, the k-th row from the bottom contains the differences of the elements of X of the form $x_{i+k} - x_i$, for $i = 0, \dots, n-k-1$.

Notice also that that the numbers decrease going down along any diagonal parallel to the sides of the pyramid.

The backtracking procedure of Skiena, Smith and Lemke positions the numbers of ΔX in the pyramid. Suppose that we positioned l numbers on the top left side of the pyramid and k numbers on the top right side of the pyramid. Then all the numbers that are in the shaded region of the pyramid in Figure 1.1 are also determined.

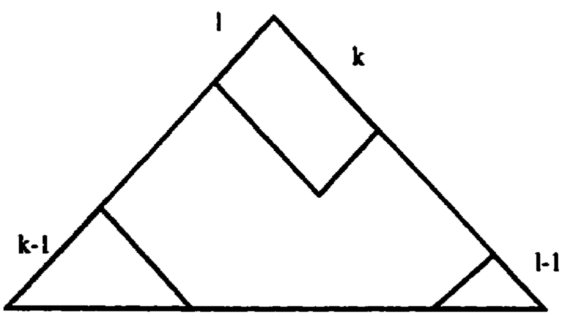

Figure 1.1: The shape of the pyramid at each step of execution of Skiena's et al. backtracking procedure.

Because of the above observations, for the largest remaining unpositioned distance, there are only two possible locations: either the topmost unfilled space on the left side of the pyramid, or the topmost unfilled space on the right side of the pyramid.

Therefore, the procedure always positions the largest remaining distance on the topmost unfilled space on the left side of the pyramid and tries to fill in all the numbers in the regions that have the shape of the shaded regions shown in Figure 1.1. If this is not possible, the procedure backtracks. The backtracking step consists of putting the largest remaining distance on the topmost unfilled space on the right side of the pyramid, and if this leads to an inconsistency, the procedure backtracks one level up.

The pseudocode for the backtracking procedure as given by Zhang [35] is shown in Figure 1.2.

Skiena, Smith and Lemke [31] proved that in general this procedure runs in $O(2^n n \log n)$-time, although instances for which it takes more than $O(n^2 \log n)$-time are rare. A class of examples for which this algorithm takes exponential time is given by Zhang [35].

Skiena and Sundaram [32] adapt this backtracking algorithm to work with data that contains experimental errors. The experimental errors they consider are "noisy" interpoint distances and missing fragments lengths.

Finally, we mention some results on the number of homeometric sets. Let $H(n)$ be the maximum possible number of mutually noncongruent and homeometric sets on n elements. In [31] Skiena, Smith and Lemke prove that

$$\frac{1}{2}n^{0.8107144} \leq H(n) \leq \frac{1}{2}n^{1.2334827},$$

where the lower bound inequality holds for an infinite number of values of n and the upper bound inequality holds for all values of n.

We say that an instance ΔX of the turnpike problem has k solutions, if k is the number of homeometric sets that have ΔX as their difference set. It can also be shown that for a given multiset ΔX, the number of solution sets is 0 or a power of 2, [31].

It is important to say that for almost all instances ΔX, the solution set X, if it exists, is unique, and therefore partial digest is a good method for restriction site analysis. However, the importance of the method is diminishing with the reduction in cost of DNA sequencing.

```
set X
int width

procedure PartialDigest( List L)
        width=DeleteMax(L);
        X = {0, width};
        Place(L);
end

procedure Place(List L)
        if L = ∅ then
             output solution X;
             exit;
        endif
        y=DeleteMax(L);
        if Δ({y} ∪ X) ⊆ L then
             X = X ∪ {y};
             Place(L − Δ({y} ∪ X)); // place on the left
             X = X − {y};
        endif
        if Δ({width − y} ∪ X) ⊆ L then
             X = X ∪ {width − y};
             Place(L − Δ({width − y} ∪ X)); // place on the right
             X = X − {width − y};
        endif
end
```

Figure 1.2: Pseudocode for Skiena's et al. backtracking procedure.

It is not known if the turnpike problem is solvable in time polynomial in the number of elements in the given multiset ΔX.

1.2 Semidefinite programming

Semidefinite programming is a special case of convex programming and a special case of linear programming over cones or cone-LP. A combinatorial optimization problem was first written in the form of a semidefinite program in the work by Lovász [18] on the Shannon capacity of a graph.

In the seminal paper [19] Lovász and Schrijver show how to use semidefinite programming to find maximum stable sets in perfect graphs.

Lately, a great deal of interest in the application of semidefinite programming in combinatorial optimization has arisen due to the paper by Goemans and Williamson [10] in which they give a 0.878-approximation algorithm for MAX CUT and MAX 2SAT and a 0.7554-approximation algorithm for MAXSAT.

Semidefinite programs can be solved within an error $\epsilon > 0$ in polynomial time using the ellipsoid algorithm, standard polynomial time algorithms for convex programming or interior point methods.

1.2.1 Definition and Basic Facts

A convex optimization problem in which the feasible region consists of real symmetric positive-semidefinite matrices X whose entries satisfy linear constraints, and the objective function is a linear function of the entries of X, is called a *semidefinite program.* A semidefinite program can be written in the following way:

$$\begin{aligned} \text{Min } & C \bullet X \\ & A_i \bullet X = b_i, \quad \text{for } i = 1, \dots, m \\ & X \geq 0 \end{aligned} \tag{SDP}$$

where C, A_i, $i = 1, \dots, m$ and X are $n \times n$ matrices, $X \geq 0$ indicates that X is a positive semidefinite matrix and for two matrices U and V, $U \bullet V$ denotes their Hadamard product, i.e. $U \bullet V = \sum_{i,j} U_{ij} V_{ij}$.

It is convenient to assume that the matrices C and A_i in (SDP) are symmetric. If, for example, C is not symmetric, we can replace C by $\frac{1}{2}(C+C^T)$ since $C^T \bullet X = C \bullet X$ for symmetric matrices X.

Now we can define the dual of the problem (SDP):

$$\begin{aligned} &\text{Max } \mathbf{b}^T\mathbf{y} \\ &C - \sum_{i=1}^{m} y_i A_i \geq 0. \end{aligned} \tag{Dual}$$

The duality theory for semidefinite programming can be viewed as a special case of the cone duality for the general convex programs. There are many similarities between the duality theory for semidefinite programming and the duality theory for linear programming. Under some additional assumptions, one can prove a version of Farkas' Lemma, the strong duality theorem and the complementary slackness theorem for semidefinite programs.

1.2.2 Methods for Solving SDP

Various methods for solving linear programs and general convex programs can be applied to solve semidefinite programs. For example the ellipsoid method and the interior point methods can be used to solve a semidefinite program within an additive constant $\epsilon > 0$. Grötschel, Lovász and Schrijver [12] proved that there exists a polynomial time algorithm for solving a positive semidefinite program. They obtained this result as a direct consequence of the general results on applications of the ellipsoid algorithm to convex programming.

In practice, the ellipsoid method is slow. In [1] Alizadeh has adapted the interior point method of Ye [34] to semidefinite programming. He also claims that many other interior point methods for linear programming can be extended to polynomial time methods for semidefinite programming in the same way.

1.3 Applications to Combinatorial Optimization

So far semidefinite programming has been used in combinatorial optimization to prove that certain optimization problems can be solved in polynomial time (see [12], [13], [19]) and to obtain better approximation algorithms for NP-hard problems (see [8], [9], [10], [18], [20], [22]).

The paper by Lovász [18] was a pioneering work in the application of positive-semidefinite programming to combinatorial optimization. It described the ideas which were later generalized and used to solve other problems. In this paper Lovász defined the now famous Lovász number $\nu(G)$ of a graph G. He uses $\nu(G)$ to bound the Shannon Capacity of G. Later he proved that $\nu(G)$ is actually sandwiched between the chromatic number of the complement of G and independence number of G, i.e. $\alpha(G) \leq \nu(G) \leq \chi(\overline{G})$.

Solutions of many problems in combinatorial optimization can be written as $0-1$ vectors, which are characteristic vectors of appropriate sets. The convex hull C of those vectors can be described as the set of solutions of a system of linear inequalities. The problem is that the convex hull C might have exponentially many facets and can only be described by a linear system of exponential size. So, research has been centered on trying to find an approximation to C, i.e. a convex set that would be bigger than C but over which we can optimize in polynomial time. One way, for example, would be to take a polynomial sized subset of the set of linear inequalities that describe C. In [19] Lovász and Schrijver give a general technique to construct higher dimensional polyhedra (or more generally, convex sets) whose projections approximate the convex hull C of $0-1$ vectors and over which we can optimize in polynomial time. Here we sketch their construction.

First, we need a couple definitions:

In order to have a homogeneous system of inequalities, the n-dimensional space is embedded in $\mathbb{R}^{n+1}$ as the hyperplane $x_0 = 1$.

For a convex cone K in $\mathbb{R}^{n+1}$ let K^0 denote the cone spanned by all $0-1$ vectors in K.

Let K^* be the polar cone of a cone K, i.e.

$$K^* = \{u \in \mathbb{R}^{n+1} : u^T x \geq 0 \text{ for all } x \in K\}.$$

Let Q denote the cone spanned by all $0-1$ vectors $x \in \mathbb{R}^{n+1}$ with $x_0 = 1$.

Furthermore, let $\mathbf{e}_i$ denote the i-th unit vector in $\mathbb{R}^{n+1}$ and let $\mathbf{f}_i = \mathbf{e}_i - \mathbf{e}_0$. It is easy to see that the dual cone Q^* is spanned by the set of vectors $\mathbf{e}_i$ and $\mathbf{f}_i$, $i = 0, \ldots, n$.

For any matrix Y, we denote the vector composed of diagonal entries of Y by $\text{diag}(Y)$.

For a convex cone $K \subseteq \mathbb{R}^{n+1}$, the higher dimensional cone whose projection would approximate K^0 consists of the symmetric $(n+1) \times (n+1)$ matrices Y that include xx^t, where $x \in K^0$ and $x_0 = 1$. The diagonal of Y is an element of the cone K. The idea is that the constraints on the elements of matrices Y can induce cuts of the cone K, so it approximates the cone K^0 better. This motivates the definitions of the cones $M(K_1, K_2)$ and $M_+(K_1, K_2)$ below. The reason why two cones K_1 and K_2 are considered is technical. Only two special cases are considered: $K_1 = K_2 = K$ and $K_1 = K$ and $K_2 = Q$.

Now we define the cones $M(K_1, K_2)$ and $M_+(K_1, K_2)$.

Let $K_1, K_2 \subseteq Q$ be convex cones. Let $M(K_1, K_2) \subseteq \mathbb{R}^{(n+1)\times(n+1)}$ be the cone which consists of all matrices Y which satisfy the following conditions:

(i) Y is symmetric;

(ii) $\text{diag}(Y) = Y\mathbf{e}_0$, i.e. $y_{ii} = y_{0i}$ for all $1 \leq i \leq n$;

(iii) $\mathbf{u}^T Y \mathbf{v} \geq 0$ holds for every $u \in K_1^*$ and $v \in K_2^*$.

Note that (iii) can be rewritten as

(iii') $YK_2^* \subseteq K_1$.

We also consider the cone $M_+(K_1, K_2)$ which consists of matrices Y which satisfy (i)-(iii) and the additional constraint:

(iv) Y is positive semidefinite.

Since K_1 and K_2 are contained in Q, Q^* is contained in K_1^* and K_2^*, and every matrix Y contained in $M(K_1, K_2)$, and therefore in $M_+(K_1, K_2)$, satisfies

$$\begin{aligned} &y_{ij} \geq 0, \\ &y_{ij} \leq y_{ii} = y_{0i} \leq y_{00}, \text{ and} \\ &y_{ij} \geq y_{ii} + y_{jj} - y_{00}. \end{aligned} \tag{1.1}$$

We can project cones $M(K_1, K_2)$ and $M_+(K_1, K_2)$ to $(n+1)$-dimensional space by defining cones

$$N(K_1, K_2) = \{Ye_0 : Y \in M(K_1, K_2)\} = \{\text{diag}(Y) : Y \in M(K_1, K_2)\}$$

and

$$N_+(K_1, K_2) = \{Ye_0 : Y \in M_+(K_1, K_2)\} = \{\text{diag}(Y) : Y \in M_+(K_1, K_2)\}.$$

It is easy to see that if K_1 and K_2 are polyhedral cones, then $M(K_1, K_2)$ and $N(K_1, K_2)$ are polyhedral cones as well, [19]. The cones $M_+(K_1, K_2)$ and $N_+(K_1, K_2)$ are also convex but generally not polyhedral.

Note that if $\mathbf{x}$ is a $0-1$ vector in $K_1 \cap K_2$ then the matrix $\mathbf{x}\mathbf{x}^{\mathbf{T}}$ satisfies conditions (i)-(iv). Moreover, the following lemma is proved in [19].

Lemma 1.1 $(K_1 \cap K_2)^0 \subseteq N_+(K_1, K_2) \subseteq N(K_1, K_2) \subseteq K_1 \cap K_2$.

In general, $N(K_1, K_2)$ is much smaller than $K_1 \cap K_2$. We only consider two special cases $K_1 = K_2 = K$ and $K_1 = K$, $K_2 = Q$. Although, $N(K, K) \subseteq N(K, Q)$ we consider $N(K, Q)$ because it behaves better algorithmically. Because of (iii') we can notice that a matrix $Y \in M(K, Q)$ has the property

(iii") Every column of Y is in K and the difference of the first column and any other columns is in K.

To abbreviate the notation we write $N(K) = N(K, Q)$, $M(K) = M(K, Q)$, $N_+(K) = N_+(K, Q)$ and $M_+(K) = M_+(K, Q)$.

An element of the cone $N(K)$, and therefore also $N_+(K)$, can be represented as a sum of two elements of cone K that on a position i either have 0 or an entry that is equal to the entry on the position 0. More precisely, if $H_i = \{x \in \mathbb{R}^{n+1} | x_i = 0\}$ and $G_i = \{x \in \mathbb{R}^{n+1} | x_i = x_0\}$, Lovász and Schrijver [19] prove the following lemma

Lemma 1.2 *For every convex cone $K \subseteq Q$ and every $i \in \{1, \dots, n\}$,*

$$N(K) \subseteq (K \cap H_i) + (K \cap G_i).$$

We can get better approximations of the cone K^0 by iterating the operator N and N_+, i.e. we can define $N^0(K) = K$, $N^t(K) = N(N^{t-1}(K))$ and similarly $N^0_+(K) = K$, $N^t_+(K) = N_+(N^{t-1}_+(K))$. Lovász and Schrijver [19] prove the following theorem.

Theorem 1.3 $N^n(K) = K^0$.

The importance of the above theory lies in the fact that we can optimize linear functions over $N(K)$ and $N_+(K)$ in polynomial time. The following theorem was also proved in [19]:

Theorem 1.4 *Suppose that we have a weak separation oracle for K. Then the weak separation problem for $N(K)$ and $N_+(K)$ can be solved in polynomial time.*

1.3.1 Stable Sets in Graphs

Lovász and Schrijver [19] apply their results to obtain polynomial time algorithms for finding maximum stable sets in certain classes of graphs.

A *stable set* in a graph $G = (V, E)$ is a subset of the set of vertices V, such that no two of them are adjacent.

A *maximum stable set* in a graph G is a stable set whose cardinality is maximal over all stable sets of the graph.

The problem of finding a maximum stable set in a general graph is NP-hard.

In order to apply the results of the previous section, we have to define the following convex sets.

Let $G = (V, E)$ be a graph with vertices $V = \{1, 2, \dots, n\}$. For each $X \subseteq V$, let $\chi^X \in \{0, 1\}^V$ denote its characteristic vector, i.e. the vector that for every vertex

$i \in \{1, \dots, n\}$ of G has 1 as its i-th coordinate if $i \in X$, and 0 otherwise. The *stable set polytope of* G is defined as

$$\text{STAB}(G) = \text{conv}\{\chi^X | X \text{ is a stable set for G}\},$$

i.e. the convex hull of characteristic vectors of all stable sets of G.

Let

$$\text{ST}(G) = \{(1, x) | x \in \text{STAB}(G)\}.$$

i.e. the set of vectors obtained by adding prefix 1 to each vector of STAB(G).

Also, define the cone FR(G) $\subseteq \mathbb{R}^{n+1}$, such that for any vector $(x_0, x_1, \dots, x_n) \in$ FR(G) the nonnegativity constraints

$$x_i \geq 0 \text{ for every } 0 \leq i \leq n$$

and edge constraints

$$x_i + x_j \leq x_0 \text{ for each edge } ij \text{ of } G$$

are valid.

For any matrix $Y = (y_{ij}) \in M(\text{FR}(G))$, the following is valid:

For any edge ij of G, $y_{ij} = 0$ because of (iii'). The constraint $x_i + x_j \leq x_0$ must be satisfied by Ye_i, and therefore $y_{ii} + y_{ji} \leq y_{0i} = y_{ii}$, which implies $y_{ij} = 0$.

Also, Ye_k must satisfy the same inequality $x_i + x_j \leq x_0$, and therefore

$$y_{ik} + y_{jk} \leq y_{kk}.$$

Moreover, $Ye_0 - Ye_k$ must satisfy the same inequality, so

$$(y_{ii} - y_{ik}) + (y_{jj} - y_{jk}) \leq y_{00} - y_{kk},$$

i.e.

$$y_{ik} + y_{jk} \geq y_{ii} + y_{jj} + y_{kk} - y_{00}.$$

Note that the intersection of the cone FR^0(G) with the hyperplane $x_0 = 1$ is equal to ST(G). The cone FR(G) is described by the number of inequalities that is polynomial in n, the number of vertices in G. We can therefore optimize any linear function over $\mathrm{FR}^0(G)$ in polynomial time. Unfortunately, $\mathrm{FR}(G) \cap H_0 = ST(G)$, where H_0 is the hyperplane $x_0 = 1$, holds only for bipartite graphs, [13].

Using the previous section, we look at the cones $N^i(\mathrm{FR}(G))$ and $N^i_+(\mathrm{FR}(G))$ for $0 \leq i \leq n$. Lovász and Schrijver [19] prove the following theorem:

Theorem 1.5 *The maximum stable set problem is polynomial time solvable for graphs G for which there exist a constant c such that* $\mathrm{ST}(G) = N^c_+(\mathrm{FR}(G))$.

For perfect graphs we have $\mathrm{ST}(G) = N_+(\mathrm{FR}(G)) \cap H_0$, which enables us to construct the only known algorithm for finding maximum stable set in these graphs.

In fact, Lovász and Schrijver [19] prove that for perfect graphs $\mathrm{ST}(G)$ is determined by diagonal elements of the matrices that satisfy only a subset of constraints in the definition of $M_+(\mathrm{FR}(G))$. For a graph G we define the cone M_{TH} that consist of $(V \cup \{0\}) \times (V \cup \{0\})$ matrices that satisfy the following constraints:

1. Y is a symmetric positive semidefinite matrix;

2. $y_{ii} = y_{0i}$, for every $i \in V$;

3. $y_{ij} = 0$, for every edge $ij \in E(G)$.

Now we can prove the following lemma:

Lemma 1.6 *For a perfect graph G,*

$$\mathrm{ST}(G) = \{Ye_0 | Y \in M_{TH}, (Y)_{00} = 1\}.$$

No class of graphs for which $\mathrm{ST}(G) = N^c_+(\mathrm{FR}(G))$, for some constant $c > 0$ is known.

1.3.2 Quadratic Programs

In this subsection, we show how to relax a general $0-1$ quadratic program to a semidefinite program. We follow the exposition of Helmberg et al. [14], although similar results can be found in earlier papers by Balas et al. [5]. The results of this subsection follow very closely the results from the previous subsection, but are more general. An integer $0-1$ program can be written as a quadratic $0-1$ program because for the $0-1$ variables x_i of an integer program we have that $x_i^2 = x_i$.

A quadratic $0-1$ program, is an optimization problem defined in the following way:

$$\begin{aligned} \text{Max } & x^t C x \\ & x^t A_i x \leq b_i, \text{ for } i = 1, \dots, k \\ & x \in \{0, 1\}, \end{aligned} \tag{QP}$$

where x is an n-dimensional vector, C and A_i, $i = 1, \dots, k$, are real symmetric $n \times n$ matrices, and b_i, $i = 1, \dots, k$ are real numbers. Note that, since the entries x_i of the vector x are either 0 or 1, i.e. $x_i^2 = x_i$, the linear constraints on the entries of x can be written using a diagonal matrix A.

Solving a quadratic $0-1$ program is NP-hard. One way of relaxing a general quadratic $0-1$ program is to write it as a semidefinite program. The key idea is to use a $n \times n$ symmetric matrix Y to represent the pairwise product of entries of the vector x, so that

$$\begin{aligned} y_{ij} &= x_i x_j, \text{ for } i, j \in \{1, \dots, n\} \\ y_{ii} &= x_i^2 = x_i \text{ for } i \in \{1, \dots, n\}. \end{aligned}$$

Inequalities similar to the inequalities (1.1) can be obtained by exploiting the $0-1$ properties of the variables x_i. We have

$$\begin{aligned} & x_i x_j \geq 0, \\ & x_i(1 - x_j) \geq 0, \\ & (1 - x_i)(1 - x_j) \geq 0, \end{aligned} \tag{1.2}$$

and therefore

$$\begin{aligned} & y_{ij} \geq 0, \\ & y_{ii} \leq y_{ij}, \text{ and} \\ & y_{ij} \geq y_{ii} + y_{jj} - 1. \end{aligned} \tag{1.3}$$

The matrix Y is of the form xx^T and is therefore positive semidefinite. Also $\text{diag}(Y) = x$. Furthermore, the matrix

$$(x+v)(x+v)^T = xx^T + xv^T + vx^T + vv^T$$

is positive semidefinite for any vector $v \in \mathbb{R}^n$. Hence, Y can be constrained to satisfy

$$Y + \text{diag}(Y)v^T + v\text{diag}(Y)^T + vv^T \succeq 0, \tag{1.4}$$

for any vector $v \in \mathbb{R}^n$. The condition (1.4) can be rewritten as

$$Y + (\text{diag}(Y) + v)(\text{diag}(Y) + v)^T - \text{diag}(Y)\text{diag}(Y)^T \succeq 0. \tag{1.5}$$

The above constraint is in particular valid when $v = -\text{diag}(Y)$, so the intersection over all vectors $v \in \mathbb{R}^n$ of the constraints (1.5) is characterized by

$$Y - \text{diag}(Y)\text{diag}(Y)^T \succeq 0. \tag{1.6}$$

The constraint (1.6) is not a linear constraint on the entries of Y, but it can be rewritten using the Schur complement as

$$\begin{bmatrix} 1 & \text{diag}(Y)^T \\ \text{diag}(Y) & Y \end{bmatrix} \succeq 0.$$

Now we can relax (QP) to a semidefinite program. For a $n \times n$ matrix U, let U' denote the $(n+1) \times (n+1)$ matrix, indexed by $0, 1, \ldots, n$, whose entries of the 0-th row and column are equal to 0, and

$$U'_{ij} = U_{ij}, \text{ for } i, j \in \{1, \ldots, n\}.$$

The relaxation of (QP) as a semidefinite program is given by

$$\begin{aligned} &\text{Max } C' \bullet Y \\ &\quad A'_i \bullet Y \leq b_i, \text{ for } i = 1, \ldots, k \\ &\quad y_{00} = 1 \\ &\quad y_{0i} = y_{ii} \text{ for } i \in \{1, \ldots, n\} \\ &\quad y_{ij} \geq 0 \text{ for } i, j \in \{1, \ldots, n\} \\ &\quad y_{ii} \geq y_{ij} \text{ for } i, j \in \{1, \ldots, n\} \\ &\quad y_{00} + y_{ij} \geq y_{ii} + y_{jj} \text{ for } i, j \in \{1, \ldots, n\}. \end{aligned} \tag{SDP}$$

Note that for any entry y_{ij} of Y, from the positive semidefinitness of Y, we have that $y_{ij} \leq 1$. To see that, we can first look at the submatrix of Y indexed by the 0th row and any other row. Because of the constraints $y_{0i} = y_{ii}$, this matrix has the form

$$\begin{bmatrix} 1 & y_{ii} \\ y_{ii} & y_{ii} \end{bmatrix},$$

from which directly follows that $y_{ii} \leq 1$, for $i \in \{1, \dots, n\}$.

Also any submatrix indexed by some $i, j \in \{1, \dots, n\}$ has the form

$$\begin{bmatrix} y_{ii} & y_{ij} \\ y_{ij} & y_{jj} \end{bmatrix},$$

and since it is positive semidefinite $y_{ij}^2 \leq y_{ii}y_{jj} \leq 1$.

There are other constraints that a $0-1$ matrix feasible for (SDP) satisfies and that can be added into the definition of (SDP). For example we have the triangle inequalities:

$$y_{ik} + y_{jk} \leq y_{kk} + y_{ij}$$

and

$$y_{ik} + y_{jk} + y_{ij} \geq y_{ii} + y_{jj} + y_{kk} - y_{00},$$

for $i, j, k \in \{1, \dots, n\}$.

These inequalities are sometimes used to improve the approximation, although they contribute substantially to the computing time.

1.4 Thesis Overview

In this section we give a brief overview of the thesis.

The main theoretical results of the thesis are given in Chapter 2. In that chapter we write the turnpike problem as a $0-1$ quadratic program.

For the $0-1$ quadratic program we introduce a sequence of semidefinite relaxations, similar to the sequence of semidefinite relaxations proposed by Lovász and Schrijver [19].

We show that there exists a polynomial time algorithm for solving the turnpike problem on classes of instances for which there exist a constant c, such that the instances are solved by the c-th semidefinite relaxation in the sequence.

In Chapter 3 we give classes of instances that are solved by the first relaxation in the sequence, (S_1). In fact most of the instances are solved by a relaxation that is weaker then (S_1).

We show that if for a given set X, the instance ΔX can be solved by the relaxation (S_1), than the instance ΔY, where ΔY is the difference set of

$$Y = X \cup (X + a) \cup \ldots \cup (X + (m-1)a),$$

and a is greater than the maximum element of X, can also be solved by the relaxation (S_1).

Also, if the instance ΔX can be solved by the relaxation (S_1) and has the property that every solution contains a point that is not in any other solution, the instance ΔY, where ΔY is the difference set of

$$Y = X \cup (X + a_1) \cup \ldots \cup (X + a_k),$$

where

$$a_1 \geq 3d_M + 1$$
$$a_i \geq 3a_{i-1} + d_M + 1 \text{ for } i \in \{2, \ldots, k\},$$

can be solved by the relaxation (S_1). Here d_M denotes the largest element of ΔX.

We also prove that the relaxation (S_1) solves the instances constructed by Zhang, [35]. Previously it was not known how to solve these instances in polynomial time.

In Chapter 3 we also consider the instances ΔX that have unique solutions and all the differences in ΔX are different. We show that if during the execution of the Skiena's et al. backtracking procedure, k is the biggest number of steps that the procedure has to backtrack, then the $(k+1)$-st relaxation in the sequence described in Chapter 2 solves the instance ΔX. That means that if for a class of instances k is a constant, the relaxation (S_{k+1}) has polynomial size and the turnpike problem can be solved in polynomial time for the instances of the class.

In Chapter 4 we show how to develop heuristics for solving the turnpike problem, based on the theoretical results of Chapter 2.

In the first section we describe a heuristic that is based on the relaxation (S_1). It also uses cuts from the second relaxation in the sequence from Chapter 2 and a rounding technique.

In the second section we show how the relaxation (S_1) can be used to reduce the number of backtracking steps of the backtracking procedure of Skiena et al.

In Chapter 5 we enumerate the instances of the turnpike problem for which their relaxations (S_1) were implemented. The computational results show that most of the examined instances are solved by their relaxation (S_1), and the ones that are not have a feasible point of the form

$$Y = \sum_{i=1}^{k} \lambda_i z_i z_i^T, \tag{1.7}$$

where $\lambda_i \geq 0$ and z_i are $0-1$ vectors for $i \in \{1, \ldots, k\}$, but not necessarily characteristic vectors of the solutions of ΔX.

In particular we give some instances that are not solvable by their relaxation (S_1) and show how to use them to construct classes of instances that are not solvable by the relaxation (S_1).

We do not have an instance of the turnpike problem which is not solved by the second relaxation (S_2) of the sequence described in Chapter 2.

In Chapter 6 we present two relaxations of the turnpike problem proposed by A. Schrijver [29]. These relaxations are interesting from the theoretical point of view.

First, the turnpike problem is formulated as a $0-1$ quadratic program, whose semidefinite relaxation is too large for practical purposes. We use association schemes and some other methods, to reduce the size of the $0-1$ quadratic program to obtain a semidefinite relaxation which is smaller and practically possible to solve using today's computers.

Finally, we present an instance ΔX such that ΔX is not a difference set, but its relaxation is feasible.

Chapter 2

Theoretical Results

2.1 Introduction

In this chapter we write the turnpike problem as a $0-1$ quadratic program. The backtracking algorithm described in Chapter 1 takes into account only a certain number of differences at any given time during the execution, whereas a quadratic program treats all the differences simultaneously, which is naturally more powerful.

For the $0-1$ quadratic program we introduce a sequence of semidefinite relaxations, similar to the sequence of semidefinite relaxations proposed by Lovász and Schrijver [19]. Although a powerful tool, this method has not been used except in their original paper to develop a polynomial time algorithm for finding stable sets in perfect graphs as outlined in Chapter 1. Here we give some theoretical results on these relaxations.

2.2 Main results

Throughout this chapter we assume that ΔX contains $\binom{n}{2}$ elements, and that

$$\Delta X' = \{d_1 < d_2 < \cdots < d_M\} \cup \{d_0\},$$

where $d_0 = 0$, is the set that consists of all different elements of the multiset ΔX and 0. Also, for $i > 0$, $v(d_i)$ denotes the multiplicity of d_i in ΔX, i.e. the number of times the number d_i appears in ΔX.

We only consider solution sets X containing 0, so we have that $X \subseteq \Delta X'$. Then we can assign a $0-1$ variable x_{d_i} to each element d_i of $\Delta X'$. For a fixed solution set X, $x_{d_i} = 1$ if and only if d_i is in X. Note that $x_0 = 1$, because we assume that $0 \in X$.

Consider the set $Q \subset \{0,1\}^{M+1}$, determined by the following system:

$$\begin{aligned} \sum_{\substack{i,j=0,\dots,M \\ d_j - d_i = d_k}} x_{d_i} x_{d_j} &= v(d_k), \text{ for } k \in \{1,\dots,M\}, \\ \sum_{i=0,\dots,M} x_{d_i} &= n, \\ x_{d_i} \in \{0,1\} &\text{ for } i \in \{0,\dots,M\}. \end{aligned} \tag{Q}$$

Then the problem (P) is equivalent to the non-emptiness of the set Q and we have:

Proposition 2.1 *A multiset ΔX is a difference set if and only if the set Q is non-empty.*

Proof: If the given ΔX is the difference set of a set X, we can set the variables x_{d_i} to 1 for all $d_i \in X$, and to 0 for all $d_i \notin X$. If there are $\binom{n}{2}$ elements in ΔX, there are n elements in X and therefore

$$\sum_{i=0,\dots,M} x_{d_i} = n.$$

For each difference $d_k \in \Delta X'$, $k > 0$, there exist exactly $v(d_k)$ pairs (d_i, d_j) in $X \times X$, such that $d_j - d_i = d_k$, and therefore also $x_{d_i} x_{d_j} = 1$, and therefore

$$\sum_{\substack{i,j=0,\dots,M \\ d_j - d_i = d_k}} x_{d_i} x_{d_j} = v(d_k).$$

Conversely, if Q is non-empty, for a point $(x_{d_0}, x_{d_1}, \dots, x_{d_M}) \in Q$ we can set

$$X = \{d_i : x_{d_i} = 1\}.$$

Then the set X contains n elements. For every $d_k \in \Delta X'$, $k > 0$, in the equation

$$\sum_{\substack{i,j=0,\dots,M \\ d_j - d_i = d_k}} x_{d_i} x_{d_j} = v(d_k).$$

there are exactly $v(d_k)$ summands $x_{d_i} x_{d_j}$ equal to 1, which means that d_i and d_j are in X and that there are exactly $v(d_k)$ pairs (d_i, d_j) in $X \times X$, such that $d_j - d_i = d_k$.

■

Problem (Q) describes a feasible region of a quadratic $0-1$ program. We can not test for feasibility of a quadratic $0-1$ program in polynomial time (unless "P=NP"). Our approach is to relax (Q) to a program which we can test for feasibility in polynomial time.

One way to relax (Q) is to assume that each variable x_{d_i} is a vector, as described in Chapter 1. We get a feasible region of a semidefinite program by introducing new variables x_{d_i,d_j} for the dot product $x_{d_i} x_{d_j}$ of two vector variables x_{d_i} and x_{d_j}. The variables x_{d_i,d_j} can be organized in a $(M+1) \times (M+1)$ symmetric matrix X_1 whose rows and columns are indexed by the elements of $\Delta X'$, i.e. $0, d_1, \dots, d_M$. So, on the position (d_i, d_j) of X_1 we have x_{d_i,d_j}.

Since we assumed that $x_0 = 1$, for the $0-1$ variables x_i we have $x_0 x_i = x_i x_i$ and therefore we set the constraint $x_{0,d_i} = x_{d_i,d_i}$, for $i \in \{1, \dots, M\}$ to hold for X_1.

Also, if d_i and d_j can not simultaneously be in any solution, $x_{d_i} x_{d_j} = 0$ and in X_1 we can constrain $x_{d_i,d_j} = 0$.

This leads to the convex region R_1 described by the following constraints:

$$\begin{aligned}
&\sum_{\substack{i,j=0,\dots,M \\ d_j - d_i = d_k}} x_{d_i,d_j} = v(d_k) \text{ for } k = 1, \dots, M \\
&\sum_{i=0,\dots,M} x_{d_i,d_j} = n x_{0,d_j} \text{ for } j = 0, \dots, M \\
&x_{0,0} = 1, \\
&x_{0,d_i} = x_{d_i,d_i}, \text{ for } i = 1, \dots, M, \\
&x_{d_i,d_j} = 0, \text{ if } d_i \text{ and } d_j \text{ can not both be in a solution,}
\end{aligned} \qquad (R_1)$$

$$x_{d_i,d_j} \geq 0, \text{ for } i,j = 1,\ldots,M,$$
$$X_1 \text{ positive semidefinite.}$$

The constraints of the type $x_{d_i,d_j} = 0$, if d_i and d_j can not both be in a solution, are called the *pyramid constraints* because the numbers in the pyramid constructed from the difference set of the set $\{0, d_i, d_j, d_M\}$ must form a submultiset of ΔX.

Note that for any point of R_1

$$\sum_{i=0,\ldots,M} x_{d_i,d_i} = n,$$

because

$$\sum_{i=0,\ldots,M} x_{d_i,d_i} = \sum_{i=0,\ldots,M} x_{d_i,d_0} =$$
$$= nx_{0,0} = n.$$

We now prove that (R_1) is a relaxation of (Q) in the sense that the $0-1$ solutions of (R_1) are related to the elements of Q. The elements of Q are $(M+1)$-tuples and the elements of R_1 are $(M+1) \times (M+1)$ matrices. The idea is that the set of vectors determined by $0-1$ diagonals of matrices in R_1 is equal to Q. More precisely, let K denote the projection cone determined by the diagonal elements of the matrices X_1 feasible for (R_1), and let K^0 denote the cone spanned by all $0-1$ vectors of K.

Then we can prove

Proposition 2.2 *The convex hull of Q is equal to K^0.*

Proof: We can arrange the $0-1$ values x_{d_i} of a point in Q into a vector y of size $M+1$, indexed by the differences d_i of $\Delta X'$. Similarly as in the proof of Proposition 2.1 we can see that the matrix yy^T satisfies all the constraints that define R_1, and therefore $y \in K^0$.

Conversely, if $y \in K^0$, then the matrix $yy^T \in R_1$ and it is easy to see that y satisfies all the constraints in the definition of Q. ∎

The convex region R_1 is a feasible region of a semidefinite program and therefore we can optimize over R_1 in polynomial time using standard algorithms for solving semidefinite programs.

In order to make the exposition clear, we homogenize R_1, to obtain a convex cone S_1, in the following way:

$$\begin{aligned}
&\sum_{\substack{i,j=0,\dots,M \\ d_j-d_i=d_k}} x_{d_i,d_j} = v(d_k)x_{0,0}, \text{ for } k=1,\dots,M \\
&\sum_{i=0,\dots,M} x_{d_i,d_j} = nx_{0,d_j} \text{ for } j=0,\dots,M \\
&x_{0,d_i} = x_{d_i,d_i}, \text{ for } i=1,\dots,M, \\
&x_{d_i,d_j} = 0, \text{ if } d_i \text{ and } d_j \text{ can not both be in a solution,} \\
&x_{d_i,d_j} \geq 0, \text{ for } i,j=1,\dots,M, \\
&X_1 \text{ positive semidefinite.}
\end{aligned} \qquad (S_1)$$

Note that we can obtain R_1 by intersecting the cone S_1 with the hyperplane $x_{0,0} = 1$. The computer implementation of the relaxation (R_1) shows that the instances for which that relaxation does not give the right answer to the turnpike problem (P) are rare and some classes are given in Chapter 5.

Another way of relaxing (Q) to a semidefinite program is to look at the vector x indexed by the pairs of elements of $\Delta X'$: $(0,0)$ and (d_i,d_j), for $i,j \in \{0,\dots,M\}$, $i<j$.

Again we can look at the matrix $X_2 = xx^T$. The diagonal elements of this matrix are

$$x_{00,00}$$

and

$$x_{d_id_j,d_id_j}, \text{ for } i,j \in \{0,\dots,M\}, i<j.$$

The off-diagonal elements of X_2 are

$$x_{00,d_id_j} \text{ for } i,j \in \{0,\dots,M\}, i<j$$

and

$$x_{d_id_j,d_kd_l}, \text{ for } i,j,k,l \in \{0,\dots,M\},\ i<j,\ k<l.$$

The matrix X_2 satisfies the following constraints:

for every $a \in \{1,\dots,M\}$ and $k,l \in \{0,\dots,M\}$:

$$\sum_{\substack{i,j=0,\dots,M \\ d_j-d_i=d_a}} x_{d_id_j,d_kd_l} = v(d_a)x_{00,d_kd_l},$$

for every $j,k,l \in \{0,\dots,M\}$:

$$\sum_{i=0,\dots,M} x_{d_id_j,d_kd_l} = nx_{0d_j,d_kd_l},$$

the pyramid constraints:

$$x_{d_id_j,d_kd_l} = 0, \text{ if } d_i,d_j,d_k,d_l \text{ can not simultaneously all occur in a solution,} \qquad (S_2)$$

and the mixing conditions for $i,j,k,l \in \{0,\dots M\}$:

$$\begin{aligned}
x_{00,d_id_j} &= x_{d_id_j,d_i,d_j},\\
x_{0d_i,0d_j} &= x_{d_id_j,d_i,d_j},\\
x_{0d_k,d_id_j} &= x_{0d_i,d_kd_j},\\
x_{0d_k,d_id_j} &= x_{0d_j,d_id_k},\\
x_{0d_k,d_id_j} &= x_{d_kd_i,d_kd_j},\\
x_{0d_k,d_id_j} &= x_{d_jd_i,d_jd_k},\\
x_{0d_k,d_id_j} &= x_{d_id_j,d_id_k},\\
x_{0d_i,d_jd_k} &= x_{Md_i,d_jd_k},
\end{aligned}$$

$$x_{d_id_j,d_kd_l} = x_{d_id_k,d_jd_l},$$
$$x_{d_id_j,d_kd_l} = x_{d_id_l,d_jd_k}.$$

$$x_{d_id_j,d_kd_l} \geq 0,$$
$$X_2 \text{ positive semidefinite.}$$

Note that in the above definition of S_2, in order to simplify the notation, we did not always specify that for a variable $x_{d_id_j,d_kd_l}$ $d_i < d_j$ and $d_k < d_l$. Because of the mixing conditions, all variables indexed by the elements of the set $\{d_i, d_j, d_k, d_l\}$ have the same value, so we can index a variable by the set $\{d_i, d_j, d_k, d_l\}$.

The mixing constraints arise from the fact that an element $x_{d_id_j,d_kd_l}$ of a matrix X_2 in S_2 can be regarded as a product of four indicator $0-1$ variables x_{d_i}, x_{d_j}, x_{d_k} and x_{d_l}. Also we know that $x_0 = x_{d_M} = 1$.

The pyramid constraints got their name because for any $\{d_i, d_j, d_k, d_l\} \subset \Delta X'$ we can calculate the difference set of the set $\{d_i, d_j, d_k, d_l, 0, d_M\}$ and organize the differences in the pyramid as described in Chapter 1. The entry $x_{d_id_j,d_kd_l}$ is equal to 0 if the elements of the pyramid are not a subset of ΔX.

A matrix X_2 feasible for (S_2) has many interesting properties, one of the most interesting being that it contains matrices feasible for (S_1) that have some additional properties.

Let $Z^2_{d_id_j}$ be the matrix whose elements are the elements of the row of X_2 whose diagonal element is $x_{d_id_j,d_id_j}$ such that for $d_a, d_b \in \Delta X'$

$$(Z^2_{d_id_j})_{d_ad_a} = (X_2)_{d_id_j,0d_a},$$

and

$$(Z^2_{d_id_j})_{d_ad_b} = (X_2)_{d_id_j,d_ad_b}.$$

The matrices $Z^2_{d_id_j}$ are $(M+1)\times(M+1)$ matrices and we prove that they satisfy all the equality constraints in (S_1).

We also prove that the matrices $Z^2_{0d_i}$ that arise from the rows of X_2 indexed by the pair of differences $(0, d_i)$ are positive-semidefinite for any $i \in \{0, \dots, M\}$. We denote these matrices by

$$X^2_{d_i} = Z^2_{0d_i}.$$

The matrix Z^2_{00} is of special interest and is denoted by X^2_1. So,

$$X^2_1 = Z^2_{00}.$$

If X_2 is a matrix in $S_2 \cap H_2$, where H_2 is the hyperplane $x_{00,00} = 1$, then X^2_1 is an element of $S_1 \cap H_1$, where H_1 is the hyperplane $x_{0,0} = 1$. The matrix X^2_1 has the property that for any $i \in \{1, \dots, M-1\}$, it can be represented as a convex combination of two matrices that have 1 on the diagonal position indexed by the difference 0, and 0 or 1 on the diagonal position indexed by d_i. In order to see this, first we prove

Proposition 2.3 *The matrices $Z^2_{d_id_j}$ satisfy all the equality constraints in (S_1).*

Proof: Note that the equality constraints of (S_1) are a subset of the equality constraints of (S_2). For example, the constraint

$$\sum_{d_j - d_i = d_a} x_{d_id_j,d_kd_l} = v(d_a)x_{00,d_kd_l}$$

holds for any choice of d_k and d_l, so in particular when $d_k = d_l = 0$, we have

$$\sum_{d_j - d_i = d_a} x_{d_id_j,00} = v(d_a)x_{00,00}.$$

More precisely, we have:

$$\begin{aligned}\sum_{\substack{a,b=0,\dots,M \\ d_b - d_a = d_k}} (Z^2_{d_id_j})_{d_ad_b} = \sum_{\substack{a,b=0,\dots,M \\ d_b - d_a = d_k}} x_{d_id_j,d_ad_b} &= \\ &= v(d_k)x_{d_id_j,00} \\ &= v(d_k)(Z^2_{d_id_j})_{00} \text{ for } k \in \{1, \dots, M\}\end{aligned}$$

and

$$\sum_{a=0,\ldots,M} (Z^2_{d_i d_j})_{d_a d_b} = \sum_{a=0,\ldots,M} x_{d_i d_j, d_a d_b} =$$
$$= n x_{d_i d_j, 0 d_b} =$$
$$= n (Z^2_{d_i d_j})_{0 d_b} \text{ for } b \in \{0, \ldots, M\}$$

and

$$(Z^2_{d_i d_j})_{0 d_a} = x_{d_i d_j, 0 d_a} = (Z^2_{d_i d_j})_{d_a d_a},$$

and

$$(Z^2_{d_i d_j})_{d_k d_l} = 0, \text{ if } d_k \text{ and } d_l \text{ can not both be in a solution,}$$

because of the pyramid constraints. The above calculations prove the claim of the proposition. ∎

In the next proposition we prove that X_{d_i} is positive-semidefinite for any $i \in \{0, \ldots, M\}$ and therefore, because of Proposition 2.3, feasible for (S_1).

The next proposition also proves that the matrix

$$X^2_1 - X^2_{d_i}$$

is feasible for (S_1) for $i \in \{0, \ldots, M\}$.

Proposition 2.4 *For any $i \in \{0, \ldots, M\}$:*

1. *The matrix $X^2_{d_i}$ is feasible for (S_1).*
2. *The matrix $X^2_1 - X^2_{d_i}$ is feasible for (S_1).*
3. $X^2_1 = X^2_{d_i} + (X^2_1 - X^2_{d_i})$.

Proof: Since X_2 is a positive-semidefinite matrix, there exists a matrix V, such that

$$X_2 = V^T V.$$

For any two differences $d_a, d_b \in \Delta X'$, $d_a < d_b$, let $v_{d_a d_b}$ be the column vector of V indexed by the pair of differences d_a, d_b.

For a fixed $i \in \{0, \dots, M\}$, let V_{d_i} be the matrix whose columns are the vectors $v_{d_i d_j}$ for $j \in \{0, \dots, M\}$. Then

$$X^2_{d_i} = V^T_{d_i} V_{d_i}.$$

because

$$(X^2_{d_i})_{d_k, d_l} = (X_2)_{0d_i, d_k d_l},$$

and

$$(V^T_{d_i} V_{d_i})_{d_k, d_l} = v_{d_i d_k} v_{d_i d_l} = (X_2)_{0d_i, d_k d_l}.$$

Therefore, $X^2_{d_i}$ is positive-semidefinite and satisfies all the equality constraints from (S_1) because of Proposition 2.3, so we can conclude that $X^2_{d_i}$ is in S_1.

To see that $X^2_1 - X^2_{d_i}$ is in S_1, let W_{d_i} be the matrix whose columns are the vectors $v_{0d_j} - v_{d_i d_j}$ for $j = 0, \dots, M$. Then

$$\begin{aligned}
(W^T_{d_i} W_{d_i})_{d_k, d_l} &= (v_{0d_k} - v_{d_i d_k})(v_{0d_l} - v_{d_i d_l}) = \\
&= v_{0d_k} v_{0d_l} - v_{d_i d_k} v_{0d_l} - v_{0d_k} v_{d_i d_l} + v_{d_i d_k} v_{d_i d_l} = \\
&= (X_2)_{d_k d_l, d_k d_l} - (X_2)_{d_i d_l, d_i d_k} - (X_2)_{d_i d_l, d_i d_k} + (X_2)_{d_i d_l, d_i d_k} = \\
&= (X_2)_{d_k d_l, d_k d_l} - (X_2)_{d_i d_l, d_i d_k} = \\
&= (X^2_1)_{d_k d_l} - (X^2_{d_i})_{d_k d_l},
\end{aligned}$$

because of the definition of X^2_1, $X^2_{d_i}$ and the mixing constraints in (S_1).

Therefore,

$$W^T_{d_i} W_{d_i} = X^2_1 - X^2_{d_i}.$$

The matrix $X^2_1 - X^2_{d_i}$ is positive-semidefinite and satisfies all the equality constraints in (S_1) because these constraints are homogeneous and both matrices X^2_1 and $X^2_{d_i}$ satisfy them.

The third part of the proposition follows directly from the first two. ■

The third part of Proposition 2.4 is important because of the following properties of the matrices $X_{\bar{1}}^2$, $X_{d_i}^2$ and $X_{\bar{1}}^2 - X_{d_i}^2$:

The diagonal entry of $X_{d_i}^2$ indexed by the difference d_i is equal to the diagonal entry indexed by the difference 0, i.e.

$$(X_{d_i}^2)_{d_i,d_i} = (X_{d_i}^2)_{00},$$

because

$$(X_{d_i}^2)_{d_i,d_i} = (X_2)_{d_id_i,0d_i} = (X_2)_{00,0d_i} = (X_{d_i}^2)_{00}.$$

and the diagonal entry of $(X_{\bar{1}}^2 - X_{d_i}^2)$ indexed by the difference d_i is 0, i.e.

$$(X_{\bar{1}}^2 - X_{d_i}^2)_{d_i,d_i} = 0,$$

because

$$(X_{\bar{1}}^2 - X_{d_i}^2)_{d_i,d_i} = (X_2)_{00,0d_i} - (X_2)_{d_id_i,0d_i} = 0.$$

So, the third part of Proposition 2.4 tells us that for any $i \in \{1, \ldots, M-1\}$, the matrix $X_{\bar{1}}^2$ that is a submatrix of a matrix X_2 feasible for (S_2) can be represented as a sum of two matrices such that one of them has the diagonal entry indexed by the difference d_i equal to the diagonal entry indexed by 0, and the other matrix has the diagonal entry indexed by d_i equal to 0.

Moreover, if $X_{\bar{1}}^2 \in S_1 \cap H_1$, where H_1 is the hyperplane $x_{0,0} = 1$ and $(X_2)_{0d_i,0d_i} = \alpha$ we have

$$X_{\bar{1}}^2 = \alpha(\frac{1}{\alpha} X_{d_i}^2) + (1-\alpha)(\frac{1}{1-\alpha}(X_{\bar{1}}^2 - X_{d_i}^2)).$$

so $X_{\bar{1}}^2$ is represented as a convex combination of two matrices that have 1 on the diagonal entry indexed by the difference 0, and 0 or 1 on the diagonal entry indexed by the difference d_i.

This additional property of the matrices $X_{\bar{1}}^2$ that arise from the matrices feasible for (S_2) induces a cut on the cone K determined by the diagonal elements of the matrices feasible for (S_1). This is because the matrix X_1^2 has the property that for each $i = 1, \ldots, M$, the row indexed by the difference d_i is in the projection cone K, and the difference of the zero-th row of X_1^2 and the row indexed by d_i is in K. The row of X_1^2 indexed by d_i is just the diagonal of X_{d_i} and the difference of the zero-th row of X_1^2 and the row indexed by d_i is the diagonal of $X_1^2 - X_{d_i}^2$.

The cone determined by the diagonal elements of the matrices X_1^2 that arise from feasible matrices for the relaxation (S_2), denoted K_2, is similar to the cone $N_+(K)$ introduced by Lovász and Schrijver in [19] in the sense that every element of $K_2 \cap H_0$, where H_0 is the hyperplane $x_0 = 1$, can be represented as a convex combination of two elements of K that have a 0 or 1 on a position indexed by the difference d_i, for any $i \in \{1, \ldots, M-1\}$. This result is similar to Lemma 1.2.

Unfortunately, the matrices $X_{d_i}^2$ and $(X_1^2 - X_{d_i}^2)$ do not obviously arise from matrices feasible for (S_2), so the above decomposition can not be iterated.

In order to iterate the above construction we can define a sequence of feasible regions for semidefinite programs. The first region in the sequence is (S_1), the second is (S_2).

Any other relaxation (S_k) in the sequence has the property that it contains a feasible matrix for the relaxation (S_1) that can be represented as a sum of matrices whose entries on $k-1$ fixed diagonal places are either 0 or equal to the entry indexed by $0, 0$.

In the k-th relaxation, we consider the vector x indexed by the k-tuples of elements of $\Delta X'$:

$$\underbrace{00\ldots0}_{k \text{ times}}$$

$$\underbrace{00\ldots0}_{k-1 \text{ times}} d_{i_1}$$

$$\underbrace{00\ldots0}_{k-2 \text{ times}} d_{i_1} d_{i_2}$$

$$\vdots$$

$$\vdots$$

$$0, d_{i_1} d_{i_2} \dots d_{i_{k-1}}$$

$$d_{i_1} d_{i_2} \dots d_{i_{k-1}} d_{i_k},$$

where $d_{i_1} < d_{i_2} < \cdots < d_{i_k}$.

The elements of the vector x can be regarded as the product of k indicator variables, i.e. variables whose value is equal to 1 if the indicated difference is in a solution and 0 otherwise.

For any multiset of differences $A = \{d_{i_1}, d_{i_2}, \dots, d_{i_k}\}$, where the differences $d_i \in \Delta X'$ can appear at most once and 0 can appear more than once, the permutation of A obtained by sorting the differences in ascending order is called the *proper index* of A.

Let I be the set of all proper indices.

In order to simplify the notation, we assume that all the permutations of the multiset A are equivalent to the proper index of A, and sometimes we index the elements of x by the multiset associated with the proper index.

Again, we can look at the matrix $X_k = xx^T$. This matrix must satisfy the following conditions:

for every $a \in \{1, \dots, M\}$ and every $A = \{d_{i_1}, d_{i_2}, \dots, d_{i_k}\}$, and any $i \in I$:

$$\sum_{\substack{i_1, i_2 = 0, \dots, M \\ d_{i_2} - d_{i_1} = d_a}} x_{d_{i_1} d_{i_2} d_{i_3} \dots d_{i_k}, i} = v(d_a) x_{00 d_{i_3} \dots d_{i_k}, i},$$

for every $A = \{d_{i_2}, \dots, d_{i_k}\}$, and any $i \in I$

$$\sum_{i_1 = 0, \dots, M} x_{d_{i_1} d_{i_2} d_{i_3} \dots d_{i_k}, i} = n x_{0 d_{i_2} \dots d_{i_k}, i},$$

the pyramid constraints:

$$x_{l_1, l_2} = 0, \text{ if the elements of the indices } l_1 \text{ and } l_2 \text{ can not simultaniously all occur in a solution,} \qquad (S_k)$$

the mixing conditions similar to the mixing conditions in (S_2)

$$x_{l_1,l_2} = x_{k_1,k_2},$$

for any four proper indices such that the elements different from 0 and M are the same in $l_1 \cup l_2$ and $k_1 \cup k_2$, although some might appear different number of times in $l_1 \cup l_2$ than in $k_1 \cup k_2$

$$x_{l_1,l_2} \geq 0, \text{ for any two proper indices } l_1, l_2.$$

$$X_k \text{ positive semidefinite.}$$

The mixing conditions are obtained from the fact that a variable $x_{d_{i_1},d_{i_2},\ldots,d_{i_k}}$ is actually a product of k indicator $0-1$ variables.

Note that the mixing constraints contain the following: if the multiset $l_1 \cup l_2$ contains two copies of the difference d_j, we can replace one copy of d_j by 0 in $l_1 \cup l_2$ and write a mixing constraint. If l_1 or l_2 contains 0, we can replace it with M and write a mixing constraint.

Now, we examine the properties of a matrix X_k feasible for (S_k). These properties are similar to the properties of the matrices X_2 feasible for (S_2). First we prove that a matrix X_k in $S_k \cap H_k$, where H_k is the hyperplane

$$x_{\underbrace{00\ldots0}_{k \text{ times}}\underbrace{00\ldots0}_{k \text{ times}}} = 1$$

contains as a submatrix a matrix that is contained in $S_i \cap H_i$, for any $i \in \{1, \ldots, k\}$.

For $j \in \{0, \ldots, M\}$, let X_j^k be the submatrix of $X_k \in S_k$ determined by the rows and columns indexed by all possible submultisets $\{d_{i_1}, \ldots, d_{i_j}\}$ that determine a proper index in (S_j). The diagonal elements of X_j^k are

$$(X_j^k)_{d_{i_1}\ldots d_{i_j},d_{i_1}\ldots d_{i_j}} = (X_k)_{\underbrace{00\ldots0}_{k-j \text{ times}}d_{i_1}\ldots d_{i_j},\underbrace{00\ldots0}_{k-j \text{ times}}d_{i_1}\ldots d_{i_j}},$$

and the off-diagonal elements of X_j^k are

$$(X_j^k)_{d_{i_1}\dots d_{i_j},d_{l_1}\dots d_{l_j}} = (X_k)_{\underbrace{00\dots0}_{k-j \text{ times}} d_{i_1}\dots d_{i_j},\underbrace{00\dots0}_{k-j \text{ times}} d_{l_1}\dots d_{l_j}},$$

The matrices X_j^k are just a generalization of the matrices X_1^2 defined previously. We can prove:

Proposition 2.5 *The matrix X_j^k is feasible for (S_j), for any $j = 1, \dots, k$.*

Proof: The matrix X_j^k is positive-semidefinite as a submatrix of the positive semidefinite matrix X_k. The equality constraints of (S_j) hold because they are just a subset of the equality constraints of (S_k). The formal proof follows that of Proposition 2.3. ■

Now we generalize the matrices $X_{d_i}^2$. We define the matrix $(X_{d_i}^k)$ in the following way:

$$(X_{d_i}^k)_{d_{a_1}\dots d_{a_{k-1}},d_{b_1}\dots d_{b_{k-1}}} = (X_k)_{d_i,d_{a_1}\dots d_{a_{k-1}},d_i d_{b_1}\dots d_{b_{k-1}}},$$

for any two proper indices $d_{a_1}\dots d_{a_{k-1}}$ and $d_{b_1}\dots d_{b_{k-1}}$ for (S_{k-1}).

We can now generalize Proposition 2.4:

Proposition 2.6 *For any $i \in \{1, \dots, M\}$:*

1. *The matrix $X_{d_i}^k$ is feasible for (S_{k-1}).*

2. *The matrix $X_{k-1}^k - X_{d_i}^k$ is feasible for (S_{k-1})*

3. $X_{k-1}^k = X_{d_i}^k + (X_{k-1}^k - X_{d_i}^k)$.

Proof: We just outline the proof since it is essentially the same as the proof of Proposition 2.4.

Since X_k is a positive-semidefinite matrix, there exist a matrix V such that

$$X_k = V^T V.$$

Let $v_{d_{i_1}\dots d_{i_k}}$, for any proper index $d_{i_1},\dots,d_{i_k}$, be the column vectors of V.

For a fixed i let V_{d_i} be the matrix whose column vectors are

$$v_{d_i d_{a_1}\dots d_{a_{k-1}}}$$

for any proper index $d_{a_1}\dots d_{a_{k-1}}$ for (S_{k-1}).

Similarly, let W_{d_i} be the matrix whose columns are

$$v_{0 d_{a_1}\dots d_{a_{k-1}}} - v_{d_i d_{a_1}\dots d_{a_{k-1}}}.$$

Then it is easy to see that

$$X^k_{d_i} = V^T_{d_i} V_{d_i}$$

and

$$X^k_{k-1} - X^k_{d_i} = W^T_{d_i} W_{d_i},$$

and therefore the matrices $X^k_{d_i}$ and $X^k_{k-1} - X^k_{d_i}$ are positive-semidefinite.

To see that these matrices satisfy all the equality constraints follow the proof of Proposition 2.3. ■

The matrices X^k_{k-1}, $X^k_{d_i}$ and $X^k_{k-1} - X^k_{d_i}$ from Proposition 2.6 contain as submatrices matrices feasible for (S_1). Notice that the matrix $(X^k_{k-1})^{k-1}_1$ is equal to the matrix X^k_1, i.e.

$$(X^k_{k-1})^{k-1}_1 = X^k_1.$$

The matrix

$$(X^k_{d_i})^{k-1}_1$$

is feasible for (S_1) and has the diagonal entries

$$(X_k)_{d_i 0\dots 0 d_j, d_i 0\dots 0 d_j} \text{ for } j = 0,\dots,M.$$

and the matrix

$$(X_{k-1}^k - X_{d_i}^k)_1^{k-1}$$

is also feasible for (S_1) and has the diagonal entries

$$(X_k)_{0\ldots 0d_j,0\ldots 0d_j} - (X_k)_{d_i 0\ldots 0d_j,d_i 0\ldots 0d_j} \text{ for } j = 0,\ldots,M.$$

In particular, the diagonal entry of $(X_{d_i}^k)_1^{k-1}$ indexed by the difference d_i is the same as the diagonal entry indexed by the difference 0, and the diagonal entry of $(X_{k-1}^k - X_{d_i}^k)_1^{k-1}$ indexed by the difference d_i is equal to 0.

Also, we have

$$(X_{k-1}^k)_1^{k-1} = (X_{d_i}^k)_1^{k-1} + (X_{k-1}^k - X_{d_i}^k)_1^{k-1}$$

which follows from Proposition 2.6 by taking the appropriate submatrices of X_{k-1}^k, $X_{d_i}^k$ and $X_{k-1}^k - X_{d_i}^k$.

Therefore, we have proved:

Lemma 2.7 *Let X_k be a feasible matrix for (S_k) and let X_1^k be its submatrix feasible for (S_1). Then for any $i \in \{1,\ldots,M\}$, X_1^k can be represented as a sum of two matrices that are feasible for (S_1). One matrix has the diagonal entry on the position indexed by the difference d_i equal to the diagonal entry indexed by the difference 0. The other matrix has the diagonal entry indexed by the difference d_i equal to 0.*

Moreover, if the matrix X_k is in $S_k \cap H_k$, where H_k is the hyperplane $x_{0\ldots 0,0\ldots 0} = 1$, than X_1^k can be represented as a convex combination of the two matrices. Both matrices have 1 on the position indexed by the difference 0, and one has 1 on the position indexed by the difference d_i and the other has 0 on that position.

■

Now we are ready to prove the central theorem of this section.

Theorem 2.8 *A feasible matrix X_k for (S_k) contains the matrix X_1^k feasible for (S_1) that can be represented as a sum of matrices that are feasible for (S_1) and on fixed $k-1$ diagonal places have entries that are either equal to the diagonal entry indexed by the difference 0, or are equal to 0.*

If $X_k \in S_k \cap H_k$, where H_k is the hyperplane $x_{0\ldots0,0\ldots0} = 1$, the matrix X_1^k can be represented as a convex combination of matrices that are feasible for (S_1) and have 0 or 1 on fixed $k-1$ diagonal places.

Proof: We prove the first statement of the theorem by induction on k. The second statement can be proved by slight modifications of this proof.

When $k = 2$, the statement follows directly from Lemma 2.7.

So, let us assume that for a $k > 2$, any feasible matrix X_k for (S_k), the matrix X_1^k feasible for (S_1) can be represented as a sum of matrices that are feasible for (S_1) and on fixed $k-1$ diagonal places have entries that are either equal to the diagonal entry indexed by the difference 0, or are 0. Let these places be the entries indexed by differences $d_{i_1}, \ldots, d_{i_k}$.

Let X_{k+1} be a feasible matrix for (S_{k+1}) and let $d_{i_{k+1}}$ be a difference different than $d_{i_1}, \ldots, d_{i_k}$. By Proposition 2.6 and Lemma 2.7, we can represent the submatrix X_k^{k+1} as the sum of two matrices

$$X_k^{k+1} = X_{d_{k+1}}^{k+1} + (X_k^{k+1} - X_{d_{k+1}}^{k+1})$$

such that for the submatrices $(X_k^{k+1})_1^k$, $(X_{d_{k+1}}^{k+1})_1^k$ and $(X_k^{k+1} - X_{d_{k+1}}^{k+1})_1^k$ we have

$$(X_k^{k+1})_1^k = (X_{d_{k+1}}^{k+1})_1^k + (X_k^{k+1} - X_{d_{k+1}}^{k+1})_1^k.$$

Furthermore, the matrix $(X_{d_{k+1}}^{k+1})_1^k$ has the diagonal entry indexed by the difference d_{k+1} equal to the diagonal entry indexed by 0, and the matrix $(X_k^{k+1} - X_{d_{k+1}}^{k+1})_1^k$ has 0 on the diagonal entry indexed by d_{k+1}.

The matrix $(X_k^{k+1})_1^k$ is equal to the matrix X_1^{k+1}.

Now the matrices $X_{d_{k+1}}^{k+1}$ and $X_k^{k+1} - X_{d_{k+1}}^{k+1}$ are feasible for (S_k) by Proposition 2.6 and by the induction hypothesis the matrices $(X_{d_{k+1}}^{k+1})_1^k$ and $(X_k^{k+1} - X_{d_{k+1}}^{k+1})_1^k$ can be represented as a sum of matrices that are feasible for (S_1) and on diagonal places indexed by the differences $d_{i_1}, \ldots, d_{i_k}$ have entries that are either equal to the entry indexed by the difference 0, or are equal to 0.

Therefore the matrix X_1^{k+1} can be represented as a sum of matrices that are feasible for (S_1), and on fixed k diagonal places have entries that are either equal to the entry indexed by the difference 0, or are equal to 0.

This completes the proof. ■

Let K_i be the projection cone determined by the diagonal elements of the matrices X_1^i, for $i \in \{1, \ldots, M+1\}$. Then $K_{i+1} \subseteq K_i$, and every point of $K_{i+1} \cap H_0$, where H_0 is the hyperplane $x_0 = 1$, can be represented as a convex combination of two elements of K_i that have 0 or 1 on the position indexed by some difference d_j, for any $j \in \{1, \ldots, M-1\}$. Therefore we have obtained a sequence of cones such that

$$K_{M+1} \subseteq K_M \subseteq \ldots K_2 \subseteq K,$$

and each relaxation (S_i) introduces further cuts on the cone K. The cone K_{M+1} is obviously equal to K^0. Unfortunately, the size of the problem (S_{M+1}) is not obviously polynomial.

Furthermore, we have a theorem similar to Theorem 1.5:

Theorem 2.9 *The turnpike problem is polynomial time solvable for classes of instances for which there exist a constant c, such that $K^0 = K_c$ for each instance in the class.*

In practice, no instances of the turnpike problem for which the relaxation (S_2) does not give the correct answer are known. That is, there is no known instance of the turnpike problem for which $K^0 \neq K_2$.

In Chapter 5, we discuss instances for which relaxations (S_1) and (S_2) give the correct answer to problem (P).

2.3 Some additional properties of the matrices X_k

Here we list some interesting properties of the matrices X_k. For example we can also generalize the matrices Z_{d_i,d_j} defined in the previous section, in the following way:

Let X_k be a feasible matrix for (S_k) and let $l = \lfloor \frac{k}{2} \rfloor$. Let

$$(Z^k_{d_1 \ldots d_k})_{d_{a_1} \ldots d_{a_l}, d_{b_1} \ldots d_{b_l}} = (X_k)_{d_1 \ldots d_k, d_{a_1} \ldots d_{a_l} d_{b_1} \ldots d_{b_l}}$$

if k is even and

$$(Z^k_{d_1 \ldots d_k})_{d_{a_1} \ldots d_{a_l}, d_{b_1} \ldots d_{b_l}} = (X_k)_{d_1 \ldots d_k, 0 d_{a_1} \ldots d_{a_l} d_{b_1} \ldots d_{b_l}}$$

if k is odd.

Then we can prove a proposition similar to Proposition 2.3:

Proposition 2.10 *The matrices $(Z^k_{d_1 \ldots d_k})$ satisfy all the equality constraints in (S_l).*

Proof: Similar to the proof of Proposition 2.3. ■

Another way of generalizing matrices $X^2_{d_i}$ is to define matrices $X^{2k}_{d_{i_1} \ldots d_{i_k}}$ in the following way.

Let X_{2k} be a matrix feasible for (S_{2k}) and let X^{2k}_k be its submatrix feasible for (S_k) as in Proposition 2.5. For any proper index $\{d_{i_1}, \ldots, d_{i_k}\}$ let

$$X^{2k}_{d_{i_1} \ldots d_{i_k}}$$

be the matrix determined by the elements of the row of X_{2k} indexed by the $d_{i_1}, \ldots, d_{i_k}$, i.e. the matrix such that

$$(X^{2k}_{d_{i_1} \ldots d_{i_k}})_{d_{a_1} \ldots d_{a_k}, d_{b_1} \ldots d_{b_k}} = (X_{2k})_{d_{i_1} \ldots d_{i_k} d_{a_1} \ldots d_{a_k}, d_{i_1} \ldots d_{i_k} d_{b_1} \ldots d_{b_k}}$$

Let

$$Y^{2k}_{d_{i_1} \ldots d_{i_k}} = X^{2k}_k - X^{2k}_{d_{i_1} \ldots d_{i_k}}.$$

Then another generalization of Proposition 2.4 is:

Proposition 2.11 *For any proper index $\{d_{i_1}, \ldots, d_{i_k}\}$:*

1. *The matrix $X^{2k}_{d_{i_1} \ldots d_{i_k}}$ is feasible for (S_k).*

2. *The matrix $Y^{2k}_{d_{i_1} \ldots d_{i_k}}$ is feasible for (S_k) and*

3. $X_k^{2k} = X_{d_{i_1}\dots d_{i_k}}^{2k} + Y_{d_{i_1}\dots d_{i_k}}^{2k}$.

Proof: The proof is similar to the proof of Proposition 2.4. For any proper index $\{d_{i_1}, \dots, d_{i_k}\}$, it is easy to check that the matrix $X_{d_{i_1}\dots d_{i_k}}^{2k}$ satisfies all the equality constraints in (S_k). It is just a matter of recognizing that these constraints are the subset of the equality constraints for (S_{2k}).

It is a bit harder to prove that these matrices are positive-semidefinite.

Let

$$X_{2k} = V^T V,$$

and let $v_{d_{j_1}\dots d_{j_{2k}}}$ be the column vectors of V.

Then

$$X_{d_{i_1}\dots d_{i_k}}^{2k} = W^T W, \tag{2.1}$$

where W is a matrix whose columns are column vectors of V

$$v_{d_{i_1}\dots d_{i_k}, d_{a_1}\dots d_{a_k}},$$

for $d_{a_1}, \dots, d_{a_k}$ all possible indices for matrix feasible for (S_k).

The equation (2.1) is easy to verify, by using the definitions of $X_{d_{i_1}\dots d_{i_k}}^{2k}$, and the mixing constraints of (S_{2k}) and is essentially the same as the proof in Proposition 2.4.

The matrix $X_k^{2k} - X_{d_{i_1}\dots d_{i_k}}^{2k}$ satisfies all the equality constraints in (S_k) because it is a difference of two matrices that satisfy those constraints. We need to prove that $X_k^{2k} - X_{d_{i_1}\dots d_{i_k}}^{2k}$ is positive-semidefinite. So, let U be the matrix whose columns are the vectors

$$v_{0\dots 0, d_{a_1}\dots d_{a_k}} - v_{d_{i_1}\dots d_{i_k}, d_{a_1}\dots d_{a_k}}$$

for $d_{a_1} \dots d_{a_k}$ all possible indices for matrix feasible for (S_k).

Note that

$$\begin{aligned}
&(v_{0\ldots0,d_{a_1}\ldots d_{a_k}} - v_{d_{i_1}\ldots d_{i_k},d_{a_1}\ldots d_{a_k}})(v_{0\ldots0,d_{b_1}\ldots d_{b_k}} - v_{d_{i_1}\ldots d_{i_k},d_{b_1}\ldots d_{b_k}}) = \\
&= (X_{2k})_{0\ldots0d_{a_1}\ldots d_{a_k},0\ldots0d_{b_1}\ldots d_{b_k}} - (X_{2k})_{0\ldots0d_{a_1}\ldots d_{a_k},d_{i_1}\ldots d_{i_k}d_{b_1}\ldots d_{b_k}} + \\
&+ (X_{2k})_{d_{i_1}\ldots d_{i_k}d_{a_1}\ldots d_{a_k},0\ldots0d_{b_1}\ldots d_{b_k}} - (X_{2k})_{d_{i_1}\ldots d_{i_k}d_{a_1}\ldots d_{a_k},d_{i_1}\ldots d_{i_k}d_{b_1}\ldots d_{b_k}} = \\
&= (X_{2k})_{0\ldots0d_{a_1}\ldots d_{a_k},0\ldots0d_{b_1}\ldots d_{b_k}} - (X_{2k})_{0\ldots0d_{a_1}\ldots d_{a_k},d_{i_1}\ldots d_{i_k}d_{b_1}\ldots d_{b_k}} = \\
&= (X_k^{2k})_{d_{a_1}\ldots d_{a_k},d_{b_1}\ldots d_{b_k}} - (X_{d_{i_1},d_{i_k}}^{2k})_{d_{a_1}\ldots d_{a_k},d_{b_1}\ldots d_{b_k}}.
\end{aligned}$$

Therefore,

$$X_k^{2k} - X_{d_{i_1},\ldots,d_{i_k}}^{2k} = U^T U,$$

which completes the proof. ∎

Now, we show a property of the matrices $X_{d_{i_1},\ldots,d_{i_k}}^{2k}$, namely, we have:

Proposition 2.12 *The matrix*

$$\frac{1}{(X_{d_{i_1}\ldots d_{i_k}}^{2k})_{0\ldots0,0\ldots0}} X_{d_{i_1}\ldots d_{i_k}}^{2k}$$

has 2^k *diagonal entries equal to* 1.

Proof: The diagonal entries of the matrix $(X_{d_{i_1}\ldots d_{i_k}}^{2k})$ are

$$(X_{d_{i_1}\ldots d_{i_k}}^{2k})_{d_{a_1}\ldots d_{a_k},d_{a_1}\ldots d_{a_k}} = (X_{2k})_{d_{i_1}\ldots d_{i_k}d_{a_1}\ldots d_{a_k},d_{i_1}\ldots d_{i_k}d_{a_1}\ldots d_{a_k}}.$$

For any set A, if $A \subseteq \{d_{i_1}, \ldots, d_{i_k}\}$ the diagonal elements determined by A and the appropriate number of zeros are equal to $x_{0,\ldots0,d_{i_1}\ldots d_{i_k},0,\ldots0,d_{i_1}\ldots d_{i_k}} = (X_{d_{i_1}\ldots d_{i_k}}^{2k})_{0\ldots0,0\ldots0}$ because of the mixing constraints in (S_{2k}). This proves the first statement of the theorem.

∎

Therefore, any matrix X_k feasible for (S_k) contains a submatrix feasible for (S_1) that has $k-1$ diagonal entries equal to the diagonal entry indexed by the difference 0.

Chapter 3

Classes for which the Turnpike Problem is Solvable in Polynomial Time

3.1 Introduction

In this chapter we give classes of instances of the turnpike problem that can be solved in polynomial time.

We say that an instance ΔX of the turnpike problem is solved by its relaxation (S_k) from Chapter 2, if the submatrix Y determined by the diagonal elements $x_{0\ldots 0d_i,0\ldots 0d_i}$, $d_i \in \Delta X'$, of a feasible matrix X_k for the relaxation (S_k) is of the form

$$Y = \sum_i \alpha_i s_i s_i^T,$$

where s_i is a characteristic vector of a solution of the turnpike instance ΔX and $\alpha_i > 0$.

In the first section we look at the instances that are solved by the relaxation (S_1) with modified pyramid constraints, i.e. the relaxation in which we only take a subset of the pyramid constraints valid for (S_1). We show that if for a given set X, the instance ΔX can be solved by the relaxation (S_1) with modified pyramid constraints, than the instance ΔY, where ΔY is the difference set of

$$Y = X \cup (X + a) \cup \ldots \cup (X + (m-1)a),$$

can be solved by the relaxation (S_1) with modified pyramid constraints. Here we have to choose a to be greater than the maximum element of ΔX,

Also, if the instance ΔX can be solved by the relaxation (S_1) with modified pyramid constraints and has the property that every solution contains a point that is not in any other solution, the instance ΔY, where ΔY is the difference set of

$$Y = X \cup (X + a_1) \cup \ldots \cup (X + a_k),$$

then the instance ΔY can be solved by the relaxation (S_1) with modified pyramid constraints. The numbers $a_1, \ldots, a_k$ have to satisfy

$$\begin{aligned} &a_1 \geq 3d_M + 1, \\ &a_i \geq 3a_{i-1} + d_M + 1 \text{ for } i \in \{2, \ldots, k\}, \end{aligned}$$

where d_M is the maximum element of ΔX.

Note that the instances that have only one solution, satisfy the above property.

In the second section we show that the relaxation (S_1) solves the instances constructed by Zhang, [35]. Therefore our technique solves the turnpike problem on these instances in polynomial time, whereas the backtracking procedure of Skiena et al. takes exponential time on these instances.

Finally in the last section we consider the instances ΔX that have unique solutions and all the differences in ΔX are different. We show that if during the execution of the Skiena's et al. backtracking procedure, k is the biggest number of steps that the procedure has to backtrack, the relaxation (S_{k+1}) solves the instance ΔX. That means that for this class of instances if k is constant, the relaxation (S_{k+1}) has polynomial size and can therefore be solved in polynomial time.

3.2 Generating bigger instances solvable in polynomial time from smaller instances solvable in polynomial time

In this section we show how to generate bigger instances that are solvable by the relaxation (S_1) with modified pyramid constraints, from smaller ones that can be solved by the same kind of relaxation.

By modified pyramid constraints, we mean that in the relaxation (S_1) of an instance ΔX of the turnpike problem we only include the pyramid constraints of the type

$$x_{d_i, d_M - d_i} = 0, \text{ if } d_i \text{ and } d_M - d_i \text{ can not both be in a solution,}$$

where $d_i \in \Delta X$. Note that this type of constraints depends only on the multiplicity of d_i and $d_M - d_i$ in ΔX. We can write the above constraint if and only if $v(d_i) = 1$ or $v(d_M - d_i) = 1$.

We denote the relaxation (S_1) with modified pyramid constraints by (S_1'). Note that the relaxation (S_1') is weaker than the relaxation (S_1) in the sense that any matrix feasible for (S_1) is also feasible for (S_1').

Some of the generated instances have more solutions than the instances they were derived from.

For a given set X, we define another set Y by

$$Y = X \cup (X + a) \cup \ldots \cup (X + (m-1)a),$$

where a is greater than the maximum element of X.

If ΔX is the difference set of X, and ΔY the difference set of Y, it is easy to see that ΔY is well-defined, in the sense that if X_1 is another solution set of the instance ΔX and

$$Y_1 = X_1 \cup (X_1 + a) \cup \ldots \cup (X_1 + (m-1)a),$$

then

$$\Delta Y = \Delta Y_1.$$

Now, we can prove:

Theorem 3.1 *Let $X = \{0 < a_1 < \cdots < a_{n-1}\}$ be a set and let ΔX be its difference set. Let a be a number such that $a > a_{n-1}$, and let $m > 1$ be an integer.*

Furthermore, let

$$Y = X \cup (X + a) \cup \ldots \cup (X + (m-1)a).$$

Then if the relaxation (S_1') solves the instance X of turnpike problem, it also solves the instance Y.

Proof: First notice that the cardinality of Y is mn and that the multiplicity of the difference a in ΔY is $(m-1)n$ and the multiplicity of the difference $(m-1)a$ in ΔY is n.

Let

$$A = V^T V$$

be a feasible matrix for the relaxation (S_1') of the instance ΔY of the turnpike problem. For ease of presentation, let us assume that the matrix A is indexed by all numbers between 0 and $a_{n-1} + (m-1)a$, and let v_i be the column of V indexed by i.

Now we can look at

$$\begin{aligned}
&\sum_{i=0}^{a_{n-1}+(m-2)a} (v_i - v_{i+a})^2 + \sum_{i=0}^{a_{n-1}} (v_i - v_{i+(m-1)a})^2 + \sum_{i=a_{n-1}+1}^{a-1} v_i^2 + \sum_{i=a_{n-1}+(m-2)a+1}^{(m-1)a-1} v_i^2 = \\
&= \sum_{i=0}^{a_{n-1}+(m-2)a} v_i^2 + \sum_{i=a}^{a_{n-1}+(m-1)a} v_i^2 - 2 \sum_{i=0}^{a_{n-1}+(m-2)a} v_i v_{i+a} + \\
&+ \sum_{i=0}^{a_{n-1}} v_i^2 + \sum_{i=(m-1)a}^{a_{n-1}+(m-1)a} v_i^2 - 2 \sum_{i=0}^{a_{n-1}} v_i v_{i+(m-1)a} + \\
&+ \sum_{i=a_{n-1}+1}^{a-1} v_i^2 + \sum_{i=a_{n-1}+(m-2)a+1}^{(m-1)a-1} v_i^2 =
\end{aligned} \tag{3.1}$$

$$= 2\sum_{i=0}^{a_{n-1}+(m-1)a} v_i^2 - 2\sum_{i=0}^{a_{n-1}+(m-2)a} v_i v_{i+a} - 2\sum_{i=0}^{a_{n-1}} v_i v_{i+(m-1)a} =$$
$$= 2mn - 2(m-1)n - 2n =$$
$$= 0.$$

Note that the above calculations do not change if we know that some of the vectors v_i are zero vectors.

From (3.1), we have

$$\begin{aligned}
&v_0 = v_a = v_{2a} = \ldots = v_{(m-1)a} \\
&v_{a_1} = v_{a_1+a} = v_{a_1+2a} = \ldots = v_{a_1+(m-1)a} \\
&v_{a_2} = v_{a_2+a} = v_{a_2+2a} = \ldots = v_{a_2+(m-1)a} \\
&\vdots \\
&v_{a_{n-1}} = v_{a_{n-1}+a} = v_{a_{n-1}+2a} = \ldots = v_{a_{n-1}+(m-1)a}
\end{aligned} \tag{3.2}$$

Since $a > a_{n-1}$ we have that in ΔY the multiplicity of $a_i + (m-1)a$ is equal to the multiplicity of a_i in ΔX, for $i \in \{1, \ldots, n-1\}$.

This combined with the fact that

$$\begin{aligned}
&v_0 = v_{(m-1)a} \\
&v_{a_i} = v_{a_i+(m-1)}a, \text{ for } i \in \{1, \ldots, n-1\}
\end{aligned}$$

which is a part of (3.2), enables us to conclude that the submatrix of A indexed by the elements of X satisfies all the constraints of the relaxation (S_1') of the instance ΔX.

The vectors $v_0, \ldots, v_{a_{n-1}}$ can be arranged as column vectors of a matrix U.

Since we assumed that the relaxation (S_1') solves the instance ΔX, the matrix $U^T U$ has the form

$$U^T U = \sum \alpha_i s_i s_i^T,$$

where $\alpha_i > 0$ and s_i are characteristic vectors of the solutions of the instance ΔX.

Therefore

$$V^TV = \begin{bmatrix} U^TU & \cdots & U^TU \\ \vdots & & \vdots \\ U^TU & \cdots & U^TU \end{bmatrix} = \sum \alpha_i z_i z_i^T$$

where for each solution X_i whose characteristic vector is s_i, z_i is the characteristic vector of $X_i \cup (X_i + a) \cup \ldots (X_i + (m-1)a)$. ∎

Next we prove that, under certain conditions, the set Y from Theorem 3.1 can be constructed by adding numbers that are not multiples of a single number to the elements of X. Namely we look at the sets Y that have the form

$$Y = X \cup (X + a_1) \cup \ldots \cup (X + a_k),$$

where

$$a_1 \geq 3d_M + 1$$
$$a_i \geq 3a_{i-1} + d_M + 1 \text{ for } i \in \{2, \ldots, k\},$$

and show that the result analogous to Theorem 3.1 holds for these sets under some condition on set X. The proof of that fact is substantially harder than the proof of Theorem 3.1. First we prove three lemmas.

The first lemma shows us how to construct a matrix V from the solutions $X_1, \ldots,$ X_k of an instance ΔX of the turnpike problem, such that the matrix

$$Y = VV^T$$

is feasible for the relaxation (S_1), and therefore (S_1'), of the instance ΔX of the turnpike problem.

Lemma 3.2 *Let X be a set on n elements, $0 \in X$ and let ΔX be the difference set of X. Let $X_1, \ldots, X_k$ be solution sets for the turnpike instance ΔX and let us assume that $0 \in X_i$ for $i \in \{1, \ldots, k\}$.*

Furthermore, let $\{u_1, \ldots, u_k\}$ *be a set of mutually orthogonal vectors in* $\mathbf{R}^k$ *and let*

$$v_i = \sum_{j=1}^{k} \chi_{i,j} u_j, \text{ for } i \in \Delta X'$$

where

$$\chi_{i,j} = \begin{cases} 1, & \text{if the difference } i \text{ is in the solution set } X_j; \\ 0, & \text{otherwise.} \end{cases}$$

Let V *be the matrix whose row vectors are the vectors* v_i, $i \in \Delta X'$. *Then the matrix*

$$Y = VV^T$$

is feasible for the relaxation (S_1) *of the instance* ΔX.

Proof: Note that

$$y_{0,0} = \sum_{i=1}^{k} u_i^2$$

since $0 \in X_i$ for $i \in \{1, \ldots, k\}$.

Let us now look at the constraint for the difference $a \in \Delta X$. We have

$$\begin{aligned} \sum_{\substack{b,c \in \Delta X' \\ c-b=a}} y_{b,c} &= \sum_{\substack{b,c \in \Delta X' \\ c-b=a}} v_b v_c = \\ &= \sum_{\substack{b,c \in \Delta X' \\ c-b=a}} \left(\sum_{i=1}^{k} \chi_{b,i} u_i \right) \left(\sum_{j=1}^{k} \chi_{c,j} u_j \right) = \\ &= \sum_{\substack{b,c \in \Delta X' \\ c-b=a}} \sum_{i=1}^{k} \chi_{b,i} \chi_{c,i} u_i^2 = \\ &= \sum_{i=1}^{k} u_i^2 \sum_{\substack{b,c \in \Delta X' \\ c-b=a}} \chi_{b,i} \chi_{c,i}. \end{aligned}$$

But for every solution set X_i

$$\sum_{\substack{b,c \in \Delta X' \\ c-b=a}} \chi_{b,i}\chi_{c,i} = v(a)$$

because the difference a in the solution X_i must appear $v(a)$ times.

Therefore

$$\sum_{\substack{b,c \in \Delta X' \\ c-b=a}} y_{b,c} = v(a) \sum_{i=1}^{k} u_i^2 =$$

$$= v(a) y_{0,0}$$

Similarly

$$\begin{aligned}
\sum_{b \in \Delta X'} y_{b,c} &= \sum_{b \in \Delta X'} v_b v_c = \\
&= \sum_{b \in \Delta X'} \left(\sum_{i=1}^{k} \chi_{b,b} u_i \right) \left(\sum_{j=1}^{k} \chi_{c,j} u_j \right) = \\
&= \sum_{b \in \Delta X'} \sum_{i=1}^{k} \chi_{b,i}\chi_{c,i} u_i^2 = \\
&= \sum_{i=1}^{k} u_i^2 \chi_{c,i} \sum_{b \in \Delta X'} \chi_{b,i} = \\
&= \sum_{i=1}^{k} u_i^2 \chi_{c,i} n = \\
&= n y_{c,c}.
\end{aligned}$$

The pyramid constraints hold because if two differences b and c are not together in any solution set X_i, then

$$\sum_{i=1}^{k} \chi_{b,i}\chi_{c,i} = 0,$$

from which we can conclude that

$$y_{b,c} = 0,$$

which completes the proof. ∎

Next we show that any matrix of the form $Y = \sum_{i=1}^{k} s_i s_i^T$ where s_i are the characteristic vectors of solutions of the turnpike problem for instance ΔX, can be decomposed as VV^T such that the matrix V is of the form described in Lemma 3.2.

Lemma 3.3 *Let X be a set, $0 \in X$ and let ΔX be the difference set of X. Let Y be a matrix such that $Y = \sum_{i=1}^{k} \lambda_i s_i s_i^T$ where s_i are the characteristic vectors of solutions of the turnpike problem for instance ΔX, and $\lambda_i > 0$, for $i \in \{1, \dots, k\}$. Then there exist a matrix V such that*

$$Y = VV^T$$

and the row vectors v_i, $i \in \Delta X'$ satisfy

$$v_i = \sum_{l=1}^{k} \sqrt{\lambda_i}(s_l)_i u_l.$$

for some orthonormal set of vectors $\{u_1, \dots, u_k\}$.

Proof: Look at

$$\begin{aligned} v_i v_j &= \sum_{l=1}^{k} \sqrt{\lambda_l}(s_l)_i u_l \sum_{m=1}^{k} \sqrt{\lambda_m}(s_m)_j u_m = \\ &= \sum_{l=1}^{k} \sum_{m=1}^{k} \sqrt{\lambda_l}\sqrt{\lambda_m}(s_l)_i (s_m)_j u_l u_m = \\ &= \sum_{l=1}^{k} \lambda_l (s_l)_i (s_l)_j u_l^2 = \\ &= \sum_{l=1}^{k} \lambda_l (s_l)_i (s_l)_j. \end{aligned}$$

But

$$\sum_{l=1}^{k} \lambda_l (s_l)_i (s_l)_j = y_{i,j}.$$

which completes the proof. ∎

In the next lemma we consider the instances ΔX that have the property that every solution contains a point that is not in any other solution. For example, the instances that have unique solutions satisfy this property. We prove that under certain conditions a matrix feasible for the relaxation (S_1') of such instance can be split into two matrices feasible for the same relaxation.

We have:

Lemma 3.4 *Let X be a set and let ΔX be the difference set of X and assume that the instance ΔX has the property that every solution contains a point that is not in any other solutions.*

Let Y be a matrix such that $Y = \sum_{i=1}^{k} \lambda_i s_i s_i^T$ where s_i are the characteristic vectors of solutions of the turnpike problem for instance ΔX, and $\lambda_i > 0$, for $i \in \{1, \dots, k\}$. Furthermore, let

$$Y = VV^T$$

and let v_i, $i \in \Delta X'$ be row vectors of V. If there exist vectors a_i and b_i, for $i \in \Delta X'$ such that

$$v_i = a_i + b_i, \ \text{for } i \in \Delta X'$$

and

$$a_i b_j = 0, \ \text{for } i, j \in \Delta X'$$

and

$$a_i a_j \geq 0,$$
$$b_i b_j \geq 0, \ \text{for } i, j \in \Delta X'.$$

Then the matrices $Y_1 = AA^T$ and $Y_2 = BB^T$, where A is the matrix whose rows are vectors a_i and B is the matrix whose rows are vectors b_i, for $i \in \Delta X'$, are feasible for the relaxation (S_1) of the instance ΔX.

Proof: Because of Lemma 3.3 we can choose V such that its row vectors are

$$v_i = \sum_{l=1}^{k} \sqrt{\lambda_l}(s_l)_i u_l,$$

for some orthonormal set of vectors $\{u_1, \ldots, u_k\}$.

First, notice that if the vectors v_i, $i \in \Delta X'$ satisfy the equation

$$\sum_{i \in \Delta X'} \alpha_i v_i = 0 \tag{3.3}$$

the vectors a_i and vectors b_i satisfy the same equation. i.e

$$\sum_{i \in \Delta X'} \alpha_i a_i = 0 \tag{3.4}$$

and

$$\sum_{i \in \Delta X'} \alpha_i b_i = 0. \tag{3.5}$$

To see this look at

$$\begin{aligned} 0 &= (\sum_{i \in \Delta X'} \alpha_i v_i)^2 = \\ &= (\sum_{i \in \Delta X'} \alpha_i (a_i + b_i))^2 = \\ &= \sum_{i \in \Delta X'} \alpha_i^2 a_i^2 + \sum_{i \in \Delta X'} \alpha_i^2 b_i^2 + 2 \sum_{\substack{i,j \in \Delta X' \\ i<j}} \alpha_i \alpha_j a_i a_j + 2 \sum_{\substack{i,j \in \Delta X' \\ i<j}} \alpha_i \alpha_j b_i b_j = \\ &= (\sum_{i \in \Delta X'} \alpha_i a_i)^2 + (\sum_{i \in \Delta X'} \alpha_i b_i)^2. \end{aligned}$$

For each solution X_i, let x_{s_i} be a number that is in X_i, but not in any other solution set. Let Z be the set of numbers x_{s_i} for $i \in \{1, \ldots, k\}$.

Then the vectors $T = \{v_i | i \in Z\}$ are mutually orthogonal and therefore the vectors $\{a_i | i \in Z\}$ are mutually orthogonal and the vectors $\{b_i | i \in Z\}$ are mutually orthogonal.

Also note that the set T is an orthogonal basis of the vector space spanned by the vectors $\{v_i | i \in \Delta X\}$.

Therefore, the set $\{a_i|i \in Z\}$ is an orthogonal generating set for $\{a_i|i \in \Delta X\}$ and the set $\{b_i|i \in Z\}$ is an orthogonal generating set for $\{b_i|i \in \Delta X\}$.

Because of (3.3), (3.4) and (3.5), if

$$v_i = \sum_{l \in Z} \sqrt{\lambda_l}(s_l)_i v_l,$$

then

$$a_i = \sum_{l \in Z} \sqrt{\lambda_l}(s_l)_i a_l$$

and

$$b_i = \sum_{l \in Z} \sqrt{\lambda_l}(s_l)_i b_l,$$

for any $i \in \Delta X' - Z$.

Now, we can use Lemma 3.2 to complete the proof. ∎

Before we prove the general theorem, let us first look at an example. The exposition of this example can be easily modified into a proof of the theorem.

Let $X = \{0, 1, 4, 6\}$ and let

$$Y = X + (X + 19) + (X + 64),$$

i.e.

$$Y = \{ \begin{array}{llll} 0, & 1, & 4, & 6, \\ 19, & 20, & 23, & 25, \\ 64, & 65, & 68, & 70 \end{array} \}.$$

It can easily be seen that the number 64 appears in the multiset ΔY four times, and so do numbers 45 and 19. The numbers 7,8, ... , 12, 26, 27, ... , 38, 52, 53, ... 57 are not elements of ΔY, and therefore any feasible matrix A for (S_1') has the property that

$$\begin{aligned} a_{7,7} = a_{8,8} = \ldots = a_{12,12} = 0, \\ a_{26,26} = a_{27,27} = \ldots = a_{38,38} = 0, \\ a_{52,52} = a_{53,53} = \ldots = a_{57,57} = 0. \end{aligned} \quad (3.6)$$

But if $a_{i,i} = 0$ then also $a_{70-i,70-i} = 0$ and therefore

$$\begin{aligned} a_{63,63} = a_{62,62} = \ldots = a_{58,58} = 0, \\ a_{44,44} = a_{43,43} = \ldots = a_{32,32} = 0, \\ a_{18,18} = a_{17,17} = \ldots = a_{13,13} = 0. \end{aligned} \tag{3.7}$$

Let

$$A = V^T V$$

and let v_i, $i \in \Delta Y'$ be the column vectors of V. Then because of (3.6) and (3.7)

$$\begin{aligned} v_7 = v_8 = \ldots = v_{18} = 0. \\ v_{26} = v_{27} = \ldots = v_{44} = 0. \\ v_{52} = v_{53} = \ldots = v_{63} = 0. \end{aligned}$$

Next, we look at the equation in (S_1) induced by the differences 64, 45 and 19. We have

$$\sum_{i=0}^{6} x_{i,i+64} = 4 \tag{3.8}$$

and

$$\sum_{i=0}^{6} x_{i,i+45} + \sum_{i=19}^{25} x_{i,i+45} = 4 \tag{3.9}$$

and

$$\sum_{i=0}^{6} x_{i,i+19} + \sum_{i=45}^{51} x_{i,i+19} = 4, \tag{3.10}$$

because of (3.2).

Now, we use (3.8), (3.9) , (3.10), (3.6) and (3.7) to evaluate the following sum

$$\sum_{i=0}^{6}(v_i - v_{i+64})^2 + \sum_{i=0}^{6}(v_{i+19} + v_{i+45} - v_i)^2 + \sum_{i=0}^{6}(v_{i+19} + v_{i+45} - v_{i+64})^2 =$$

$$= \sum_{i=0}^{6} v_i^2 + \sum_{i=64}^{70} v_i^2 - 2\sum_{i=0}^{6} v_i v_{i+64} +$$

$$+ \sum_{i=19}^{25} v_i^2 + \sum_{i=45}^{51} v_i^2 + \sum_{i=0}^{6} v_i^2 - 2\sum_{i=0}^{6} v_i v_{i+19} - 2\sum_{i=0}^{6} v_i v_{i+45} +$$

$$+ \sum_{i=19}^{25} v_i^2 + \sum_{i=45}^{51} v_i^2 + \sum_{i=64}^{70} v_i^2 - 2\sum_{i=19}^{25} v_i v_{i+45} - 2\sum_{i=45}^{51} v_i v_{i+19} =$$

$$= 2\sum_{i=0}^{6} a_{i,i} + 2\sum_{i=19}^{25} a_{i,i} + 2\sum_{i=45}^{51} a_{i,i} + 2\sum_{i=64}^{70} a_{i,i} -$$

$$- 2\sum_{i=0}^{6} a_{i,i+19} - 2\sum_{i=45}^{51} a_{i,i+19} -$$

$$- 2\sum_{i=0}^{6} a_{i,i+45} - 2\sum_{i=19}^{25} a_{i,i+45} -$$

$$- 2\sum_{i=0}^{6} a_{i,i+64} =$$

$$= 24 - 2 \cdot 3 \cdot 4 = 0$$

Since we started with a positive expression, we can conclude that

$$v_i = v_{i+64}, \text{ for } i \in \{0, \ldots, 6\} \tag{3.11}$$

$$v_{i+19} + v_{i+45} = v_i, \text{ for } i \in \{0, \ldots, 6\} \tag{3.12}$$

Next we prove that for any $i, j \in \{0, \ldots 6\}$,

$$v_{i+19} v_{j+45} = 0.$$

We need that

$$\sum_{i=0}^{6} v_i = 4v_0.$$

We see this by observing that

$$12 = \sum_{i=0}^{70} v_i^2 = \sum_{i=0}^{6} v_i^2 + \sum_{i=19}^{25} (v_i + v_{i+26})^2 + \sum_{i=64}^{70} v_i^2$$
$$= 3\sum_{i=0}^{6} v_i^2,$$

because of equalities (3.11) and (3.12). Therefore

$$\sum_{i=0}^{6} a_{i,i} = 4.$$

Next we prove that

$$\sum_{i=0}^{6} v_i = 4v_0. \tag{3.13}$$

Again, we have

$$(\sum_{i=0}^{6} v_i - 4v_0)^2 = \sum_{i=0}^{6} v_i^2 + 2 \sum_{\substack{i,j=0 \\ i<j}}^{6} v_i v_j - 8\sum_{i=0}^{6} v_i v_0 + 16 =$$
$$= -7\sum_{i=0}^{6} v_i^2 + \sum_{i=0}^{6}\sum_{j=65}^{70} v_i v_j + 16$$
$$= -7 \cdot 4 + 12 + 16 =$$
$$= 0.$$

because of equalities (3.11).

From (3.13) and (3.11) we have

$$\sum_{i=64}^{70} v_i = 4v_0,$$

and therefore

$$\sum_{i=19}^{25} v_i + \sum_{i=45}^{51} v_i = 4v_0.$$

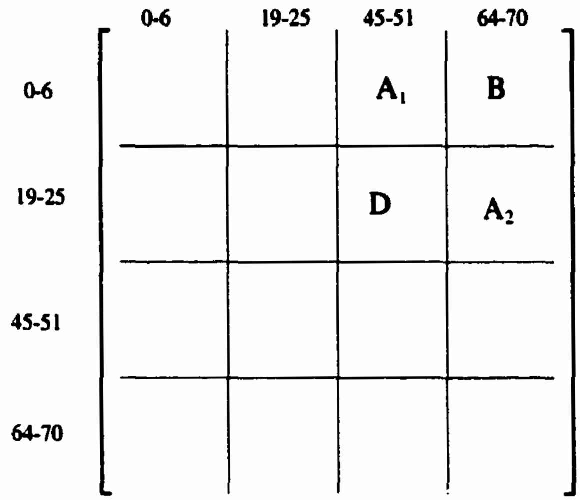

Figure 3.1: The form of a matrix feasible for the relaxation (S_1) of the instance ΔY.

Any feasible matrix for the relaxation (S_1) of the instance ΔY has the form shown in Figure 3.1

If

$$\sum_{i=19}^{25} a_{i,i} = \alpha \tag{3.14}$$

and

$$\sum_{i=45}^{51} a_{i,i} = 4 - \alpha = \beta, \tag{3.15}$$

then the entries of A in the submatrix A_1 sum to 4β and the entries in the submatrix A_2 sum to 4α.

This is because

$$\begin{aligned}\sum_{i=0}^{6}\sum_{j=45}^{51} a_{i,j} &= \sum_{i=0}^{6}\sum_{j=45}^{51} v_i v_j \\ &= \sum_{j=45}^{51} v_j \sum_{i=0}^{6} v_i = \\ &= \sum_{j=45}^{51} 4 v_j v_0 = \\ &= 4\sum_{j=45}^{51} v_j v_0 = \\ &= 4\sum_{j=45}^{51} a_{j,j} = \\ &= 4\beta.\end{aligned}$$

Similar equations can be written for $\sum_{i=19}^{25}\sum_{j=64}^{70} a_{i,j}$.
Also

$$\begin{aligned}\sum_{i=0}^{6}\sum_{j=64}^{70} a_{i,j} &= \sum_{i,j=0}^{6} v_i v_j = \\ &= 10.\end{aligned} \tag{3.16}$$

Now, observe that in ΔY there are 26 numbers that are greater than or equal to 45. The variables associated with these differences are in the submatrices A_1, A_2, B, and D. But because of (3.14), (3.15) and (3.16) the entries in the submatrices A_1, A_2, and C sum to

$$4\alpha + 4\beta + 10 = 26.$$

Therefore, the entries of the submatrix D are 0 and we have proved that

$$v_{i+19} v_{j+45} = 0, \tag{3.17}$$

and the vectors $\{v_0, \ldots, v_6\}$, $\{v_{19}, \ldots, v_{25}\}$ and $\{v_{45}, \ldots, v_{51}\}$ satisfy the conditions of Lemma 3.4.

It is easy to check that any matrix Z feasible for the relaxation (S_1) of the instance ΔX is of the form

$$Z = \alpha_1 s_1 s_1^T + \alpha_2 s_2 s_2^T,$$

where $\alpha_1, \alpha_2 \geq 0$ and s_1, s_2 are characteristic vectors of the solutions of the instance, and that s_1 is the mirror image of s_2, i.e. that the instance ΔX has only one solution.

Therefore if V_1 is the matrix whose row vectors are $\{v_{19}, \ldots, v_{25}\}$ and V_2 is the matrix whose row vectors are $\{v_{45}, \ldots, v_{51}\}$, then

$$V_1 V_1^T = \beta_1 s_1 s_1^T + \beta_2 s_2 s_2^T$$

and

$$V_2 V_2^T = \gamma_1 s_1 s_1^T + \gamma_2 s_2 s_2^T$$

because of Lemma 3.4.

Because of (3.17) matrix A can be split into two matrices Y_1 and Y_2 feasible for (S_1), such that

$$Y_1 = \begin{bmatrix} v_{19} \\ \vdots \\ v_{25} \\ v_{19} \\ \vdots \\ v_{25} \\ 0 \\ \vdots \\ 0 \\ v_{19} \\ \vdots \\ v_{25} \end{bmatrix} [v_{19}, \ldots v_{25}, v_{19}, \ldots, v_{25}, 0, \ldots, 0, v_{19}, \ldots, v_{25}]$$

and

$$Y_2 = \begin{bmatrix} v_{45} \\ \vdots \\ v_{51} \\ 0 \\ \vdots \\ 0 \\ v_{45} \\ \vdots \\ v_{51} \\ v_{45} \\ \vdots \\ v_{51} \end{bmatrix} [v_{45}, \dots v_{51}, 0, \dots, 0, v_{45}, \dots, v_{51}, v_{45}, \dots, v_{51}] =$$

Now, the vectors $\{v_{19}, \dots, v_{25}\}$ represent a combination of solutions X_1 and X_2 for the instance ΔX, and therefore the matrix Y_1 represents the combination of solutions $X_1 \cup (X_1 + 19) \cup (X_1 + 64)$ and $X_2 \cup (X_2 + 19) \cup (X_2 + 64)$. Similar statement can be written for the matrix Y_2.

Now we state and prove the general theorem:

Theorem 3.5 *Let X be the set of cardinality n and let $0 \in X$. Let a and be be such that*

$$a \geq 3d_M + 1$$

and

$$b \geq 3a + d_M + 1,$$

where d_M is the largest element of ΔX.

Furthermore, let

$$Y = X \cup (X + a) \cup (X + b).$$

Then if the instance ΔX is solved by its relaxation (S_1'), and has the property that its every solution contains a point that is not contained in any other solution, then the instance ΔY is solved by its relaxation (S_1').

Proof: Let A be a feasible matrix for the relaxation (S_1) of the instance ΔY and let

$$A = VV^T,$$

and let v_i, $i \in \Delta Y'$ be the row vectors of V.

Notice that the numbers

$$\begin{aligned} &d_M + 1, d_M + 2, \dots, a - d_M - 1, \\ &a + d_M + 1, a + d_M + 2, \dots, b - a - d_M - 1, \\ &d_M + b - a + 1, d_M + b - a + 2, \dots, b - d_M - 1 \end{aligned}$$

do not appear in ΔY and hence

$$\begin{aligned} &v_{d_M+1} = v_{d_M+2} = \dots = v_{a-d_M-1} = 0, \\ &v_{a+d_M+1} = v_{a+d_M+2} = \dots = v_{b-a-d_M-1} = 0 \\ &v_{d_M+b-a+1} = v_{d_M+b-a+2} = \dots = v_{b-d_M-1} = 0 \end{aligned}$$

and

$$\begin{aligned} &v_{b-1} = v_{b-2} = \dots = v_{2d_M+b-a+1} = 0, \\ &v_{b-a-1} = v_{b-a-2} = \dots = v_{2d_M+a+1} = 0 \\ &v_{a-1} = v_{a-2} = \dots = v_{2d_M+1} = 0. \end{aligned}$$

So, because of the conditions on a and b, the only non-zero vectors v_i of V are a subset of

$$\begin{aligned} &v_0, v_1, \dots, v_{d_M}, \\ &v_a, v_{a+1}, \dots, v_{a+d_M}, \\ &v_{b-a}, v_{b-a+1}, \dots, v_{b-a+d_M}, \\ &v_b, v_{b+1}, \dots, v_{b+d_M}. \end{aligned}$$

Also, the differences a, $b - a$ and b appear in the multiset ΔY exactly n times each.

Therefore

$$\sum_{i=0}^{d_M} v_i v_{i+b} = n, \tag{3.18}$$

and

$$\sum_{i=0}^{d_M+b-a} v_i v_{i+a} = \sum_{i=0}^{d_M} v_i v_{i+a} + \sum_{i=b-a}^{d_M+b-a} v_i v_{i+a} = n, \tag{3.19}$$

and

$$\sum_{i=0}^{d_M+a} v_i v_{i+b-a} = \sum_{i=0}^{d_M} v_i v_{i+b-a} + \sum_{i=a}^{d_M+a} v_i v_{i+b-a} = n. \tag{3.20}$$

We now present two crucial parts of the proof which ensure that Lemma 3.4 can be invoked and the matrix A split as in the above example. We prove that

$$v_i = v_{i+b}, \text{ for } i \in \Delta X' \tag{3.21}$$

$$v_{i+a} + v_{i+b-a} = v_i, \text{ for } i \in \Delta X' \tag{3.22}$$

and

$$v_{i+a} v_{j+b-a} = 0, \text{ for } i, j \in \Delta X'. \tag{3.23}$$

To prove (3.21) and (3.22) let us look at

$$\sum_{i=0}^{d_M}(v_i - v_{i+b})^2 + \sum_{i=0}^{d_M}(v_{i+a} + v_{i+b-a} - v_i)^2 + \sum_{i=0}^{d_M}(v_{i+a} + v_{i+b-a} - v_{i+b})^2 =$$

$$= \sum_{i=0}^{d_M} v_i^2 + \sum_{i=b}^{d_M+b} v_i^2 - 2\sum_{i=0}^{d_M} v_i v_{i+b} +$$

$$+ \sum_{i=a}^{d_M+a} v_i^2 + \sum_{i=b-a}^{d_M+b-a} v_i^2 + \sum_{i=0}^{d_M} v_i^2 - 2\sum_{i=0}^{d_M} v_i v_{a+i} - 2\sum_{i=0}^{d_M} v_i v_{i+b-a} +$$

$$+ \sum_{i=a}^{d_M+a} v_i^2 + \sum_{i=b-a}^{d_M+b-a} v_i^2 + \sum_{i=b}^{d_M+b} v_i^2 - 2\sum_{i=a}^{d_M+a} v_i v_{i+b-a} - 2\sum_{i=b-a}^{d_M+b-a} v_i v_{i+a} =$$

$$= 2\left(\sum_{i=0}^{d_M} v_i^2 + \sum_{i=b}^{d_M+b} v_i^2 + \sum_{i=a}^{d_M+a} v_i^2 + \sum_{i=b-a}^{d_M+b-a} v_i^2\right) -$$

$$- 2\sum_{i=0}^{d_M} v_i v_{i+b} -$$

$$\begin{aligned}
&-2\sum_{i=0}^{d_M} v_i v_{a+i} - 2\sum_{i=b-a}^{d_M+b-a} v_i v_{i+a} - \\
&-2\sum_{i=0}^{d_M} v_i v_{i+b-a} - 2\sum_{i=a}^{d_M+a} v_i v_{i+b-a} = \\
&= 2\cdot 3\cdot n - 2n - 2n - 2n \\
&= 0
\end{aligned}$$

because (3.18), (3.19), (3.20) and the fact that $b-2a$ is not in ΔY, which completes the proof of (3.21) and (3.22).

To prove (3.23) notice that in ΔY there are

$$n^2 + \binom{n}{2}$$

numbers bigger than $b-a$, because there are n^2 numbers obtained as $x_i - x_j$ where $x_i \in X+b$ and $x_j \in X$ and there are $\binom{n}{2}$ numbers in the range $b \ldots b+d_M$.

Now, the matrix A looks like the matrix on Figure 3.1. Similarly as in the above example, we can show that the entries of the submatrices A_1, A_2 and B sum to $n^2+\binom{n}{2}$ and therefore the entries of D sum to 0. This proves (3.23). ■

The statement of the theorem also holds if $a=1$, which can be shown in the same way.

Next we prove that we can add more than two different numbers to the elements of the set X in order to obtain a set similar to the set Y from Theorem 3.5 that satisfies the claim of that theorem.

We have

Theorem 3.6 *Let X be a set of cardinality n and let $0 \in X$. Let a_i, $i \in \{1,\ldots,k\}$ be numbers such that*

$$a_1 \geq 3d_M + 1$$

and

$$a_i \geq 3a_{i-1} + d_M + 1 \textit{ for } i \in \{2,\ldots,k\},$$

where d_M is the largest element of ΔX.

Furthermore, let

$$Y = X \cup (X + a_1) \cup \ldots \cup (X + a_k).$$

Then if the instance ΔX is solved by its relaxation (S_1'), and has the property that its every solution contains a point that is not contained in any other solution, then the instance ΔY is solved by its relaxation (S_1').

Proof: The proof is by induction on k.

When $k = 1$ the statement follows from Theorem 3.1.

When $k = 2$ the statement follows from Theorem 3.5. Let us assume that the statement of the theorem holds for any set Y that is a union of k sets

$$Y = X \cup (X + a_1) \cup \ldots \cup (X + a_{k-1}).$$

and assume that a set Y is a union of $k + 1$ sets.

Let A be a feasible matrix for the relaxation (S_1) of the instance ΔY and let

$$A = VV^T,$$

and let v_i, $i \in \Delta Y'$ be the row vectors of V.

By careful examination of ΔY we can see that the only vectors v_i, $i \in \Delta Y'$ that can be different from the null vector are

$$\begin{array}{l}
v_0, v_1, \ldots, v_{d_M}, \\
v_{a_1}, v_{a_1+1}, \ldots, v_{a_1+d_M}, \\
v_{a_2}, v_{a_2+1}, \ldots, v_{a_2+d_M}, \\
\vdots \\
v_{a_k}, v_{a_k+1}, \ldots, v_{a_k+d_M}, \\
v_{a_k-a_{k-1}}, v_{a_k-a_{k-1}+1}, \ldots, v_{a_k-a_{k-1}+d_M}, \\
v_{a_k-a_{k-2}}, v_{a_k-a_{k-2}+1}, \ldots, v_{a_k-a_{k-2}+d_M}, \\
\vdots \\
v_{a_k-a_1}, v_{a_k-a_1+1}, \ldots, v_{a_k-a_1+d_M}.
\end{array} \tag{3.24}$$

Also, the differences $a_i - a_j$, $i > j$, appear in ΔY exactly n times and, because of (3.24), in the relaxation (S_1) for the instance ΔY the equation corresponding to the difference $a_i - a_j$ is

$$\sum_{l=a_j}^{a_j+d_M} v_l v_{l+a_i-a_j} + \sum_{l=a_k-a_j}^{a_k-a_j+d_M} v_l v_{l+a_i-a_j} = n. \tag{3.25}$$

Next we prove that

$$v_i = v_{i+a_k}, \text{ for } i \in \Delta X' \tag{3.26}$$

and

$$\begin{aligned} v_{i+a_1} &= v_{i+a_l}, \text{ for } i \in \Delta X', l \in \{2, \dots, k-1\} \\ v_{i+a_k-a_1} &= v_{i+a_k-a_l}. \end{aligned} \tag{3.27}$$

If k is odd, we look at the following sum

$$\begin{aligned} &\sum_{i=0}^{d_M}((v_i - v_{i+a_k})^2 + (v_{i+a_1} - v_{i+a_2})^2 + \dots + (v_{i+a_{k-2}} - v_{i+a_{k-1}})^2 + \\ &+ (v_{i+a_k-a_1} - v_{i+a_k-a_2})^2 + \dots + (v_{i+a_k-a_{k-2}} - v_{i+a_k-a_{k-1}})^2) = \\ &= \sum_{i=0}^{a_k+d_M} (v_i^2 - 2(v(a_k) + v(a_2 - a_1) + \dots + v(a_{k-1} - a_{k-2}))) = \\ &= (k+1)n - 2(n + \frac{k-1}{2}n) = \\ &= 0 \end{aligned}$$

which proves (3.26) and (3.27).

If k is even, similarly we can calculate that the following sum is 0:

$$\begin{aligned} &\sum_{i=0}^{d_M}(2(v_i - v_{i+a_k})^2 + (v_{i+a_1} - v_{i+a_2})^2 + (v_{i+a_2} - v_{i+a_3})^2 + \dots \\ &\dots + (v_{i+a_{k-2}} - v_{i+a_{k-1}})^2 + (v_{i+a_{k-1}} - v_{i+a_1})^2 + \\ &+ (v_{i+a_k-a_1} - v_{i+a_k-a_2})^2 + (v_{i+a_k-a_2} - v_{i+a_k-a_3})^2 + \dots \\ &\dots + (v_{i+a_k-a_{k-2}} - v_{i+a_k-a_{k-1}})^2 + (v_{i+a_k-a_{k-1}} - v_{i+a_k-a_1})^2) = 0. \end{aligned}$$

Similarly, we can prove that for $l \in \Delta X'$ and $i \in \{1, \dots, k-1\}$

$$v_{l+a_i} + v_{l+a_k-a_i} = a_l. \tag{3.28}$$

Now, we look at the submatrix B of A indexed by the differences $i \in \Delta Y'$ such that $v_i \neq 0$ (as in (3.24)) other than

$$v_{a_1}, v_{a_1+1}, \dots, v_{a_1+d_M},$$

$$v_{a_k-a_1}, v_{a_k-a_1+1}, \dots, v_{a_k-al+d_M}.$$

Because of (3.26), (3.27) and (3.28) we can conclude that B satisfies all the constraints of the relaxation (S_1') of the instance

$$X \cup (X + a_2) \cup \dots \cup (X + a_k).$$

Now, we can invoke the induction hypothesis to conclude that this instance is solved properly by its relaxation (S_1'). Now, it follows directly that the instance ΔY is solved by its relaxation (S_1') because from (3.27) we have

$$v_{i+a_1} = v_{i+a_2}, \text{ for } i \in \Delta X'$$

which completes the proof. ∎

3.3 Zhang's instances

In [35] Zhang constructed a class of instances for which Skiena, Smith and Lemke's backtracking procedure takes exponential time to find a solution. The instances have unique solutions and are difference sets of the sets A defined in the following way.

Let $0 < \epsilon < \frac{1}{12}n$. Let

$$\begin{aligned}
A_2 &= \{\epsilon, 2\epsilon, \dots, n\epsilon, \} \\
A_3 &= \{(n+1)\epsilon, (n+2)\epsilon, \dots, 2n\epsilon\}, \\
A_4 &= \{(2n+1)\epsilon, (2n+2)\epsilon, \dots, 3n\epsilon\}, \\
A_5 &= \{1 - 3n\epsilon, \dots, 1 - (2n+2)\epsilon, 1 - (2n+1)\epsilon\}. \\
A_1 &= \{1 - n\epsilon, \dots, 1 - 2\epsilon, 1 - \epsilon\},
\end{aligned}$$

Let F and G be disjoint sets such that $A_3 = F \cup G^*$, and let $D = F \cup G$, where $G^* = \{1 - g | g \in G\}$. Let $A = A_1 \cup A_2 \cup A_4 \cup A_5 \cup D \cup \{0, 1\}$.

Zhang [35], proved the following proposition

Proposition 3.7 *With the above notation, we can choose D such that, giving ΔA to the Skiena at al. backtracking algorithm, it takes at least $\Omega(2^{n-1})$ time to find A.*

We prove that the relaxation (S_1') solves these instances. We need the following lemma:

Lemma 3.8 *Let a_i, $i = 1, \ldots, 4$, and x be vectors in $\mathbf{R}^n$ such that*

$$a_1 + a_2 = x,$$
$$a_3 + a_4 = x,$$
$$a_i a_i = a_i x \text{ for } i = 1, \ldots, 4.$$

If $a_1 a_3 + a_2 a_4 = x^2$ then $a_1 = a_3$ and $a_2 = a_4$. If $a_1 a_3 = 0$ and $a_2 a_4 = 0$ then $a_1 = a_4$ and $a_2 = a_3$.

Proof: From

$$\begin{aligned} x^2 &= a_1 a_3 + a_2 a_4 = a_1 a_3 + (x - a_1)(x - a_3) = \\ &= x^2 - a_1^2 - a_3^2 + 2a_1 a_3, \end{aligned}$$

we have

$$(a_1 - a_3)^2 = 0,$$

and therefore $a_1 = a_3$. Similarly we can show $a_2 = a_4$, which proves the first statement.

If $a_1 a_3 = 0$ and $a_2 a_4 = 0$ then

$$x^2 = (a_1 + a_2)(a_3 + a_4) = a_1 a_4 + a_2 a_3,$$

and we can use the first part of the lemma to conclude that $a_1 = a_4$ and $a_2 = a_3$. ■

Now we are ready to prove the main result of this section, i.e. that the relaxation S_1' defined in the previous section, of the instance ΔA solves the turnpike problem on that instance.

Theorem 3.9 *For the above defined sets A, the instance ΔA of the turnpike problem can be solved by its relaxation (S_1').*

Proof: First we list some properties of ΔA:

(a)
$$\begin{aligned}
A_1 - A_2 &\subset [1-2n\epsilon, 1-2\epsilon],\\
A_1 - A_3 &\subset [1-3n\epsilon, 1-(n+2)\epsilon],\\
A_1 - A_4 &\subset [1-4n\epsilon, 1-(2n+2)\epsilon],\\
A_1 - A_5 &\subset [(n+1)\epsilon, (3n-1)\epsilon],\\
A_3 - A_2 &\subset [\epsilon, (2n-1)\epsilon],\\
A_4 - A_2 &\subset [(n+1)\epsilon, (3n-1)\epsilon],\\
A_5 - A_2 &\subset [1-4n\epsilon, 1-(2n+2)\epsilon],\\
A_4 - A_3 &\subset [\epsilon, (2n-1)\epsilon],\\
A_5 - A_3 &\subset [1-5n\epsilon, 1-(3n+2)\epsilon],\\
A_5 - A_4 &\subset [1-6n\epsilon, 1-(4n+2)\epsilon],\\
A_1 - A_3^* &\subset [\epsilon, (2n-1)\epsilon],\\
A_3^* - A_2 &\subset [1-3n\epsilon, 1-(n+2)\epsilon],\\
A_3^* - A_4 &\subset [1-5n\epsilon, 1-(3n+2)\epsilon],\\
A_3^* - A_1 &\subset [\epsilon, (2n-1)\epsilon],
\end{aligned}$$

(b) The numbers in ΔA are of the form $k\epsilon$, where k is a non-negative integer less or equal to $3n$, or $1-k\epsilon$, where k is a nonnegative integer less or equal to $6n$. Therefore, in ΔA there are no numbers in the interval $(3n\epsilon, 1-6n\epsilon)$.

(c) The k-th largest element of A_1, $1-k\epsilon$, appears in ΔA $k+1$ times.

Proof: The number $1-k\epsilon$ is the difference of the following numbers:

$$\begin{array}{ll}
1-k\epsilon & -0,\\
1-(k-1)\epsilon & -\epsilon,\\
\vdots & \vdots\\
1-\epsilon & -(k-1)\epsilon,\\
1 & -k\epsilon,
\end{array}$$

where the first number of the difference is an element of A_1 or 1, and the second is an element of A_2 or 0. ∎

(d) The k-th smallest element of A_4, $(2n+k)\epsilon$, appears in ΔA $2(n-k+1)$-times.

Proof: The number $(2n+k)\epsilon$, is the difference of the following numbers:

$$\begin{array}{ll}
(2n+k)\epsilon & -0, \\
(2n+k+1)\epsilon & -\epsilon, \\
(2n+k+2)\epsilon & -2\epsilon, \\
\vdots & \vdots \\
3n\epsilon & -(n-k)\epsilon, \\
1-(n-k)\epsilon & -1-3n\epsilon \\
\vdots & \vdots \\
1-2\epsilon & -(1-(2n+k+2)\epsilon), \\
1-\epsilon & -(1-(2n+k+1)\epsilon), \\
1 & -(1-(2n+k)\epsilon).
\end{array}$$

The first number in the above differences is from A_4 or 1, and the second is from A_2 or 0, or the first number is from A_1 and the second is from A_5. ∎

(e) The k-th smallest element of A_3^*, $1-(2n-k+1)\epsilon$ appears in ΔA $n+1$-times.

Proof: The number $1-(2n-k+1)\epsilon$ is the difference of the following numbers:

$$\begin{array}{ll}
1-n\epsilon & -(n-k+1)\epsilon, \\
1-(n-1)\epsilon & -(n-k+2)\epsilon, \\
\vdots & \vdots \\
1-(n-k+1)\epsilon & -n\epsilon,
\end{array}$$

where the first number is in A_1 and the second number is in A_2. In this way, we represented the number $1-(2n-k+1)\epsilon$ as a difference of k pairs of numbers. To get the additional $n-k+1$ pairs, depending on the partition of A_3, we can represent $1-(2n-k+1)\epsilon$ as a difference of

$$\begin{array}{lclclcl} 1-(2n-k+1)\epsilon & - & 0 & \text{or} & 1 & - & (2n-k+1)\epsilon, \\ 1-(2n-k)\epsilon & - & \epsilon & \text{or} & 1-\epsilon & - & (2n-k)\epsilon, \\ \vdots & & \vdots & & \vdots & & \vdots \\ 1-(n+1)\epsilon & - & (n-k)\epsilon & \text{or} & 1-(n-k)\epsilon & - & (n+1)\epsilon. \end{array}$$

■

(f) The number of differences in ΔA that are between ϵ and 2ϵ or between 1 and $1-(2n+3)\epsilon$ depends on the partition of A_3. In particular the number of elements of A_5, that are in ΔA depends on the partition of A_3, with the exception of number of the differences $1-(2n+1)\epsilon$ and $1-(2n+2)\epsilon$. The difference $1-(2n+1)\epsilon$ appears in ΔA $n+2$-times, as the difference of the following numbers

$$\begin{array}{lclclcl} 1-2n\epsilon & - & 0 & & & & \\ 1 & - & (2n+1)\epsilon & & & & \\ 1-2n\epsilon & - & \epsilon & \text{or} & 1-\epsilon & - & 2n\epsilon \\ 1-(2n-1)\epsilon & - & 2\epsilon & \text{or} & 1-2\epsilon & - & (2n-1)\epsilon \\ \vdots & & & & \vdots & & \\ 1-(n+1)\epsilon & - & n\epsilon & \text{or} & 1-n\epsilon & - & (n+1)\epsilon. \end{array}$$

Similarly, we can see that the difference $1-(2n+2)\epsilon$ appears in ΔA $n+3$ times.

■

Now, let X be a feasible matrix for the relaxation (S_1') of the instance ΔA, and

$$X = VV^T$$

and let v_i be the row vectors of V, for $i \in \Delta A$.

Because of (b), for the differences i in ΔA, that are in the range $(3n\epsilon, 1-6n\epsilon)$ or $(6n\epsilon, 1-3n\epsilon)$, we have $v_i = 0$. Note that because of the condition that $n < \frac{1}{12}n$, the intervals $(3n\epsilon, 1-6n\epsilon)$ and $(6n\epsilon, 1-3n\epsilon)$ form a continuous interval $(3n\epsilon, 1-3n\epsilon)$.

Because of (c), in the relaxation (S_1') for the numbers in A_1 we have the following constraints:

$$\begin{aligned} x_{0,1-\epsilon} + x_{\epsilon,1} &= 2, \\ x_{0,1-2\epsilon} + x_{\epsilon,1-\epsilon} + x_{2\epsilon,1} &= 3, \\ &\vdots \\ x_{0,1-n\epsilon} + x_{\epsilon,1-(n-1)\epsilon} + x_{2\epsilon,1-(n-2)\epsilon} + \cdots + x_{(n-1)\epsilon,1-\epsilon} + x_{n\epsilon,1} &= n+1. \end{aligned}$$

We can therefore conclude that

$$\begin{aligned} &v_\epsilon = v_{1-\epsilon} = v_0, \\ &v_{2\epsilon} = v_{1-2\epsilon} = v_0, \\ &\vdots \\ &v_{n\epsilon} = v_{1-n\epsilon} = v_0. \end{aligned} \tag{3.29}$$

Because of (d) for numbers in A_4 we get:

$$\begin{aligned} x_{0,3n\epsilon} + x_{1-3n\epsilon,1} &= 2, \\ x_{0,(3n-1)\epsilon} + x_{\epsilon,3n\epsilon} + x_{1-\epsilon,1-3n\epsilon} + x_{1,1-(3n-1)\epsilon} &= 4, \\ &\vdots \\ x_{0,(2n+1)\epsilon} + x_{\epsilon,(2n+2)\epsilon} + x_{2\epsilon,(2n+3)\epsilon} + \cdots + x_{1-(2n+2)\epsilon,1-\epsilon} + x_{1,1-(2n+1)\epsilon} &= 2(n+1). \end{aligned} \tag{3.30}$$

We can combine (3.29) and (3.30) to conclude that

$$\begin{aligned} &v_{3n\epsilon} = v_{1-3n\epsilon} = v_0, \\ &v_{(3n-1)\epsilon} = v_{1-(3n-1)\epsilon} = v_0, \\ &\vdots \\ &v_{(2n+1)\epsilon} = v_{1-(2n+1)\epsilon} = v_0. \end{aligned} \tag{3.31}$$

Because of (e), (3.29) and (3.31) for the numbers in A_3^* we get:

$$\begin{aligned} x_{0,1-(n+1)\epsilon} + x_{(n+1)\epsilon,1} &= 1, \\ x_{0,1-(n+2)\epsilon} + x_{(n+2)\epsilon,1} &= 1, \\ &\vdots \\ x_{0,1-2n\epsilon} + x_{2n\epsilon,1} &= 1. \end{aligned} \tag{3.32}$$

Now, let us look at the equations we have for the elements of A_5. Because of (f) we have:

$$\begin{aligned} &x_{0,1-(2n+1)\epsilon} + x_{\epsilon,1-2n\epsilon} + x_{2\epsilon,(2n-1)\epsilon} + \cdots + x_{2n\epsilon,1-\epsilon} + x_{(2n+1)\epsilon,1} = n+2, \\ &x_{0,1-(2n+2)\epsilon} + x_{\epsilon,1-(2n+1)\epsilon} + x_{2\epsilon,2n\epsilon} + \cdots + x_{(2n+1)\epsilon,1-\epsilon} + x_{(2n+2)\epsilon,1} = n+3, \\ &x_{0,1-(2n+3)\epsilon} + x_{\epsilon,1-(2n+2)\epsilon} + x_{2\epsilon,(2n+1)\epsilon} + \cdots + x_{(2n+2)\epsilon,1-\epsilon} + x_{(2n+3)\epsilon,1} = \\ &\qquad = v(1-(2n+3)\epsilon), \\ &\vdots \\ &x_{0,1-3n\epsilon} + x_{\epsilon,1-(3n-1)\epsilon} + x_{2\epsilon,(3n-2)\epsilon} + \cdots + x_{(3n-1)\epsilon,1-\epsilon} + x_{3n\epsilon,1} = v(1-3n\epsilon). \end{aligned}$$

Using (3.29), (3.31) and (3.32), we have

$$\begin{aligned} &x_{(n+1)\epsilon,1-(n+1)\epsilon} = 0, \\ &x_{(n+1)\epsilon,1-(n+2)\epsilon} + x_{(n+2)\epsilon,1-(n+1)\epsilon} = v(1-(2n+3)\epsilon) - (n-2) - 6, \\ &x_{(n+1)\epsilon,1-(n+3)\epsilon} + x_{(n+2)\epsilon,1-(n+2)\epsilon} + x_{(n+3)\epsilon,1-(n+1)\epsilon} = v(1-(2n+4)\epsilon) - (n-3) - 8, \\ &\vdots \\ &x_{(n+1)\epsilon,1-(2n-1)\epsilon} + x_{(n+2)\epsilon,1-(2n-2)\epsilon} + \cdots + x_{(2n-1)\epsilon,1-(n+1)\epsilon} = v(3n\epsilon) - 2n - 1, \end{aligned} \tag{3.33}$$

Now, we can look at the equations

$$\begin{aligned} x_{0,1-(n+1)\epsilon} + x_{(n+1)\epsilon,1} &= 1, \\ x_{0,1-(n+2)\epsilon} + x_{(n+2)\epsilon,1} &= 1 \end{aligned}$$

from (3.32) and the equation

$$x_{(n+1)\epsilon,1-(n+2)\epsilon} + x_{(n+2)\epsilon,1-(n+1)\epsilon} = v(1-(2n+3)\epsilon) - (n-2) - 6$$

from (3.33). We can see that if the differences $(n + 1)\epsilon$ and $(n + 2)\epsilon$ are either both in F or both in G^*, then $v(1 - (2n + 3)\epsilon) - (n - 2) - 6 = 0$ and otherwise $v(1 - (2n + 3)\epsilon) - (n - 2) - 6 = 1$.

We can now apply Lemma 3.8 to conclude that $v_{(n+1)\epsilon} = v_{(n+2)\epsilon}$ and $v_{1-(n+1)\epsilon} = v_{1-(n+2)\epsilon}$ in the first case and $v_{(n+1)\epsilon} = v_{1-(n+2)\epsilon}$ and $v_{1-(n+1)\epsilon} = v_{(n+2)\epsilon}$ in the second case.

Now, we can plug in the obtained values for the vectors $v_{(n+1)\epsilon}$, $v_{(n+2)\epsilon}$, $v_{1-(n+1)\epsilon}$, $v_{1-(n+2)\epsilon}$ into the third equation in (3.33) and apply the lemma again.

Working our way from top, using the Lemma 3.8 we can conclude that the vectors assigned to each number of F are identical and that the vectors assigned to each number of G are identical. ∎

3.4 Examples that can be solved by the backtracking procedure in polynomial time

In this section we assume that the multiset ΔX is a set, i.e. that all the differences are different, and the instance ΔX has a unique solution.

We say that the Skiena's et al. algorithm, described in Chapter 1, has order k on an instance ΔX of the turnpike problem, if k is the maximum number of steps the algorithm backtracks, i.e. if we assume that an element a is in a solution set X that the algorithm is currently constructing, we need to put $k - 1$ more elements in the set X before we can conclude that the assumption that $a \in X$ was incorrect.

We say that the backtracking algorithm has order 0, if at any execution step we can conclude that either a or $d_M - a$ is in the solution set X that is currently being constructed, for a the largest difference that is in the difference set ΔX, but is not in the difference set of the partial solution set X, and d_M is the largest element of ΔX.

In this section we prove that if for an instance ΔX of the turnpike problem that has a unique solution and all the numbers in ΔX are different, the backtracking procedure has order k, then the instance ΔX is solved by the relaxation (S_{k+1}).

This result is not surprising because the relaxation (S_{k+1}) operates with the $(k+1)$-tuples of differences from $\Delta X'$ and therefore has the capability to see k steps ahead in the backtracking procedure.

The following theorem is also important because for the above described instances, if k is a constant that does not depend on the size of the instance, our relaxation has polynomial size and therefore also runs in polynomial time.

First we prove the following lemma:

Lemma 3.10 *Let Y be a feasible matrix for the relaxation (S_k) of the instance ΔX of the turnpike problem. Let*

$$Y = VV^T,$$

and let $v_{d_{i_1},\ldots,d_{i_k}}$ be the row vectors of V, for any proper index $\{d_{i_1}\ldots d_{i_k}\}$. If for any two differences $d_a, d_b \in \Delta X'$

$$v_{0\ldots 0d_a} = v_{0\ldots 0d_b}, \tag{3.34}$$

then

$$v_{0\ldots 0d_a d_b} = v_{0\ldots 0d_b}$$

and

$$v_{d_{i_1}\ldots d_{i_{k-1}}d_a} = v_{d_{i_1}\ldots d_{i_{k-1}}d_b}$$

for any proper index $\{d_{i_1}\ldots d_{i_{k-1}}\}$.

Proof: The lemma follows from the mixing constraints for the relaxation (S_k). Namely, we have

$$\begin{aligned}(v_{0\ldots 0d_a d_b} - v_{0\ldots 0d_b})^2 &= y_{0\ldots 0d_a d_b,0\ldots 0d_a d_b} - 2y_{0\ldots 0d_a d_b,0\ldots 0d_b} + y_{0\ldots 0d_b,0\ldots 0d_b} = \\ &= y_{0\ldots 00d_a,0\ldots 0d_b} - 2y_{0\ldots 0d_a,0\ldots 0d_b} + y_{0\ldots 0d_b,0\ldots 0d_b} = \\ &= y_{0\ldots 00d_a,0\ldots 0d_b} - 2y_{0\ldots 0d_a,0\ldots 0d_b} + y_{0\ldots 0d_a,0\ldots 0d_b} = \\ &= 0\end{aligned}$$

because

$$\begin{aligned} y_{0\ldots 0d_b,0\ldots 0d_b} &= v_{0\ldots 0d_b} v_{0\ldots 0d_b} = \\ &= v_{0\ldots 0d_b} v_{0\ldots 0d_a} = \\ &= y_{0\ldots 0d_b,0\ldots 0d_a}. \end{aligned}$$

Similarly,

$$\begin{aligned} (v_{d_{i_1}\ldots d_{i_{k-1}}d_a} - v_{d_{i_1}\ldots d_{i_{k-1}}d_b})^2 &= y_{d_{i_1}\ldots d_{i_{k-1}}d_a,d_{i_1}\ldots d_{i_{k-1}}d_a} - 2y_{d_{i_1}\ldots d_{i_{k-1}}d_a,d_{i_1}\ldots d_{i_{k-1}}d_b} + \\ &+ y_{d_{i_1}\ldots d_{i_{k-1}}d_b,d_{i_1}\ldots d_{i_{k-1}}d_b} = \\ &= y_{d_{i_1}\ldots d_{i_{k-1}}0,0\ldots 0d_a} - 2y_{d_{i_1}\ldots d_{i_{k-1}}0,0\ldots 0d_a d_b} + y_{d_{i_1}\ldots d_{i_{k-1}}0,0\ldots 0d_b} \\ &= v_{d_{i_1}\ldots d_{i_{k-1}}}(v_{0,0\ldots 0d_a} - 2v_{0\ldots 0d_a d_b} + v_{0\ldots 0d_b}) = \\ &= 0 \end{aligned}$$

because of the first part of the Lemma. ■

The key part of the proof of the main result of this section is the following lemma:

Lemma 3.11 *Let ΔX be an instance of the turnpike problem and let us assume that all the differences in ΔX are different and that the instance ΔX has only one solution.*

Let us assume that we know that the numbers $x_1 > \cdots > x_l$ are in the solution set X and let us assume that the backtracking procedure positioned the numbers $b_1 > \cdots > b_k$ in X, and the next to be positioned is c.

Let Y be a feasible matrix for the relaxation (S_{k+1}) of the instance ΔX, and assume that the row vectors of V satisfy

$$v_{0\ldots 0x_i} = v_{0\ldots 0x_1}.$$

Then, if $u - v = c$, $u, v \neq 0, d_M$, where d_M is the largest number in ΔX,

$$y_{uv0\ldots 0,x_1 b_1\ldots b_k} = 0.$$

Proof: Since c is the largest unpositioned difference, in the equation

$$u - v = c,$$

u and v can not both be in the set

$$\{x_1, \ldots, x_l, b_1, \ldots, b_k\}.$$

If $u = d_M - x_i$ or $u = d_M - b_i$ then obviously

$$y_{uv0\ldots0,x_1b_1\ldots b_k} = 0.$$

because of Lemma 3.10 and because the differences are unique, so the pyramid constraints can be written for the differences x_i and $d_M - x_i$ or b_i or $d_M - b_i$.

Similarly, the lemma holds if $v = d_M - x_i$ or $v = d_M - b_i$.

If $u = b_i - b_j$ or $v = b_i - b_j$, the lemma holds since the difference $b_i - b_j$ appears exactly once.

We have to examine three other possibilities:

1. $u = b_i - x_i$ and $v = b_j - x_j$,
2. $u = b_i - x_i$ and $v = x_j$,
3. $u = b_i$ and $v = b_j - x_j$.

For Case 1 we have

$$y_{uv0\ldots0,b_1\ldots b_kx_i} = 0, \tag{3.35}$$

because $b_i - u = x_i$ so the above equation is just a pyramid constraint in the relaxation (S_{k+1}). Now because of Lemma 3.11 the claim follows from (3.35).

Case 2 can be shown in the same way.

For Case 3 notice that

$$y_{uv0\ldots0,b_1\ldots b_kx_j} = 0,$$

because $b_j - x_j = v$ so we can conclude that the lemma holds in the same way as in the Case 1. ∎

Now, we are ready to prove the main theorem of this section:

Theorem 3.12 *Let ΔX be an instance of the turnpike problem and let us assume that all the differences in ΔX are different and that the instance ΔX has only one solution. If the Skiena's et al. backtracking procedure has order k on the instance ΔX, then the relaxation (S_{k+1}) solves the instance ΔX.*

Proof:

Let $Y = VV^T$ be a feasible matrix for the relaxation (S_{k+1}) of the instance ΔX and let v_i, $i \in \Delta X$ be the row vectors of V.

Assume that the backtracking procedure has constructed a partial solution set $X = \{x_1 > \cdots > x_l\}$. We prove by induction on l, the number of elements in X, that all the vectors assigned to the elements of the partial solution set X are equal, i.e. that

$$v_{x_i 0\ldots 0} = v_{x_1 0\ldots 0}, \text{ for } i \in \{1, \ldots, l\}$$

The above statement is obviously true when $l = 1$.

So assume that the statement is true when there are l elements in the partial solution set X.

Assume that the backtracking procedure extended the partial solution set X by the numbers $B = \{a > b_1 > \cdots > b_{k-1}\}$, making c the largest unpositioned difference. Also assume that the backtracking procedure can not extend the set $X \cup B$ by c or $d_M - c$. We prove that then $y_{x_1 a0\ldots 0, x_1 a0\ldots 0} = 0$ and $y_{d_M - x_1 d_M - a0\ldots 0, d_M - x_1 d_M - a0\ldots 0} = 0$ and therefore by using Lemma 3.8 we have that $v_{x_1 0\ldots 0} = v_{d_M - a0\ldots 0}$. Similar argument holds if the number of elements of the set B is less than k.

If the backtracking procedure can not put c in $X \cup B$ that means either that for some element $z \in X \cup B$, the difference $z - c$ does not exist or that there are two identical differences $z_1 - c = z_2 - z_3$, for some $z_1, z_2, z_3 \in X \cup B$.

Then because of Lemma 3.10 for the elements of the matrix Y we have

$$y_{c0\ldots 0, ab_1\ldots b_{k-1}x_1} = 0 \tag{3.36}$$

and

$$y_{d_M - c0\ldots 0, ab_1\ldots b_{k-1}x_1} = 0. \tag{3.37}$$

If we now look at the equation for the difference c

$$v_{0c0\ldots0} + \sum_{\substack{u-v=c \\ u\neq 0,m}} v_{uv0\ldots0} + v_{d_M d_M - c0\ldots0} = v_{0\ldots0}$$

and multiply it by $v_{x_1ab_1\ldots b_{k-1}}$, because of Lemma 3.11 and (3.36) and (3.37), we can conclude that

$$v_{x_1ab_1\ldots b_{k-1}} = 0. \tag{3.38}$$

Next, we prove that

$$v_{x_1ab_1\ldots b_{k-2}d_M - b_{k-1}} = 0. \tag{3.39}$$

This is obviously true if the set X can not be extended by the elements $\{a, b_1, \ldots, d_M - b_{k-1}\}$. If this is not the case, we can prove (3.39) similarly as (3.38), using the equation for some difference d, which is next to position if we put the difference $d_M - b_{k-1}$ in the partial solution set.

Now, we can multiply the equation for b_{k-1}

$$v_{0b_{k-1}0\ldots0} + \sum_{\substack{u-v=b_{k-1} \\ u\neq 0,d_M}} v_{uv0\ldots0} + v_{d_M d_M - b_{k-1}0\ldots0} = v_{0\ldots0}$$

by $v_{x_1ab_1\ldots b_{k-2}0}$ and use Lemma 3.11 and equations (3.38) and (3.39) to obtain that

$$v_{x_1ab_1\ldots b_{k-2}0} = 0.$$

Now,

$$v_{x_1ab_1\ldots d_M - b_{k-2}0} = 0,$$

because otherwise we can put $d_M - b_{k-2}$ in the partial solution set X and in the same way as above conclude that

$$v_{x_1ab_1\ldots d_M - b_{k-2}b^1_{k-1}} = 0,$$

and from there that

$$v_{x_1ab_1\dots d_M-b_{k-2}0} = 0.$$

After repeating this process $k-1$ times, we can conclude that

$$v_{x_1a0\dots0} = 0. \tag{3.40}$$

If the procedure backtrack less than k steps, we also can conclude (3.40) by same reasoning.

Similarly, by regarding the mirror image of the partial solution set X we can conclude that

$$v_{d_M-x_1d_M-a0\dots0} = 0. \tag{3.41}$$

Also, note that

$$v_{x_10\dots0} + v_{d_M-x_10\dots0} = v_{0\dots0} \tag{3.42}$$

and

$$v_{a0\dots0} + v_{d_M-a0\dots0} = v_{0\dots0}, \tag{3.43}$$

so we can use Lemma 3.8 to conclude that

$$v_{x_10\dots0} = v_{d_M-a0\dots0}.$$

This proves that the vectors assigned to the numbers of a partial solution set of size $l+1$ are identical. ∎

Chapter 4

Heuristics

4.1 Introduction

In this chapter we show how to develop heuristics for solving the turnpike problem, based on the theoretical results of Chapter 2.

In the first section we describe a heuristic that is based on the relaxation (S_1). It also uses cuts from the relaxation (S_2) and a rounding technique.

In the second section we show how the relaxation (S_1) can be used to reduce the number of backtracking steps of the backtracking procedure of Skiena et al.

4.2 Introducing cuts from (S_2) into (S_1)

As we show in Chapter 5, the instances that are not solved by their relaxation (S_1) seem to be rare and we only need to add a couple of constraints from (S_2) to the relaxation (S_1) to solve these instances.

Also, for an instance ΔX of size m, a feasible point of its relaxation (S_1) is an $(m+1) \times (m+1)$ size matrix and there are $O(m)$ constraints in the definition of (S_1). The relaxation (S_2) of ΔX is much larger; a feasible matrix for (S_2) is an $\frac{m(m+1)}{2} \times \frac{m(m+1)}{2}$ matrix and there are $O(m^3)$ constraints in the definition of (S_2). Therefore an implementation of (S_2) is much more computationally demanding than an implementation of (S_1).

Therefore, we develop heuristics that would be based on (S_1) with additional cuts from (S_2) for solving the turnpike problem.

When developing the heuristics based on the relaxation (S_1), we assume that the variable $x_{0,0} = 1$. This makes feasible matrices belonging to the solution sets $0-1$ matrices.

Semidefinite program solvers are based on interior point methods and although a feasible region of a semidefinite program might be a convex combination of $0-1$ matrices, the solver might output a matrix that is not a $0-1$ matrix as the optimum. Therefore, we have to choose an objective function for (S_1) that guarantees that if the feasible region for (S_1) is a convex combination of $0-1$ matrices, the optimum is a $0-1$ matrix. For a given instance ΔX, where

$$\Delta X' = \{d_1 < d_2 < \cdots < d_M\}$$

one such objective function is

$$\sum_{i=0}^{M} 2^i x_{d_i,d_i}. \tag{4.1}$$

To see this, assume that that the optimum is achieved for a vector $(\alpha_0, \ldots, \alpha_M)^T$, where $0 < \alpha_M < 1$. Then since we assumed that the feasible region is a convex hull of $0-1$ matrices, we have that

$$\begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{M-1} \\ \alpha_M \end{bmatrix} = \alpha_M \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_{M-1} \\ 1 \end{bmatrix} + (1-\alpha_M) \begin{bmatrix} \gamma_0 \\ \vdots \\ \gamma_{M-1} \\ 0 \end{bmatrix}.$$

Now we show that the value of the objective function (4.1) on the vector $(\beta_0, \ldots,$

$\beta_{M-1}, 1)^T$ is greater than the value on the vector $(\alpha_0, \dots, \alpha_M)^T$. This is because

$$2^M + \sum_{i=0}^{M-1} \beta_i 2^i > 2^M + \sum_{i=0}^{M-1} \alpha_M \beta_i 2^i =$$
$$= \alpha_M 2^M + (1-\alpha_M)2^M + \alpha_M \sum_{i=0}^{M-1} \beta_i 2^i \geq$$
$$\geq \alpha_M (2^M + \sum_{i=0}^{M-1} \beta_i 2^i) + (1-\alpha_M) \sum_{i=1}^{M} \gamma_i 2^i =$$
$$= \sum_{i=0}^{M} \alpha_i 2^i.$$

Now, we can see that the other coordinates of the optimal vector have to be 0 or 1 in the similar way.

If the optimal matrix of the relaxation (S_1) is not a $0-1$ matrix, we can introduce cuts from (S_2) into the relaxation (S_1). We can add four main kinds of constraints:

(i) For each pyramid constraint $x_{0d_i,d_jd_k} = 0$, where $d_i, d_j, d_k \in \Delta X'$, in (S_2), we can add the following two constraints to the relaxation (S_1)

$$x_{d_i,d_i} + x_{d_j,d_j} + x_{d_k,d_k} \leq 2,$$
$$x_{d_i,d_j} + x_{d_i,d_k} + x_{d_j,d_k} \leq 1.$$

These constraints hold for a $0-1$ matrix that is feasible for (S_1), because if $x_{0d_i,d_jd_k} = 0$, at most two of the differences d_i, d_j, d_k can be in a solution set.

Also for a pyramid constraint of the type $x_{d_id_j,d_kd_l} = 0$, where $d_i, d_j, d_k, d_l \in \Delta X'$, we can add the following constraint to the relaxation (S_1)

$$x_{d_i,d_i} + x_{d_j,d_j} + x_{d_k,d_k} + x_{d_l,d_l} \leq 3$$
$$x_{d_i,d_j} + x_{d_i,d_k} + x_{d_i,d_l} + x_{d_j,d_k} + x_{d_j,d_l} + x_{d_k,d_l} \leq 3.$$

These constraints hold for a $0-1$ matrix that is feasible for (S_1), because if $x_{d_id_j,d_kd_l} = 0$, at most three of the differences d_i, d_j, d_k, d_l can be in a solution set.

(ii) Let X_1 be a feasible matrix for the relaxation (S_1) and X_2 be a feasible matrix for the relaxation (S_2) of an instance ΔX. Since X_2 is positive semidefinite, there exists a matrix V such that

$$X_2 = VV^T.$$

Let $v_{d_i d_j}$, for $d_i, d_j \in \Delta X'$, denote the row vectors of V. Then for every constraint in the relaxation (S_1) of the type

$$\sum_{\substack{d_a, d_b \in \Delta X' \\ d_a - d_b = d_i}} x_{d_a, d_b} = v(d_i) x_{0,0}$$

there is a corresponding relation between the row vectors of V. Namely, we have

$$\sum_{\substack{d_a, d_b \in \Delta X' \\ d_a - d_b = d_i}} v_{d_a d_b} = v(d_i) v_{00}. \tag{4.2}$$

This easily follows from the constraints of (S_2), by looking at the expression

$$\Big(\sum_{\substack{d_a, d_b \in \Delta X' \\ d_a - d_b = d_i}} v_{d_a d_b} - v(d_i) v_{00} \Big)^2$$

and using the pyramid equalities to evaluate it to 0.

The equation (4.2) can be multiplied by any other row vector of V, to obtain a constraint on the elements of the matrix X_2. This constraint obviously already holds for the elements of X_2.

Assume that there exist a vector v_{0d_j}, $d_j \in \Delta X'$ such that for any vector v_{d_a,d_b}, $d_a - d_b = d_i$, $d_a, d_b \neq 0$, $d_a, d_b \neq d_M$, we either have that $v_{d_a,d_b} v_{0,d_j} = 0$, or $d_a = d_j$ or $d_b = d_j$.

We can multiply the equality (4.2) by the vector v_{0d_j} to obtain a constraint that holds for the elements of X_2

$$x_{0d_i,0d_j} + x_{d_j\, d_j + d_i, 0d_j} + x_{0\, d_M - d_i, 0d_j} = x_{0d_j,0d_j},$$

when $d_b = d_j$ or

$$x_{0d_i,0d_j} + x_{d_j\,d_j-d_i,0d_j} + x_{0\,d_M-d_i,0d_j} = x_{0d_j,0d_j},$$

when $d_a = d_j$.

This constraint can easily be translated into a constraint for (S_1):

$$x_{d_i,d_j} + x_{d_j,d_i+d_j} + x_{d_M-d_i,d_j} = x_{d_j,d_j},$$

or

$$x_{d_i,d_j} + x_{d_j,d_j-d_i} + x_{d_M-d_i,d_j} = x_{d_j,d_j}.$$

For example if $\Delta X = \{28, 26, 25, 23, \dots\}$ the constraint for the difference 23 in the relaxation (S_1) is

$$x_{0,23} + x_{2,25} + x_{3,26} + x_{5,28} = 1,$$

from which in the above described way we can obtain another constraint valid for (S_1):

$$x_{2,23} + x_{2,25} + x_{2,5} = x_{2,2}.$$

because of the pyramid constraint $x_{3\,26,0\,2} = 0$ in (S_2).

Also, if

$$\sum_{d_a,d_b \in D} v_{d_a d_b} = v_{00}, \tag{4.3}$$

for some subset D of $\Delta X'$, we can multiply (4.3) by one of the summands v_{d_u,d_v} in (4.3), to obtain a pyramid constraint $x_{d_a d_b, d_u d_v} = 0$, for $d_a, d_b \in D$, $d_a \neq d_u$ and $d_b \neq d_v$. We can add these pyramid constraints to the relaxation (S_1), as described in (i).

If

$$v_{0d_u} + \sum_{d_a,d_b \in D} v_{d_a d_b} + v_{0d_v} = 2v_{00}, \tag{4.4}$$

for some subset D of $\Delta X'$, we can multiply (4.4) by v_{d_u,d_v} to obtain a pyramid constraint $x_{d_a d_b, d_u d_v} = 0$, for $d_a, d_b \in D$, $d_a \neq d_u$ and $d_b \neq d_v$. We can add these pyramid constraints to the relaxation (S_1) too, as described in (i).

Note that the last two types of constraints are just the regular pyramid constraints if (4.3) and (4.4) are of the type (4.2). We need these constraints because a heuristic which we will construct maintains a list of constraints which hold for ΔX and partial solution set that is being constructed. The list is updated at each execution step.

(iii) If in (S_2) for some row vectors of V the following holds

$$\sum_{a=1}^{l} v_{0d_{i_a}} = j v_{00},$$

then for the variables of (S_2) we have

$$\sum_{a>b=1}^{l} x_{d_{i_a} d_{i_b}, 00} = \binom{j}{2}. \tag{4.5}$$

This follows from

$$(\sum_{a=1}^{l} v_{0d_{i_a}} - j v_{00})^2 = 0$$

by evaluating the right hand side of the above equation and using the pyramid constraints from (S_2).

We can therefore add the constraint

$$\sum_{a>b=1}^{l} x_{d_{i_a}, d_{i_b}} = \binom{j}{2}.$$

to the definition of (S_1).

(iv) In (S_2) we have that

$$x_{d_i d_j, d_k d_l} \leq x_{0d_j, d_k d_l} \leq x_{00, d_k d_l} \leq x_{00, 0d_l}.$$

These constraints can be used to obtain constraints on the variables of (S_1), from any constraint from (S_2) which represents a variable $x_{0d_i,0d_j}$, for $d_i, d_j \in \Delta X'$, as a sum of some other variables of (S_2).

Now, we can use (i)-(iv), to construct a heuristic for solving the turnpike problem. The heuristic first checks if the relaxation (S_1) solves the given instance. If not, it adds the pyramid constraints as described in (i), and checks if the new relaxation solves the instance. If it does not, the heuristic starts adding constraints described in (ii) and (iii) until it obtains a $0-1$ solution that corresponds to a solution of the instance or concludes that it can not proceed, at which point it outputs the partial solution set it constructed up to this step. The heuristics maintains a list of constraints and a list of differences in a partial solution set that it is constructing. At each step it constrains the variable corresponding to the largest unpositioned difference to be 1. Then it recomputes the list of the positioned differences and the list of constraints and applies (ii) and (iii) to obtain more constraints from the updated list.

Now, we give the heuristic:

Given an instance ΔX of the turnpike problem, where

$$\Delta X' = \{d_1 < d_2 < \cdots < d_M\}$$

and the size of $\Delta X = \binom{n}{2}$:

1. Write and solve the relaxation (S_1) of the instance ΔX with the objective function (4.1).

2. If the solution is a $0-1$ matrix, output the solution and stop. Otherwise initialize the set of differences S that are set to 1 or 0 to be the empty set. Initialize the list of constraints to the equality constraints in (S_1). Update the list of constraints. If in the set S there are n elements whose indicator variables are set to 1, output the solution and stop. Otherwise continue with Step 3.

3. Add the pyramid constraints from (S_2), as described in (i) above, to the relaxation (S_1) and to the list of constraints. Solve the new relaxation.

4. If the solution is a $0-1$ matrix output the solution and stop. Otherwise add the constraints described in (ii) and (iii) to the relaxation and the list of constraints. Update the list of constraints. Solve the new relaxation.

5. If the solution is a $0-1$ matrix, output the solution and stop.
   Otherwise

   Set the indicator variable x_{d_i,d_i} of the largest unset difference to 1.
   Solve the new relaxation.
   If the relaxation is feasible:

   Add d_i to the set S.
   Update the list of constraints.
   If in the set S there are less than n differences whose indicator variables are set to 1, solve the new relaxation and go to step 4.

   If the relaxation is not feasible

   Set $x_{d_M-d_i,d_M-d_i}$ to 1.
   Solve the new relaxation.
   If the relaxation is feasible:

   Add $d_M - d_i$ to the set S.
   Update the list of constraints.
   If in the set S there are less than n differences whose indicator variables are set to 1, solve the new relaxation and go to step step 4.

   If the relaxation is infeasible, output the elements of the set S and stop.

To update the constraints we repeat the following until no further changes to the list of constraints are possible or inconsistency is found:

If x_{d_i,d_i} is set to 1, for every constraint in the list, replace any occurrence of $x_{d_id_j}$ by $x_{d_jd_j}$ for $d_j \in \Delta X'$. If in any constraint the value of one or more variables is 1, subtract them from the value on the right side. If there is a new constraint of the type

$$\sum_{l=1}^{k} x_{d_{i_l},d_{i_l}} = 0,$$

put the differences d_{i_l}, for $l = 1, \ldots, k$ in S and in any constraint erase the summands of the form $x_{d_{i_l},d_j}$, for $l = 1, \ldots, k$ and $j \in \Delta X'$.

If there is a new constraint of the type

$$\sum_{l=1}^{k} x_{d_{i_l},d_{i_l}} = k,$$

put d_{i_l} in S and set $x_{d_{i_l},d_{i_l}}$ to 1 for $l = 1, \ldots, k$.

If an inconsistency is found, print out a message and stop.

This heuristic is used to solve some instances in Chapter 5. It was noticed that only one iteration of step 4 was sufficient to solve instances that we examined.

4.3 Using the relaxation (S_1) in conjunction with the backtracking procedure

The backtracking procedure by Skiena et al. takes into account only a certain number of differences at any given time during the execution, whereas the relaxation (S_1) treats all the differences simultaneously.

We could therefore solve the relaxation (S_1) at each step of the backtracking procedure. If the backtracking procedure is currently positioning a difference d_i and if it can position d_i in the partial solution set without creating a conflict, the relaxation (S_1) can serve as another check for the validity of that positioning. We can constrain the indicator variables of the elements of the partial solution set that the backtracking procedure is constructing and the indicator variable x_{d_i,d_i} of the relaxation (S_1) to be equal to 1, and solve the relaxation. If the relaxation is feasible we put the difference d_i in the partial solution set and continue the execution of the backtracking procedure.

If the relaxation is not feasible, we bypass the backtracking steps and immediately assume that the difference $d_M - d_i$ is in the partial solution set. Again we can establish the feasibility of the relaxation (S_1) under this new assumption. If it is infeasible, we immediately backtrack.

This heuristic can be quite powerful, especially since the instances of the class constructed by Zhang, [35], on which the backtracking procedure takes exponential time, are solved by the their relaxations (S_1), as shown in Chapter 3.

Chapter 5

Computational Results

5.1 Introduction

In this chapter we enumerate the instances of the turnpike problem for which their relaxations (S_1'), (S_1) and (S_2) were implemented. The computational results show that most of the examined instances are solved by their relaxation (S_1') and the ones that are not, have a feasible point of the form

$$Y = \sum_{i=1}^{k} \lambda_i z_i z_i^T, \tag{5.1}$$

where, $\lambda_i \geq 0$ and z_i are $0-1$ vectors for $i \in \{1, \ldots, k\}$, but not necessarily characteristic vectors of the solutions of ΔX. When implementing the relaxation (S_1') and (S_1) we constrained the element $y_{0,0}$ of any feasible matrix Y for these relaxations to be 1. That means that the combination (5.1) is a convex combination, i.e.

$$\sum_{i=1}^{k} \lambda_i = 1.$$

In particular we give five instances that are not solvable by their relaxation (S_1') and show how to use them construct classes of instances that are not solvable by the relaxation (S_1').

We do not have an instance of the turnpike problem which is not solved by its relaxation (S_2).

5.2 Result Description

The instances ΔX of the turnpike problem for which the relaxation (S_1') was implemented are

1. all ΔX containing 10 numbers, all of which are different and the largest difference in ΔX is less or equal to 21,
2. all ΔX containing 15 numbers, all of which are different and the largest difference in ΔX is less or equal to 24,
3. all ΔX containing 21 numbers, all of which are different and the largest difference in ΔX is less or equal to 28,
4. all ΔX containing 28 numbers, all of which are different and the largest difference in ΔX is less or equal to 31,
5. all ΔX is a difference of a set X that contains at most 13 elements and if we sort the elements of X, the difference of two consecutive elements is at most 2,
6. the numbers of ΔX are chose randomly with uniform distribution and the size of ΔX is less or equal to $\binom{20}{2}$ (about 10 000 instances).

Semidefinite programs are solved using SDPSOL, developed by Wu and Boyd, [6]. Although we encountered problems with the stability of the code, this package was chosen because of the nice modelling language.

The computational results show that the relaxation (S_1') solves most of the above enumerated instances ΔX, i.e. the feasible matrices Y for the relaxation (S_1') of ΔX are of the form

$$Y = \sum_{i=1}^{k} \lambda_i y_i y_i^T,$$

where $\sum_{i=1}^{k} \lambda_i = 1$ and for $i \in \{1, \ldots, k\}$, $\lambda_i \geq 0$ and y_i is a characteristic vector of a solution set X_i of the instance ΔX.

If the relaxation (S_1') does not solve an instance ΔX for the instances we examined, any feasible matrix for the relaxation (S_1') of ΔX has the form

$$Y = \sum_{i=1}^{l} \lambda_i z_i z_i^T,$$

where $\sum_{i=1}^{l} \lambda_i = 1$ and for $i \in \{1, \ldots, l\}$, $\lambda_i \geq 0$ and z_i are $0-1$ vectors, but not necessarily characteristic vectors of the solutions of ΔX.

We now list all the instances of the above enumerated examples which are not solved by their relaxation (S_1').

1. The set

$$\Delta X = \{1, 2, 3, 4, 5, 6, 7, 8, 10, 11\}$$

is not a difference set of any set X, but there is a feasible point for the relaxation (S_1') of ΔX. To see this let z_1 and z_2 be vectors indexed by the elements of ΔX. Let z_1 be 0 everywhere except on the positions 0, 4, 8, 10, 11, and let z_2 be 0 everywhere except on the positions 0, 3, 5, 10, 11. Then the matrix $Y = 0.5(z_1 z_1^T + z_2 z_2^T)$ is feasible for (S_1'). This is easily seen if we construct difference sets associated with z_1 and z_2 and organize them in pyramids as described in Chapter 1 and shown in Figure 5.1.

$$z_1 \quad \begin{array}{ccccccc} & & & 11 & & & \\ & & 10 & & 7 & & \\ & 8 & & 6 & & 3 & \\ 4 & & 4 & & 2 & & 1 \end{array} \qquad z_2 \quad \begin{array}{ccccccc} & & & 11 & & & \\ & & 10 & & 8 & & \\ & 5 & & 7 & & 6 & \\ 3 & & 2 & & 5 & & 1 \end{array}$$

Figure 5.1: Difference sets of $\{0, 4, 8, 10, 11\}$ and $\{0, 3, 5, 10, 11\}$.

Now, in the pyramid associated with z_1 we have two 4 entries and no 5 entries, and in the pyramid associated with z_2 we have no 4 entries and two 5 entries. Therefore the convex combination of the two is exactly ΔX.

2. The set

$$\Delta X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 20, 22, 23, 25\}$$

is also not a difference set of any set, but there is a feasible point for the relaxation (S_1') of the instance ΔX. In order to see that, we just write the pyramids associated with the points z_1 and z_2 as above

z_1

25
23 20
22 18 16
17 17 14 8
9 12 13 6 3
5 4 8 5 1 2

z_2

25
23 22
14 20 18
13 11 16 12
7 10 7 10 11
3 4 6 1 9 2

Now we look at the number of times the elements of ΔX appear in the difference sets of the sets $Z_1 = \{0, 5, 9, 17, 22, 23, 25\}$ (labelled z_1 above) and $Z_2 = \{0, 3, 7, 13, 14, 23, 25\}$ (labelled z_2 above). In Table 1, we have all the elements of ΔX, that either appear in the multisets ΔZ_1 and ΔZ_2 more than once or not at all.

	5	7	8	10	11	17
ΔZ_1	2	0	2	0	0	2
ΔZ_2	0	2	0	2	2	0

Table 1

Now it is easy to see that if z_1 is the characteristic vector of Z_1 indexed by the elements of ΔX, and z_2 is the characteristic vector of Z_2 indexed by the elements of ΔX, that the matrix

$$Y = 0.5(z_1 z_1^T + z_2 z_2^T)$$

is feasible for the relaxation (S_1') of the instance ΔX.

3. The set

$$\Delta X = \{1,2,3,4,5,6,7,8,10,11,13,14,16,17,19,21,24,25,26,27\}$$

is not a difference set, but a feasible point of the relaxation (S_1') can be constructed similarly as above, if we consider the following pyramids

$$
z_1 \quad
\begin{array}{ccccccccccc}
 & & & & & 27 & & & & & \\
 & & & & 25 & & 26 & & & & \\
 & & & 20 & & 24 & & 21 & & & \\
 & & 17 & & 19 & & 19 & & 10 & & \\
 & 6 & & 16 & & 14 & & 8 & & 7 & \\
1 & & 5 & & 11 & & 3 & & 5 & & 2
\end{array}
$$

$$
z_2 \quad
\begin{array}{ccccccccccc}
 & & & & & 27 & & & & & \\
 & & & & 25 & & 26 & & & & \\
 & & & 21 & & 24 & & 13 & & & \\
 & & 17 & & 20 & & 11 & & 10 & & \\
 & 14 & & 16 & & 7 & & 8 & & 6 & \\
1 & & 13 & & 3 & & 4 & & 4 & & 2
\end{array}
$$

The elements of ΔX that appear in ΔZ_1 and ΔZ_2 more than once or not at all are given in Table 2.

	4	5	13	19
ΔZ_1	0	2	0	2
ΔZ_2	2	0	2	0

Table 2

4. The relaxation (S_1') of the instance ΔX, where ΔX is the difference set of

$$X = \{0, 2, 3, 5, 7, 9, 10, 11, 12, 13, 15, 16\}$$

is satisfied by a point that is obtained in the same way as above and determined by the pyramids:

```
                    16
                  15  14
                13  13  13
              12  11  12  11
            10  10  10  10  10
z1         8   8   9   8   9   9
          7   6   7   7   7   8   8
         6   5   5   5   6   6   7   6
        5   4   4   3   4   5   5   5   4
       3   3   3   2   2   3   4   3   3   3
      2   1   2   1   1   1   2   2   1   2   1
```

```
                    16
                  15  14
                13  13  13
              11  11  12  12
             9   9  10  11  11
z2         7   7   8   9  10  10
          6   5   6   7   8   9   9
         5   4   4   5   6   7   8   7
        4   3   3   3   4   5   6   6   5
       3   2   2   2   2   3   4   4   4   3
      2   1   1   1   1   1   2   2   2   2   1
```

In Table 3 we give the list of elements of ΔX that appear in ΔZ_1 and ΔZ_2 different number of times than in ΔX. The number of times these elements appear in ΔX, ΔZ_1 and ΔZ_2 is also given.

	2	3	4	5	8	9	10	11
ΔX	8	7	6	6	4	4	4	3
ΔZ_2	7	8	5	7	5	3	5	2
ΔZ_2	9	6	7	5	3	5	3	4

Table 3

5. The relaxation (S_1') of the instance ΔX, where ΔX is the difference set of

$$X = \{0, 3, 5, 7, 10, 11, 12\}$$

is satisfied by a point that is obtained in the same way as above and determined by the pyramids:

z_1

```
               12
            11    10
          9    9    8
        7   7    7    5
      4   5    5    4    3
    2   2   3    2    2    1
```

z_2

```
               12
            11    8
          10   7    7
        7   6    6    5
      5   3    5    4    2
    4   1   2    3    1    1
```

In Table 4 we give the list of elements of ΔX that appear in ΔZ_1 and ΔZ_2 different number of times than in ΔX. The number of times these elements appear in ΔX, ΔZ_1 and ΔZ_2 is also given.

	1	2	6	9
ΔX	2	3	1	1
ΔZ_1	1	4	0	2
ΔZ_2	3	2	2	0

Table 4

Note that

$$X_1 = \{2, 4, 5, 7, 11, 12\}$$

is another solution of the instance ΔX.

The relaxation (S_2) solves all of the above instances. The relaxation (S_2) is too large for computational purposes, so we added constraints from (S_2) to the relaxation (S_1).

In fact the instances 1, 2 and 3 are solved by the relaxation (S_1).

Let us now look at the instance 4. Assume that Y_1 is a feasible matrix for the relaxation (S_1') and Y_2 is a feasible matrix for the relaxation (S_2) of that instance. It is easy to see that the constraint for the difference 11 in (S_1') can be reduced to

$$y_{0,11} + y_{12,1} + y_{15,4} + y_{0,5} = 2.$$

If V is a matrix such that

$$Y_2 = VV^T,$$

and $v_{d_i d_j}$ are row vectors of V for $d_i, d_j \in \Delta X'$, then the constraint for the difference 11 in (S_2) can be written in terms of vectors $v_{d_i d_j}$ as

$$v_{0,11} + v_{12,1} + v_{15,4} + v_{0,5} = 2v_{0,0}.$$

We can multiply the above constraint by $v_{5,11}$ and use the pyramid equalities from the definition of (S_2) to obtain that

$$y_{12,1,5,11} + y_{15,4,5,11} = 0.$$

Using this constraint we can introduce cuts on S_1' as described in Chapter 4. The cuts are

$$y_{1,11} + y_{1,12} + y_{1,5} + y_{5,11} + y_{5,12} + y_{11,12} \leq 3, \tag{5.2}$$

$$y_{4,5} + y_{4,11} + y_{4,15} + y_{5,11} + y_{5,15} + y_{11,15} \leq 3,. \tag{5.3}$$

If we add these constraints to the constraints of (S_1'), the newly obtained relaxation solves the instance 4.

Similarly, let Y_1 be a feasible matrix for the relaxation (S_1') and Y_2 is a feasible matrix for the relaxation (S_2) of the instance 5. Again, let V be a matrix such that

$$Y_2 = VV^T,$$

and $v_{d_i d_j}$ are row vectors of V for $d_i, d_j \in \Delta X'$. The constraint for the difference 9 in (S_1') is

$$y_{0,9} + y_{1,10} + y_{2,11} + y_{3,12} = 1,$$

and therefore we have the following constraint

$$v_{0,9} + v_{1,10} + v_{2,11} + v_{3,12} = v_{0,0},$$

for the vectors $v_{0,9}$, $v_{1,10}$, $v_{2,11}$, $v_{3,12}$ and $v_{0,0}$. We can multiply the above constraint by $v_{0,1}$, $v_{0,11}$, $v_{0,2}$ and $v_{0,10}$ and use the pyramid constraints from the definition of (S_2) to obtain the following cuts on S_1', as described in Chapter 4:

$$\begin{aligned} y_{9,1} + y_{10,1} + y_{3,1} &= y_{1,1} \\ y_{9,11} + y_{2,11} + y_{3,11} &= y_{11,11} \\ y_{9,2} + y_{11,2} + y_{3,2} &= y_{2,2} \\ y_{9,10} + y_{10,1} + y_{3,10} &= y_{10,10}. \end{aligned} \tag{5.4}$$

If we add the constraints (5.4) to the constraints of (S_1') the newly obtained relaxation solves the instance 5.

For all of the above instances, the feasible points in their relaxation (S_1') are of the form

$$\sum_{i=1}^{l} \lambda_i z_i z_i^T,$$

where $\sum_{i=1}^{l} \lambda_i = 1$ and for $i \in \{1, \dots k\}$, $\lambda_i \geq 0$ and z_i are $0-1$ vectors that are not necessarily characteristic vectors of the solutions.

If Y_2 is a feasible matrix for a relaxation (S_2) of an instance ΔX of the turnpike problem, and if

$$Y_2 = \sum_{i=1}^{m} \lambda_i u_i u_i^T,$$

where $\sum_{i=1}^{m} \lambda_i = 1$ and for $i \in \{1, \dots, m\}$, $\lambda_i \geq 0$ and u_i are $0-1$ vectors, then the submatrix Y_2^1 of Y_2, determined by the diagonal elements $0d_i$, for $d_i \in \Delta X'$, is of the form

$$\sum \lambda_i s_i s_i^T,$$

where s_i are $0-1$ vectors that are characteristic vectors of the solutions of the instance ΔX.

To see this, we first prove the well known inequality between the arithmetic and quadratic mean:

Lemma 5.1 *Let x_i, $i = 1, \dots, n$ be non-negative numbers and let*

$$\sum_{i=1}^{n} \lambda_i x_i = k, \tag{5.5}$$

for $\lambda_i \geq 0$, $i = 1, \dots, k$, and $\sum_{i=1}^{k} \lambda_i = 1$. Then

$$\sum_{i=1}^{n} \lambda_i x_i^2 \geq k^2,$$

and equality holds if and only if all the numbers x_i are equal.

Proof: From (5.5) we have

$$\begin{aligned}
k^2 =& (\sum_{i=1}^{n} \lambda_i x_i)^2 = \\
=& \sum_{i=1}^{n} \lambda_i^2 x_i^2 + 2 \sum_{i<j=1}^{n} \lambda_i \lambda_j x_i x_j = \\
=& \sum_{i=1}^{n} \lambda_i x_i^2 + \sum_{i=1}^{n} \lambda_i(\lambda_i - 1) x_i^2 + 2 \sum_{i<j=1}^{n} \lambda_i \lambda_j x_i x_j = \\
=& \sum_{i=1}^{n} \lambda_i x_i^2 + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \lambda_i \lambda_j x_i^2 + 2 \sum_{i<j=1}^{n} \lambda_i \lambda_j x_i x_j = \\
=& \sum_{i=1}^{n} \lambda_i x_i^2 + \sum_{i<j=1}^{n} \lambda_i \lambda_j (x_i - x_j)^2,
\end{aligned}$$

from which the claim of the lemma follows directly because $\sum_{i<j=1}^{n} \lambda_i \lambda_j (x_i - x_j)^2 \geq 0$. ■

Now, we can prove

Lemma 5.2 *Let X_2 be a matrix feasible for the relaxation (S_2) of the instance ΔX of the turnpike problem. Let X_1^2 be the submatrix of X_2 determined by the diagonal elements $(X_2)_{0d_i,0d_i}$, for $d_i \in \Delta X$. If*

$$X_2 = \sum_{i=1}^{m} \lambda_i u_i u_i^T,$$

where $\sum_{i=1}^{m} \lambda_i = 1$ and for $i \in \{1, \dots, m\}$, $\lambda_i \geq 0$ and u_i are $0-1$ vectors, then

$$X_1^2 = \sum_{i=1}^{m} \lambda_i s_i s_i^T,$$

where vectors s_i are characteristic vectors of the solutions of the instance ΔX.

Proof: Because of the way X_1^2 is constructed, it is obvious that it is of the form

$$X_1^2 = \sum_{i=1}^{m} \lambda_i y_i y_i^T,$$

for some $0-1$ vectors y_i, $i \in \{1, \ldots, m\}$. We only have to prove that y_i are characteristic vectors of the solutions of ΔX.

For a difference $d_i \in \Delta X$, we can look at the submatrix A of X_2 determined by the diagonal elements $(X_2)_{d_j d_k, d_j d_k}$, where $d_j - d_k = d_i$. Because of the constraints in (S_2), the diagonal of A sums to $v(d_i)$, the number of times the difference d_i occurs in ΔX, and all the entries of A sum to $v(d_i)^2$.

Let $U_l = u_l u_l^T$ for $l \in \{1, \ldots, m\}$ and let x_i denote the sum of the diagonal entries of U_l. Then because of the form of Y_2 we have

$$\sum_{i=1}^{m} \lambda_i x_i = v(d_i)$$

and

$$\sum_{i=1}^{m} \lambda_i x_i^2 = v(d_i)^2.$$

Now from Lemma 5.1 we have that $x_l = v(d_i)$ for $l \in \{1, \ldots, m\}$ and therefore for a $0-1$ matrix of the form

$$y_i y_i^T$$

all the equality constraints of (S_1) hold, so we can conclude that the vectors y_i are characteristic vectors of the solutions of ΔX. ∎

Since, we have no instance for which a feasible matrix of its relaxation (S_1) would not be a convex combination of $0-1$ matrices, it is reasonable to expect that any feasible matrix of its relaxation (S_2) would also be a convex combination of $0-1$ vectors in which case, because of Lemma 5.2, the instance would be solved by that relaxation.

A class of instances which are not solvable by their relaxation (S_1) can be obtained from any of the instances 1-5, in a similar way in which new instances were constructed from smaller ones in Chapter 3.

For example, for the first instance, i.e.

$$\Delta X = \{1, 2, 3, 4, 5, 6, 7, 8, 10, 11\},$$

Let $Q(x)$ be the generating function for the multiset $\Delta X \cup (-\Delta X)$, $P_1(x)$ be the generating polynomial for the set $\{0, 4, 8, 10, 11\}$ (this is the set associated with the vector z_1 from the instance 1), and $P_2(x)$ be the generating polynomial for the set $\{0, 3, 5, 10, 11\}$ (this is the set associated with the vector z_2 from the instance 1), Then

$$Q(X) + 5 = \frac{1}{2}(P_1(x)P_1(x^{-1}) + P_2(x)P_2(x^{-1})). \tag{5.6}$$

The equation (5.6) can be multiplied by $R(x)R(x^{-1})$, where $R(x)$ is a polynomial such that the coefficients of the polynomial $R(x)P_1(x)$ are 0 or 1 and the coefficients of the polynomial $R(x)P_2(x)$ are 0 or 1. This condition is needed because it ensures that the exponents of $R(x)P_1(x)$, $R(x)P_2(x)$ respectively, form a set so we can construct their difference sets. We have

$$\begin{aligned}(Q(X) + 5)R(x)R(x^{-1}) &= \frac{1}{2}(P_1(x)R(x)P_1(x^{-1})R(x^{-1}) + P_2(x)R(x)P_2(x^{-1})R(x^{-1})) \\ &= \frac{1}{2}(T_1(x)T_1(x^{-1}) + T_2(x)T_2(x^{-1}))\end{aligned}$$

for some polynomials $T_1(x)$ and $T_2(x)$ whose coefficients are 0 or 1.

Then if

$$(Q(x) + 5)R(x)R(x^{-1}) = \sum_{i=1}^{n} a_i(x^i + x^{-i}) + m,$$

let ΔY be the instance that contains a_i copies of number i, for $i = 1, \dots, n$.

Then, it is easy to choose polynomials $R(x)$ such that the instance ΔY is not a difference set. For example, if $R(x) = 1 + \sum_{i=1}^{n} a_i x^i$ are the polynomials from Theorem 3.6, i.e. if

$$a_1 \geq 3d_M + 1 \tag{5.7}$$

and

$$a_i \geq 3a_{i-i} + d_M + 1 \text{ for } i \in \{2, \dots, n\}, \tag{5.8}$$

where d_M is the maximum element of the instance 1, then the instance ΔY is not a difference set. We can see this in the same way as in the proof of Theorem 3.6, by recognizing that the instance ΔY contains the instance 1, as a subproblem, and also that the feasibility of the relaxation (S_1') of the instance ΔY depends on the feasibility of the relaxation (S_1') of the instance 1.

However, the instances obtained from the instances 1 by using polynomials $R(x)$ that satisfy (5.7) and (5.8) are solved by their relaxation (S_2). This can be shown by closely following the reasoning behind the proof of Theorem 3.6.

The instances which are obtained from instance 1, in the way described above and using the polynomials that do not satisfy (5.7) or (5.8) were tested computationally, but not extensively, so they still might be good candidates for the instances that are not solved by their relaxation (S_2).

Similar construction can be done if we start with the instances 2-4 instead of instance 1.

Chapter 6

Other relaxations

6.1 Introduction

The two relaxations of the turnpike problem presented in this chapter were proposed by A. Schrijver [29].

First, the turnpike problem is formulated as a $0-1$ quadratic program, whose semidefinite relaxation is too large for practical purposes. We use association schemes and some other methods, to reduce the size of the $0-1$ quadratic program to obtain a semidefinite relaxation which is smaller and practically possible to solve by today's computers.

6.2 Formulation of the Relaxations

In order to simplify the exposition, we will assume that the given multiset ΔX is a set, i.e. that the numbers in ΔX do not repeat. So, let $n \in \mathbb{N}$, $m = \binom{n}{2}$, and $\Delta X = \{d_1 < d_2 < \cdots < d_m\}$. Furthermore let

$$D = \Delta X \cup (-\Delta X),$$

$$V = \{1, \ldots, n\}$$

and

$$A = \{(u, v) | u, v \in V, u \neq v\}.$$

If the given ΔX is a difference set, there exists a function

$$f : V \to \mathbf{R}$$

such that

$$\{f(u) - f(v) | (u, v) \in A\} = D.$$

First, let us look at the directed complete graph on n vertices whose vertices are labelled by the elements of X, which is a solution of the instance ΔX, and whose edges are labelled by the elements of D, such that an edge (u, v) is labelled by $v - u$. Obviously all the elements of ΔX appear as edge labels. Figure 6.1 shows the directed graphs obtained in that way from the set $X = \{0, 1, 3, 8, 14, 18\}$ and its mirror image, i.e. the set $d_m - X$.

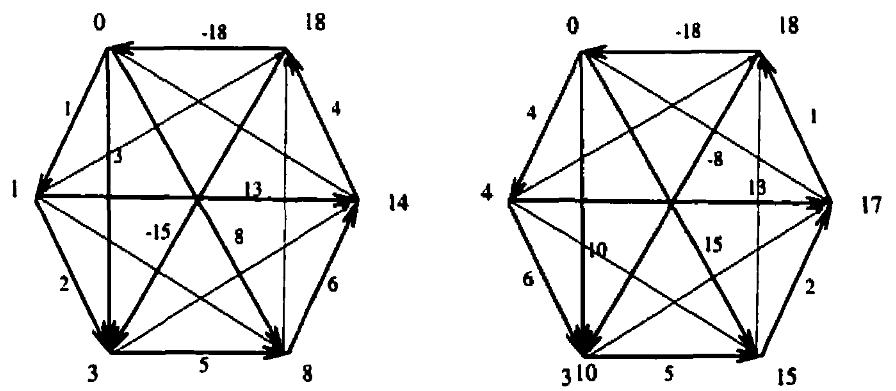

Figure 6.1: K_n labelled by the elements of ΔX.

Note that a function f that satisfies the above conditions can be obtained by assigning any permutation of V to vertices of the above graphs and defining $f(i)$ to be equal to the label of the vertex to which i is assigned. For our example f could be given by:

$$\begin{array}{ccc} f(1)=0 & f(1)=8 & f(1)=0 \\ f(2)=1 & f(2)=3 & f(2)=4 \\ f(3)=3 \text{ or} & f(3)=1 \text{ or} & f(3)=10 \\ f(4)=8 & f(4)=18 & f(4)=15 \\ f(5)=14 & f(5)=0 & f(5)=17 \\ f(6)=18 & f(6)=14 & f(6)=18 \end{array}$$

Then there also exists a bijection $g: A \to D$, such that

$$g(u,v) = -g(v,u), \text{ for all } u,v \in V,\ u \neq v$$
$$g(u,v) + g(v,w) + g(w,u) = 0, \text{ for all distinct } u,v,w \in V.$$

One such bijection is obviously $g(u,v) = f(v) - f(u)$, for any of the above functions f.

Let

$$H = \{(a,d) \in A \times D | g(a) = d\}.$$

Let $y = \chi^H$ be a characteristic vector of H in $A \times D$. So, $y_{(a,d)} = 1$ if the difference d is realized on the arc a, and 0 otherwise. Then the turnpike problem (P) is equivalent to the question of non-emptiness of the following subset of $\mathbb{R}^{n^2(n-1)^2}$:

$$\begin{aligned} & y_{(a,d_1)} y_{(a,d_2)} = 0, \text{ for every } a \in A \text{ and } d_1 \neq d_2 \in D \\ & y_{(a_1,d)} y_{(a_2,d)} = 0, \text{ for every } d \in D \text{ and } a_1 \neq a_2 \in A, \\ & \sum_{a \in A} y_{(a,d)} = 1, \text{ for every } d \in D \\ & \sum_{d \in D} y_{(a,d)} = 1, \text{ for every } a \in A \end{aligned} \tag{Q_2}$$

$$y_{(-a,-d)} = y_{(a,d)}, \text{ for every } a \in A,\ d \in D$$
$$\text{for every } h < i < j \in V \text{ and every } d \in D$$
$$y_{((h,j),d)} = \sum_{e+f=d} y_{((h,i),e)} y_{((i,j),f))}$$
$$y_{(a,d)} \in \{0,1\} \text{ for every } a \in A \text{ and } d \in D$$

We now show that (Q_2) is a good formulation of the turnpike problem, i.e. that for an instance ΔX, the points in (Q_2) correspond to the solutions of ΔX.

Let ΔX be an instance of the turnpike problem of size $\binom{n}{2}$, and let y be a feasible point in Q_2. This point induces a labelling of the edges of a complete graph K_n, such that if $y_{(a,d)} = 1$, the arc a is labelled by the difference d, for $a \in A$ and $d \in D$. Now we can label the vertices of K_n in the following way. We assign label 0 to the starting point s of the arc that is labelled with the maximum difference. If the arc (s,i) is labelled by the difference d we label the vertex i by d. It is easy to check that the labels of the vertices form a solution set for the instance ΔX.

Again we can look at the matrix $Y = \chi^H(\chi^H)^T$ to get the following relaxation of (Q_2):

$$\begin{aligned}
&y_{(a,d_1),(a,d_2)} = 0, \text{ for every } a \in A \text{ and } d_1 \neq d_2 \in D\\
&y_{(a_1,d),(a_2,d)} = 0, \text{ for every } d \in D \text{ and } a_1 \neq a_2 \in A,\\
&\sum_{a\in A} y_{(a,d),(a,d)} = 1, \text{ for every } d \in D\\
&\sum_{d\in D} y_{(a,d),(a,d)} = 1, \text{ for every } a \in A\\
&y_{(-a,-d)(-a,-d)} = y_{(a,d),(a,d)}, \text{ for every } a \in A,\ d \in D\\
&\text{for every } h < i < j \in V \text{ and every } d \in D\\
&\qquad y_{((h,j),d),((h,j),d))} = \sum_{e+f=d} y_{((h,i),e),((i,j),f))}\\
&Y \text{ is positive semidefinite}
\end{aligned} \tag{R_2}$$

The relaxation (R_2) is too big for computational purposes. Also, all the solution functions f described above are equivalent, in the sense that they represent the same solution set X. Let us therefore look at the matrix T which is the average of all matrices

$$n(n-1)P^T\chi^U(\chi^U)^TP$$

where P ranges over all permutation matrices of $A \times D$ such that there exist a permutation π of A and a permutation ρ of D, such that $\rho(d) = d$ for all $d \in D$ or $\rho(d) = -d$ for all $d \in D$ and P permutes $((u, v), d)$ to $((\pi(u), \pi(v)), \rho(d))$.

For each $a = (u, v) \in A$, let $\chi^a \in \mathbb{R}^V$, be defined by:

$$\begin{aligned}\chi^a(v) &= 1\\ \chi^a(u) &= -1\\ \chi^a(w) &= 0, \text{ for } w \neq u, v\end{aligned}$$

For two arcs $a, b \in A$, let

$$\phi(a, b) = (\chi^a)^T\chi^b.$$

Thus, $\phi(a, a) = 2$ and $\phi(a, -a) = -2$. If two arcs a and b are different and have common starting or ending points then $\phi(a, b) = 1$. If two arcs are different and the endpoint of one is the start point of the other, $\phi(a, b) = -1$. And finally, if two arcs have no point in common, $\phi(a, b) = 0$.

The matrix T arises from different labellings of the graphs on Figure 6.1. An element $t_{(a,d),(b,e)}$ of T will be 0 unless arcs a and b in some permutation P coincide with the directed edges labelled d and e. Therefore, $t_{(a,d),(b,e)}$ depends only on $\phi(a, b)$ and d and e, and there exists a number $y_{\phi,d,e}$ such that for $\phi = -2, \ldots, 2$ and $d, e \in D$

$$t_{(a,d),(b,e)} = y_{\phi(a,b),d,e}.$$

Now, for all $d, e \in D$ and all $\phi = -2, \ldots, 2$ the following holds

1. $y_{\phi,d,e} = y_{\phi,e,d}$;

2. $y_{\phi,d,e} = y_{\phi,-e,-d}$;

3. $y_{\phi,d,e} = y_{-\phi,d,-e}$;

4. $y_{2,d,e} = 1$, if $e = d$ and 0 otherwise;
   (To see that $y_{2,d,d} = 1$, note that the number of permutations of $A \times D$ in which a fixed arc is labelled d is $2(n-2)!$.)

5. $y_{-2,d,e} = 1$, if $d = -e$ and 0 otherwise;

6. $y_{\phi,d,e} = 0$, if $\phi = -1, 0, 1$ and $d = \pm e$;

7. If $d, e \in D$ and $d + e \notin D$, then $y_{-1,d,e} = 0$. If $d, e, f \in D$ and $d + e + f = 0$, then $y_{-1,d,e} = y_{-1,e,f} = y_{-1,f,d}$;

8. For every $d \in D$
$$\sum_{e \in D} y_{\phi,d,e} = 1;$$

9. For all $a \in A$ and $d, e \in D$,
$$\sum_{b \in A} t_{(a,d)(b,e)} = 1.$$

We can summarize the above if we introduce the following notation. For $\phi = -2, \ldots, 2$, let Y_ϕ be the $D \times D$ matrix defined by

$$(Y_\phi)_{d,e} = y_{\phi,d,e}$$

for $d, e \in D$. Let r_ϕ be the number of b such that $\phi(a, b) = \phi$, where a is a fixed element of A. Note that this definition does not depend on the choice of A. Then $r_2 = r_{-2} = 1$, $r_1 = 2(n-2)$ because for a fixed a, we can choose an arc b that has the same starting point as a in $(n-2)$ ways and we can choose an arc b that has the same ending point as a in $(n-2)$ ways. Similarly, $r_{-1} = 2(n-2)$. Also $r_0 = (n-2)(n-3)$, because we can choose the endpoints of an arc that is disjoint from a in $(n-2)(n-3)$ ways.

Let P be the $D \times D$ permutation matrix that permutes d to $-d$ for every $d \in D$. Then because of the above

$$
\begin{aligned}
&Y_\phi \text{ is symmetric}\\
&Y_2 = I\\
&Y_{-2} = P\\
&PY_\phi P = Y_\phi, \text{ for } \phi = -2,\dots,2\\
&Y_\phi P = Y_{-\phi}, \text{ for } \phi = -2,\dots,2\\
&\sum_{\phi=-2}^{2} r_\phi Y_\phi = J.
\end{aligned}
$$

where J is the all-one matrix.

For $\phi = -2,\dots,2$, let R_ϕ be the $A \times A$, 0-1 matrix such that $(R_\phi)_{a,b} = 1$ if and only if $\phi(a,b) = \phi$. Note that $R_2 = I$, $R_{-2} = P$, $(n-2)(n-3)R_0$ is a permutation of Y_0, $2(n-2)R_1$ is a permutation of Y_1 and $2(n-2)R_{-1}$ is a permutation of Y_{-1}.

The matrices $R_{-2},\dots,R_2$, form an association scheme. It is easy to check that they satisfy the definition of an association scheme. i.e.

1. $R_2 = I$;
2. $R_i = R_i^T$, for $i \in \{-2,\dots,2\}$;
3. $\sum_{i=-2}^{2} R_i = J$
4. $R_iR_j = \sum_{k=-2}^{2} \alpha_{ij}^k R_k$, for $i,j \in \{-2,\dots,2\}$.

The eigenspaces of this association scheme are:

$$
\begin{aligned}
S_0 &= \{x \mid \forall a,b \in A : x_a = x_b\},\\
S_1 &= \{x \mid \exists p : V \to \mathbb{R} : (p(V) = 0 \text{ and } \forall a = (u,v) : x_a = p(u) + p(v)\},\\
S_2 &= \{x \mid \exists p : V \to \mathbb{R} : (p(V) = 0 \text{ and } \forall a = (u,v) : x_a = p(u) - p(v)\},\\
S_3 &= \{x \mid \forall a \in A : x_a = x_{-a}; \forall v \in V : x(\delta^{in}(v)) = 0\},\\
S_4 &= \{x \mid \forall a \in A : x_a = -x_{-a}; \forall v \in V : x(\delta^{in}(v)) = 0\}.
\end{aligned}
$$

The eigenvalues $\lambda_{i,\phi}$ of R_ϕ corresponding to the eigenspace S_i are given in the following table:

Eigenspace	R_2	R_1	R_0	R_{-1}	R_{-2}	dim
S_0	1	$2(n-2)$	$(n-2)(n-3)$	$2(n-2)$	1	1
S_1	1	$n-4$	$-2(n-3)$	$n-4$	1	$n-1$
S_2	1	$n-2$	0	$-n+2$	-1	$n-1$
S_3	1	-2	2	-2	1	$\frac{1}{2}n(n-3)$
S_4	1	-2	0	2	-1	$\frac{1}{2}(n-1)(n-2)$

Note that $\lambda_{0,\phi} = r_\phi$ for each ϕ.

For every $i = 1, \dots, 4$ we choose a vector $u \in S_i$ such that $||u|| = 1$. Let U be the $(A \times D) \times D$ matrix defined by

$$U_{(a,d),e} = u_a \text{ if } d = e, \text{and 0 otherwise}$$

Let Z be the $D \times D$ matrix defined by

$$Z = U^T T U.$$

So, Z is positive semidefinite and for the elements of Z we have

$$\begin{aligned} Z_{d,e} &= \sum_{a,b \in A} U_{(a,d),d} T_{(a,d),(b,e)} U_{(b,e),e} = \\ &= \sum_{a,b \in A} u_a y_{\phi(a,b),d,e} u_b = \\ &= \sum_{\phi=-2}^{2} y_{\phi,d,e} u^T R_\phi u = \\ &= \sum_{\phi=-2}^{2} y_{\phi,d,e} \lambda_{i,\phi}, \end{aligned}$$

and therefore

$$Z = \sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_\phi.$$

This gives us four positive semidefinite constraints for combinations of Y_ϕ.

Note that if we know the matrix Y_{-1} we know all the matrices Y_ϕ.

Also if $i = 1$ or $i = 3$, $\lambda_{i,\phi} = \lambda_{i,-\phi}$ for each ϕ and

$$ZP = \sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_\phi P = \sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_{-\phi} = \sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_\phi = Z \qquad (6.1)$$

so $Z_{d,e} = Z_{-d,e}$ for all d, e and the condition on positive semidefinitness of Z is equivalent to that of a $\frac{1}{2}|D| \times \frac{1}{2}|D|$ submatrix.

Similarly, if $i = 2$ or $i = 4$, $\lambda_{i,\phi} = -\lambda_{i,-\phi}$ and

$$ZP = \sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_\phi P = -\sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_{-\phi} = -\sum_{\phi=-2}^{2} \lambda_{i,\phi} Y_\phi = -Z \qquad (6.2)$$

so $Z_{d,e} = -Z_{-d,e}$ for all d, e and the condition on positive semidefinitness of Z is equivalent to that of a $\frac{1}{2}|D| \times \frac{1}{2}|D|$ submatrix.

6.3 Implementation

In this section we show how to implement the above relaxation.

First note that matrices Y_ϕ, for $\phi = -2, \ldots, 2$ are indexed by the set D.

Because of the properties 1, 2, and 3 of the entries $y_{\phi,d,e}$ of the matrices Y_ϕ, $\phi = -1, 1$, these matrices have the following form

$$Y_1 = \begin{bmatrix} A & B \\ B & A \end{bmatrix}$$

and

$$Y_{-1} = \begin{bmatrix} B & A \\ A & B \end{bmatrix}$$

for some $\Delta X \times \Delta X$ matrices A and B.

Also,

$$Y_2 = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$$

and

$$Y_{-2} = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$$

and

$$Y_0 = \begin{bmatrix} C & C \\ C & C \end{bmatrix}$$

for some $\Delta X \times \Delta X$ matrix C.

Now, because of (6.1) and (6.2), we have that

$$\begin{aligned} &I + 2(n-2)A + (n-2)(n-3)C + 2(n-2)B = J \\ &I + (n-4)A - 2(n-3)C + (n-4)B \geq 0 \\ &I + (n-2)A - (n-2)B \geq 0 \\ &I - 2A + 2C - 2B \geq 0 \\ &I - 2A + 2B \geq 0. \end{aligned} \tag{6.3}$$

Also, for the elements of the matrices A and B because of the property 8 from the previous section, we have

$$\begin{aligned} &\sum_{i \in \Delta X} (a_{ij} + b_{ij}) = 1 \text{ for } j \in \Delta X \\ &\sum_{i \in \Delta X} 2c_{ij} = 1. \end{aligned} \tag{6.4}$$

From property 7, if $d, e \in D$ and $d + e \notin D$, then if $d > 0$ and $e > 0$,

$$b_{d,e} = 0, \tag{6.5}$$

and if $d > 0$ and $e < 0$ or $d < 0$ and $e > 0$

$$a_{|d|,|e|} = 0. \tag{6.6}$$

If $d, e, f \in D$ and $d + e + f = 0$ then if the sign of f is different than the sign of d and e,

$$\begin{aligned} b_{|d|,|e|} &= a_{|d|,|f|} \\ b_{|d|,|e|} &= a_{|e|,|f|}. \end{aligned} \tag{6.7}$$

If the sign of e is different than the sign of d and f,

$$\begin{aligned} b_{|d|,|f|} &= a_{|d|,|e|} \\ b_{|d|,|f|} &= a_{|e|,|f|}. \end{aligned} \tag{6.8}$$

If the sign of d is different than the sign of e and f,

$$\begin{aligned} b_{|e|,|f|} &= a_{|d|,|e|} \\ b_{|e|,|f|} &= a_{|d|,|f|}. \end{aligned} \tag{6.9}$$

We can now combine (6.3), (6.4), (6.5), (6.6), (6.7), (6.8) and (6.9) into a semidefinite program (R_3).

Next, we need to see that the program (R_3) on an instance ΔX of a turnpike problem is a relaxation of the problem, in the sense that all $0-1$ solutions of (R_3) correspond to the solutions of ΔX.

Because, of (6.4) we see that matrices A, B and C are not $0-1$ matrices and we will instead look at the matrices

$$\begin{aligned} A' &= 2(n-2)A, \\ B' &= 2(n-2)B, \\ C' &= (n-2)(n-3)C. \end{aligned}$$

We look at these matrices because if Y_{-1} corresponds to a solution of a turnpike instance of size n, from the above construction we can see that the entries of Y_{-1} are 0 and $\frac{1}{2(n-2)}$.

Now we prove that if A', B' and C' are $0-1$ matrices, the matrices Y_ϕ for $\phi = -1, 0, 1$ correspond to a solution of a turnpike instance.

The positive-semidefinite constraint from (R_3)

$$I + (n-2)A - (n-2)B \succeq 0,$$

can be written in terms of A' and B' as

$$D = 2I + A' - B' \succeq 0.$$

Now, the row of B' indexed by the largest element M of ΔX must sum to 0 because of (6.7), (6.8) and (6.9). Therefore because of (6.4), the elements of the row of A' indexed by the largest element sum to $2(n-2)$.

If the size of a solution set, n is 3, it is easy to see that if the matrices A', B' and C' are $0-1$ matrices, that they correspond to a solution of the instance.

If $n \geq 4$, there exists an element x such that $a_{u,x} = 1$ and $a_{v,x} = 1$ for some $u, v \in \Delta X$. But then also $a_{x-u,x} = 1$, $a_{x-v,x} = 1$, $b_{u,x-u} = 1$ and $b_{v,x-v} = 1$.

Now we look at the submatrix of D indexed by x, u, v and $x-v$. This matrix has the form:

$$E = \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & a_1 - b_1 & a_2 - b_2 \\ 1 & a_1 - b_1 & 2 & -1 \\ 1 & a_2 - b_2 & -1 & 2 \end{bmatrix} \tag{6.10}$$

where a_1, a_2, b_1, b_2 are either 0 or 1 and $a_i + b_i \leq 1$, because of $I + A' + C' + B' = J$.

Therefore, for the numbers a_1, a_2, b_1, b_2 we have the following possibilities:

1. $a_1 = 0$, $a_2 = 0$, $b_1 = 0$,$b_2 = 0$;
2. $a_1 = 0$, $a_2 = 0$, $b_1 = 1$,$b_2 = 0$;
3. $a_1 = 0$, $a_2 = 0$, $b_1 = 0$,$b_2 = 1$;
4. $a_1 = 0$, $a_2 = 0$, $b_1 = 1$,$b_2 = 1$;
5. $a_1 = 1$, $a_2 = 0$, $b_1 = 0$,$b_2 = 0$;
6. $a_1 = 1$, $a_2 = 0$, $b_1 = 0$,$b_2 = 1$;
7. $a_1 = 0$, $a_2 = 1$, $b_1 = 0$,$b_2 = 0$;
8. $a_1 = 0$, $a_2 = 1$, $b_1 = 1$,$b_2 = 0$;
9. $a_1 = 1$, $a_2 = 1$, $b_1 = 0$,$b_2 = 0$.

The matrix E is positive-semidefinite only in cases 5 and 7 above, which we verify using some computational tool, such as *matlab*.

If we look at the submatrix of E indexed by x, u, v, $x-v$ and $x-u$, this matrix has the form

$$E = \begin{bmatrix} 2 & 1 & 1 & 1 & 1 \\ 1 & 2 & a_1 & a_2 & -1 \\ 1 & a_1 & 2 & -1 & a_3 \\ 1 & a_2 & -1 & 2 & a_4 \\ 1 & -1 & a_3 & a_4 & 2 \end{bmatrix}, \tag{6.11}$$

where a_i for $i = 1, \dots, 4$ is 0 or 1 and $a_1 + a_2 = 1$ and similarly $a_3 + a_4 = 1$. Again, we use *matlab* to see that positive-semidefinetness of D implies that $a_1 = a_4$ and $a_2 = a_3$.

Hence, we showed that if $a_{u,x} = 1$ and $a_{v,x} = 1$, for some $u, v, x \in D$, then either $a_{u,v} = 1$, $a_{x-u,x-v} = 1$, $a_{u,x-v} = 0$ and $a_{v,x-u} = 0$ or $a_{u,x-v} = 1$, and $a_{v,x-u} = 1$, $a_{u,v} = 0$ and $a_{x-u,x-v} = 0$.

Therefore, the entries of the row of A' indexed by the largest element of ΔX are split into two classes, and they determine all the other elements of A' and B'. We show that the elements of each class determine a solution of the turnpike instance. The solution determined by one class is obviously a mirror image of the solution determined by the other class.

To see that each class determines a solution of the turnpike instance, let us assume that one of the classes is $Y = \{x_1, \dots, x_{n-1}\}$. Then because of the above $a_{x_i, x_i - x_j} = 1$ for $i > j$, and because of (6.6) $x_i - x_j \in \Delta X$. Therefore, $\Delta Y \subseteq \Delta X$. Now, because of (6.4) we can see that each difference $x_i - x_j$ in ΔY appears at most once, and therefore $\Delta X = \Delta Y$, which proves that the scaled $0-1$ solutions of the relaxation (R_3) of an instance ΔX of the turnpike problem, correspond to the solutions of the instance.

It is easy to construct $0-1$ matrices A', B' and C' that satisfy constraints (6.4), (6.5), (6.6), (6.7), (6.8) and (6.9) but not positive-semidefinite constraints (6.3), that do not correspond to a solution of an instance of the turnpike problem.

We also implemented the relaxation (R_3) using the semidefinite program solver SDPSOL [6]. Finding an instance that can not be solved by this relaxation is not hard. For example, the instance

$$\Delta X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 15, 16, 18\}$$

is not a difference set, but its relaxation (R_3) contains a feasible point. The matrices A and B that are feasible for the relaxation (R_3) of this instance are convex combination of $0-1$ matrices A_i, $i = 1, \ldots, k$ and B_i, $i = 1, \ldots, k$, respectively These matrices correspond to the solutions of subinstances of ΔX. It is easy to see that the equality constraints of (R_3) hold for these $0-1$ matrices, as do the constraints

$$I - 2A_i + 2C_i - 2B_i \geq 0$$
$$I - 2A_i + 2B_i \geq 0,$$

for $i \in \{1, \ldots, k\}$.

The constraints

$$I + (n-4)A + 2(n-3)C + (n-4)B \geq 0$$
$$I + (n-2)A - (n-2)B \geq 0$$

do not hold for each pair of $0-1$ matrices separately, but they do hold for the convex combination of sufficiently large number of $0-1$ matrices.

In this respect, the relaxation (R_3) does not seem to be very powerful.

6.4 A Connection Between (R_3) and (S_1)

To finish, we mention one more property of the matrices $A' + B'$ and $A' - B'$. If $X = \{0, x_1, \ldots, x_{n-1}\}$ is a solution of a turnpike instance ΔX, we call the sets X, $X - x_1$, $X - x_2$, $\ldots$, $X - x_{n-1}$ the *streaks* of X.

If A' and B' correspond to the solution X, then

$$A' + B' = u_0 u_0^T + \ldots + u_{n-1} u_{n-1}^T - nI, \tag{6.12}$$

where u_i is the characteristic vector of size x_{n-1}, of the set that contains absolute values of the elements of the set $X - x_i$.

Also

$$A' - B' = v_0 v_0^T + \ldots + v_{n-1} v_{n-1}^T - nI, \tag{6.13}$$

where v_i is a vector of size x_{n-1}, whose entries are 0,1, -1, and v_i has 1 on the position indexed by the difference $x_a - x_b$, $a > b$, if $x_a - x_b$ is in the steak $X - x_i$. The vector v_i has -1 on the position indexed by the difference $x_a - x_b$, $a > b$, if $-(x_a - x_b)$ is in the steak $X - x_i$

Notice that the matrices $u_0 u_0^T$ and $v_{n-1} v_{n-1}^T$ are submatrices of a feasible matrix for the relaxation (S_1).

This fact can be used to strengthen the constraints of the relaxation (R_3), i.e. we can constrain $A' + B'$ and $A' - B'$ to be of the form (6.12) and (6.13), respectively.

Chapter 7

Conclusions

In this thesis we considered the turnpike problem. Although the major open question, whether the problem is in the class P of problems solvable in polynomial time, is open, we have presented methods for solving some classes of instances in polynomial time. These classes include the class of instances constructed by Zhang, [35] on which Skiena's et al. backtracking procedure takes exponential time. There is no other known class of instances on which the backtracking procedure takes exponential time.

Our methods are based on representing the turnpike problem as a $0-1$ quadratic program which is then relaxed to a semidefinite program that can be solved in polynomial time. We represent the turnpike problem as a $0-1$ quadratic program in three different ways. For one such representation, we consider a sequence of semidefinite relaxations similar to the sequence of semidefinite relaxations proposed and used by Lovász and Schrijver in [19] to construct an algorithm for finding maximum stable sets in perfect graphs. We do not have an instance which would not be solved by the second semidefinite relaxation in the sequence. We prove that there exists a polynomial time algorithm for solving the turnpike problem on classes of instances for which there exist a constant c, such that the instances are solved by the c-th semidefinite relaxation in the sequence.

Finding instances for which the constant c is greater than two would be interesting because Lovász and Schrijver do not have a class of graphs for which the maximum stable sets could not be found by a semidefinite relaxation which corresponds to the second relaxation in our sequence.

We also performed extensive numerical testing of our methods. Since we approach the turnpike problem from the theoretical computing science viewpoint, our numerical results are obtained by examining all instances with some given properties. It would be interesting to see how our methods behave on the instances that arise from partial digest experiments.

Bibliography

[1] F. ALIZADEH, Interior Point Methods in Semidefinite Programming with Applications to Combinatorial Optimization, *SIAM J. Optimization*, Vol. 5, No. 1, 13-51, February 1995

[2] F. ALIZADEH, J.P. A. HAEBERLY AND M. OVERTON, Complementarity and Nondegeneracy in Semidefinite Programming, *Math. Programming*, Vol. 77, 111-128, 1997.

[3] F. ALIZADEH, J.P. A. HAEBERLY AND M. OVERTON, Primal-Dual Interior Point Methods for Semidefinite Programming, *Presented at the XV Symposium on Mathematical Programming, Ann Arbor, August 1994.*

[4] F. ALIZADEH AND HIS STUDENTS, Lecture Notes on Semidefinite Programming, available from the SDP homepage http://rutcor.rutgers.edu/~alizadeh/CLASSES/97sprSDP/NOTES/

[5] E, BALAS, S. CERIA, G. CORNUEJOLS, G. PATAKI Polyhedral Methods for Maximum Clique Problem, *Technical report*, Carnegie Mellon University, Pittsburg, PA, 1994

[6] S.P. WU AND S. BOYD SDPSOL: a parser/solver for SDP and MAXDET problems with matrix structure. Available from http://www.stanford.edu/~boyd/SDPSOL.html

[7] T. I DIX, D. H. KIERONSKA Errors between sites in restriction site mapping, *Comput. Appl. Biol. Sci.*, 4,1 (1988) 117-123

[8] A. FRIEZE AND M. JERRUM, Improved approximation algorithms for MAX k-cut and MAX BISECTION, *IPCO IV Proc., LNCS 920,* Springer 1995, 1-13

[9] M. X. GOEMANS AND D. P. WILLIAMSON, 0.878 Approximation Algorithm for Max CUT and Max 2SAT, *26th Annual Symposium on the Theory of Computing,* 422-431, Montreal 1994

[10] M. X. GOEMANS AND D. P. WILLIAMSON, Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming, *J. Assoc. Comput. Mach. 42 (1995),* no. 6, 1115-1145

[11] M. X. GOEMANS AND D. P. WILLIAMSON, A New 3/4-Approximation Algorithm for MAX SAT, *SIAM J. Disc. Math* 7(4), 656-666, November 1994.

[12] M. GRÖTSCHEL, L. LOVÁSZ AND SCHRIJVER, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica,* 1, 169-197, 1981

[13] M. GRÖTSCHEL, L. LOVÁSZ AND SCHRIJVER, Geometric Algorithms and Combinatorial Optimization, Springer-Verlag, Berlin 1988

[14] C. HELMBERG, S. POLJAK, F. RENDL, H. WOLKOWITZ Combining Semidefinite and Polyhedral Relaxations for Integer Programs, *Integer programming and combinatorial optimization (Copenhagen, 1995),* 124-134, Lecture Notes in Comput. Sci., 920, Springer, Berlin, 1995

[15] R. HORN AND C. JOHNSON, Matrix Analysis, Cambridge University Press

[16] P. LEMKE, M. WERMAN, On the complexity of inverting the autocorrelation function of a finite integer sequence, and the problem of locating n points on a line, given the unlabeled distances between them, *unpublished manuscript* (1988)

[17] A. K . LENSTRA, H. W. LENSTRA, L. LOVÁSZ, Factoring polynomials with rational coefficients, *Math. Ann.* 261 (1982), 515-534

[18] L. LOVÁSZ, On the Shannon capacity of a graph, *IEEE Transactions on Information Theory,* IT-25, 1-7, 1979

[19] L. LOVÁSZ AND A. SCHRIJVER, Cones of matrices and set-functions and 0-1 optimization, *SIAM Journal on Optimization*, 1, 166-190, 1990

[20] D. KARGER, R. MOTWANI AND M.SUDAN, approximate Graph Coloring by Semidefinite Programming, *35th IEEE Symposium on Foundations of Computer Science*, 1-10, October 1994

[21] S. MAHAJAN AND H. RAMESH, Correctly Derandomizing Goemans and Williamson's Max CUT algorithm, *STOC 95*

[22] T. ONO, T. HIRATA AND T. ASANO, An Approximation Algorithm for MAX 3-SAT, *In Proceedings of the 6th International Symposium, ISAAC '95 Cairns, Australia,* December 1995

[23] G.PATAKI, On the facial structure of cone-LP's and semi-definite programs, *Technical Report*, Graduate School of Industrial Administration, Carnegie Mellon University, Number 595, 1994.

[24] A. L. PATTERSON, A direct method for determination of the components of interatomic distances in crystals, *Zeitschr. Krist.* 90 (1935) 517-542

[25] A. L. PATTERSON, Ambiguities in the X-ray analysis of crystal structure, *Phys. Review.* 65 (1944) 195-201

[26] S. PICCARD, Sur les ensembles de Distances des Ensembles de points d'un Espace Euclidean, crystal structure, *Mem. Uni. Neuchatel* 13, Neuchatel Switzerland 1939

[27] R. T. ROCKAFELLAR, Convex Analysis, Princeton University Press 1970

[28] J. ROSENBLATT, P. D. SEYMOUR , The structure of Homometric Sets, *SIAM J. Alg. Disc. Meth*, Vol 3, No 3, September 1993

[29] A.SCHRIJVER Personal communication

[30] M. I. SHAMOS: Problems in computational geometry, *Unpublished manuscript, Carnegie Mellon University*, 1977

[31] S. S. SKIENA, W. D. SMITH, AND P. LEMKE, Reconstructing sets from interpoint distances, *Proc. Sixth ACM Symp. Computational Geometry*, 332 - 339, 1990.

[32] S. S. SKIENA, G. SUNDARAM, A Partial Digest Approach to restriction Site Mapping, *Bulletin of Mathematical Biology*, 56 (1994) 275-294

[33] L. VANDENBERGHE AND S. BOYD, Semidefinite Programming, *SIAM Rev.* 38 (1996), no. 1, 49-95

[34] Y. YE, A Class of Projective Transformations for Linear Programming, *SIAM J. Computing* 19 (1990), 457-466

[35] Z. ZHANG, An exponential example for partial digest mapping algorithm, *Journal of Computational Biology* 3 (1994), 235-239