

**CONVEX OPTIMIZATION AND INTERIOR POINT  
METHODS**

**FINAL PROJECT**

**0/1 INTEGER  
PROGRAMMING AND  
SEMIDEFINITE  
PROGRAMMING**

by

Lucia Buch and Natalia Viktorovna

## CONTENTS:

1.Introduction

2.Formulation

3.Application to Knapsack Problem

4.Cutting Planes and Lift-and-Project Methods

4.1 Balas-Ceria-Cornuéjols Construction and Example

4.2 Lovász-Schrijver Construction

4.3 Sherali-Adams Construction

5.Worst Case Examples

6.References

## 1. INTRODUCTION

The definition of a 0/1 integer linear programming model is the following:

$$\max c^T x \quad \text{subject to} \quad Ax \leq b, x \in \{0,1\}^n \quad (1.1)$$

Before going any further, the convex hull has to be defined as well:

$$\begin{aligned} \max c^T x \quad \text{subject to} \quad x \in P, \\ P := \text{conv}\left(\left\{ x \in \{0,1\}^n \mid Ax \leq b \right\}\right) \end{aligned} \quad (1.2)$$

Usually, the problems represented in this way are NP-Hard, which makes them very difficult to solve. It is a common approach to use LP modeling stated below:

$$K := \left\{ x \in \mathbb{R}_+^n \mid Ax \leq b \right\}$$

However, if K does not represent the convex hull of the feasible region, we will not get a feasible 0/1 solution. The solution that we obtain is then called LP Relaxation. If the polytope K is the same as the convex hull, we will get the optimal solution just by solving the corresponding relaxation. In order to find K that will be the same as P, one of the most common methods is the cutting planes method which will be described in this project.

Before that, we will introduce an alternative formulation for the 0/1 Programming Model, using semidefinite programming.

## 2. FORMULATION

Let the matrix Y be defined as following:

$$Y = \begin{bmatrix} 1 \\ x \end{bmatrix} \begin{bmatrix} 1 & x^T \end{bmatrix} = \begin{bmatrix} 1 & x^T \\ x & xx^T \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_2 & \cdots & x_n \\ x_1 & x_1^2 & x_1x_2 & \cdots & x_1x_n \\ x_2 & x_1x_2 & x_2^2 & \cdots & x_2x_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n & x_1x_n & x_2x_n & \cdots & x_n^2 \end{bmatrix} \quad (2.1)$$

We will define the matrix X, as a submatrix of Y. The matrix X is:

$$X = xx^T = \begin{bmatrix} x_1^2 & x_1x_2 & \cdots & x_1x_n \\ x_1x_2 & x_2^2 & \cdots & x_2x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1x_n & x_1x_n & \cdots & x_n^2 \end{bmatrix} \quad (2.2)$$

In order to model the Objective Function, we will use the trace operator,

$$\max \text{trace}(\text{diag}(c) \cdot X) \quad (2.3)$$

which in the common definition represents:

$$\max \sum_{i=1}^n c_i x_i^2 \quad (2.4)$$

The difference in both objective functions is the square variable  $x$ . However, this does not play an important role due to the fact that  $x$  is binary.

In order to model the 0/1 constraint, we have two possible alternatives.

#### *Alternative 1*

First of all, we use the fact that  $Y$  is positive semidefinite, so we state:

$$Y = \begin{bmatrix} 1 & x_1 & x_2 & \cdots & x_n \\ x_1 & x_1^2 & x_1 x_2 & \cdots & x_1 x_n \\ x_2 & x_1 x_2 & x_2^2 & \cdots & x_2 x_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n & x_1 x_n & x_2 x_n & \cdots & x_n^2 \end{bmatrix} \succeq 0 \quad (2.5)$$

This is true always due to the 1-rank matrixes behavior. The following constraint is totally necessary:

$$Y_{ii} = Y_{0i}, \quad \forall i = 1, \dots, n \quad (2.6)$$

It makes the 0/1 constraint:  $Y_{ii} = Y_{0i} \Rightarrow x_i = x_i^2$

#### *Alternative 2*

In this alternative, we define differently the matrix  $Y$ . In other words, we plug in the constraint (2.6) into the  $Y$  matrix obtaining the following constraint:

$$Y' = \begin{bmatrix} 1 & x_1^2 & x_2^2 & \cdots & x_n^2 \\ x_1^2 & x_1^2 & x_1 x_2 & \cdots & x_1 x_n \\ x_2^2 & x_1 x_2 & x_2^2 & \cdots & x_2 x_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n^2 & x_1 x_n & x_2 x_n & \cdots & x_n^2 \end{bmatrix} \succeq 0 \quad (2.7)$$

There are 3 alternatives to represent the  $Ax \leq b$  system of inequalities. Let's consider only one inequality  $a^T x \leq \beta$  from  $Ax \leq b$

*Alternative 1.*

$$\text{trace}(\text{diag}(a) * X) \leq \beta \implies \max \sum_{i=1}^n a_i x_i^2 \quad (2.8)$$

*Alternative 2.*

$$\text{trace}(aa^T * X) \leq \beta^2 \text{ for } a, \beta \geq 0 \implies \left( \sum_{i=1}^n a_i x_i \right)^2 \leq \beta^2 \quad (2.9)$$

*Alternative 3.*

$$\text{trace}\left(\left(aa^T - \beta \cdot \text{diag}(a)\right) \cdot X\right) \leq 0 \text{ for } a, \beta \geq 0 \implies \sum_{i=1}^n (a_i x_i)^2 \leq \beta \sum_{i=1}^n (a_i x_i) \quad (2.10)$$

All these alternatives are equivalent.

### **3. NUMERICAL EXAMPLE**

In this section we will apply the constructions provided in the previous section, in order to formulate a small instance of the Knapsack problem as a semidefinite programming.

The instance is:

$$\begin{aligned} \max & 3x_1 + 4x_2 + x_3 \\ \text{s.t.} & 5x_1 + 6x_2 + 2x_3 \leq 10 \\ & x_1, x_2 \in \{0,1\} \end{aligned}$$

The semidefinite formulation will be the following:

$$\max \text{trace} \left( \begin{bmatrix} 3 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1^2 & x_1 x_2 & x_1 x_3 \\ x_1 x_2 & x_2^2 & x_2 x_3 \\ x_1 x_3 & x_2 x_3 & x_3^2 \end{bmatrix} \right)$$

Subject to:

$$\begin{bmatrix} 1 & x_1^2 & x_2^2 & x_3^2 \\ x_1^2 & x_1^2 & x_1x_2 & x_1x_3 \\ x_2^2 & x_1x_2 & x_2^2 & x_2x_3 \\ x_3^2 & x_1x_3 & x_2x_3 & x_3^2 \end{bmatrix} \succeq 0$$

and

$$\text{trace} \left( \begin{bmatrix} 5 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_1^2 & x_1x_2 & x_1x_3 \\ x_1x_2 & x_2^2 & x_2x_3 \\ x_1x_3 & x_2x_3 & x_3^2 \end{bmatrix} \right) \leq 10$$

or

$$\text{trace} \left( \begin{bmatrix} 25 & 30 & 10 \\ 30 & 36 & 12 \\ 10 & 12 & 4 \end{bmatrix} \cdot \begin{bmatrix} x_1^2 & x_1x_2 & x_1x_3 \\ x_1x_2 & x_2^2 & x_2x_3 \\ x_1x_3 & x_2x_3 & x_3^2 \end{bmatrix} \right) \leq 100$$

or

$$\text{trace} \left( \begin{bmatrix} 25 & 30 & 10 \\ 30 & 36 & 12 \\ 10 & 12 & 4 \end{bmatrix} \cdot \begin{bmatrix} x_1^2 & x_1x_2 & x_1x_3 \\ x_1x_2 & x_2^2 & x_2x_3 \\ x_1x_3 & x_2x_3 & x_3^2 \end{bmatrix} \right) \leq 10 \cdot \text{trace} \left( \begin{bmatrix} 5 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_1^2 & x_1x_2 & x_1x_3 \\ x_1x_2 & x_2^2 & x_2x_3 \\ x_1x_3 & x_2x_3 & x_3^2 \end{bmatrix} \right)$$

#### **4. CUTTING PLANES AND LIFT-AND-PROJECT METHODS**

The 0/1 linear program that we are going to use in this section looks like

$$\max c^T x \quad \text{subject to} \quad Ax \leq b, x \in \{0,1\}^n.$$

The reformulation of this problem is in (1.2).

As explained in the introduction of this project, the main idea behind the relaxation methods is to get as close as possible to the convex hull representation of the feasible region, e.g. we want to get a linear description or at least good linear relaxations of P.

If the original relaxation does not have that property, we can construct some cutting planes in order to eliminate the non-integer extreme points of the relaxation.

The technique that this project presents for the construction of such cutting planes is the Lift-and-Project method.

In general, those methods seek to find a polytope Q, whose projection P would represent the convex hull we look for. The main advantage of these methods is the fact that the Q can have less facets than the projection itself. This allows us to get a

polynomial polytope  $Q$ , even if the convex hull  $P$  is not. If we know that  $Q$  has a polynomial number of facets and its dimension is polynomial in the dimension of  $P$ , then our linear optimization problem over  $P$  is solvable in polynomial time. This is when lift-and-project method come into play.

The easiest linear relaxation which we are going to start with is

$$K = \{x \in [0,1]^n \mid Ax \leq b\}$$

and we want to add constraints such that we end up in

$$P = \text{conv}(K \cap \{0,1\}^n).$$

In this section we are going to present three lift-and-project methods that start with the polyhedron  $K$  and allow us to get the convex hull  $P$  of our problem (1.2) which means that the optimal solution of  $\max c^T x$  subject to  $Ax \leq b$  is attained for Boolean values of the components of  $x$ . These three methods are the Balas-Ceria-Cornuéjols method (BCC), also known as lift-and-project for BCC, the Lovász-Schrijver method (LS), also called matrix-cuts for LS, and the Sherali-Adams method (SA), also called Reformulation-Linearization Technique for SA. These constructions proceed the following way: Multiply each row of the system  $Ax \leq b$  by various products of  $x_i$  and  $1 - x_i$ , then substitute the products  $x_i x_j$ ,  $i \neq j$ , for a new variable  $y_{ij}$  and substitute the squares  $x_j^2$  for  $x_j$ . The resulting polyhedron lies in a higher dimensional space than the original problem. Therefore, we project the polyhedron onto  $\mathbb{R}^n$ , because the projection is still contained in  $K$  and contains  $P$ . After at most  $n$  iterations the projection is equal to the convex hull  $P$  of problem (1.2).

We are going to assume without loss of generality that the inequalities  $0 \leq x_i \leq 1$  for  $i \in \{1, 2, \dots, n\}$  are included in  $Ax \leq b$ .

#### 4.1 Balas-Ceria-Cornuéjols Construction and Example

The construction of the BCC projections goes as follows:

Step 1: Fix an index  $j \in \{1, 2, \dots, n\}$ .

Step 2: Achieve the nonlinear system

$$(Ax - b) * x_j \leq 0$$

$$(Ax - b) * (1 - x_j) \leq 0$$

by multiplying the original system  $Ax \leq b$  by  $x_j$  and respectively  $1 - x_j$ .

Step 3: Substitute  $y_i$  for  $x_i x_j$  ( $i=1, \dots, n$ ,  $i \neq j$ ) and  $x_j$  for  $x_j^2$  in order to receive the linearized system

$$Ay - bx_j \leq 0$$

$$A(x - y) - b(1 - x_j) \leq 0$$

Step 4: Project the resulting polytope onto  $\mathbb{R}^n$  by eliminating  $y$  in the linearized system.

The projection  $P_j(K)$  satisfies  $P \subseteq P_j(K) \subseteq K$ .

Step 5: Define  $P_{h,j}(K) := P_h(P_j(K))$

and iterate from step 1 on with index  $h \neq j$  until every element of  $\{1, 2, \dots, n\}$  has occurred exactly once.

The main purpose of this algorithm is to give the inequalities that describe the convex hull of the set of feasible solutions in problem (1.2) and we reach the convex hull within  $n$  iterations.

We have

$$\begin{aligned}
 P_j(K) &= \text{conv}(K \cap \{x \mid x_j \in \{0,1\}\}) \\
 P_{h,j}(K) &:= P_h(P_j(K)) = \text{conv}(K \cap \{x \mid x_h \in \{0,1\}, x_j \in \{0,1\}\}) \\
 P_{1,2,\dots,n}(K) &= \text{conv}(K \cap \{0,1\}^n)
 \end{aligned}$$

and therefore we obtain the inclusions  $K \supseteq P_j(K) \supseteq P_{h,j}(K) \supseteq \dots \supseteq P_{1,2,\dots,n}(K) = P$

*Example:*

One simple example to illustrate the algorithm is one knapsack problem in  $n=3$  variables.

$$\begin{array}{ll}
 \max 3x_1 + 4x_2 + x_3 & \max 3x_1 + 4x_2 + x_3 \\
 \text{s.t. } 5x_1 + 6x_2 + 2x_3 \leq 10 & \text{with its linear relaxation K s.t. } 5x_1 + 6x_2 + 2x_3 \leq 10 \\
 x_1, x_2, x_3 \in \{0,1\} & x_i \leq 1 \text{ for } i \in \{1,2,3\} \\
 & -x_i \leq 0 \text{ for } i \in \{1,2,3\}
 \end{array}$$

The first iteration goes as following:

Step 1: Choose  $j \in \{1,2,3\}$ , WLOG  $j=1$ .

Step 2: Multiply the constraints by  $x_1$  and  $1 - x_1$  to obtain the system of nonlinear equations

$$\begin{array}{ll}
 (5x_1 + 6x_2 + 2x_3) x_1 & \leq 10x_1 \\
 (5x_1 + 6x_2 + 2x_3) (1 - x_1) & \leq 10(1 - x_1) \\
 x_1 x_1 & \leq 1x_1 \\
 x_1 (1 - x_1) & \leq 1(1 - x_1) \\
 -x_1 x_1 & \leq 0x_1 \\
 -x_1 (1 - x_1) & \leq 0(1 - x_1) \\
 x_2 x_1 & \leq 1x_1 \\
 x_2 (1 - x_1) & \leq 1(1 - x_1) \\
 -x_2 x_1 & \leq 0x_1 \\
 -x_2 (1 - x_1) & \leq 0(1 - x_1) \\
 x_3 x_1 & \leq 1x_1 \\
 x_3 (1 - x_1) & \leq 1(1 - x_1) \\
 -x_3 x_1 & \leq 0x_1 \\
 -x_3 (1 - x_1) & \leq 0(1 - x_1)
 \end{array}$$

Step 3: Substitute  $y_i$  for  $x_1 x_i$  and  $x_i$  for  $x_1^2$ .

$$\begin{array}{ll}
 -5x_1 & + 6y_2 + 2y_3 & \leq 0 \\
 10x_1 + 6x_2 + 2x_3 - 6y_2 - 2y_3 - 10 & \leq 0 \\
 & 0 \leq 0 \sqrt{\phantom{x}} \\
 x_1 - 1 & & \leq 0 \\
 -x_1 & & \leq 0 \\
 & 0 \leq 0 \sqrt{\phantom{x}} \\
 -x_1 & + y_2 & \leq 0 \\
 x_1 + x_2 & - y_2 & - 1 \leq 0 \\
 & -y_2 & \leq 0 \\
 -x_2 & + y_2 & \leq 0 \\
 -x_1 & & + y_3 \leq 0 \\
 x_1 & + x_3 & - y_3 - 1 \leq 0 \\
 & & -y_3 \leq 0 \\
 & -x_3 & + y_3 \leq 0
 \end{array}$$

Step 4: Project the system from step 3 onto  $\mathbb{R}^3$  by eliminating  $y_i, i \in \{1,2,3\} \setminus \{1\} = \{2,3\}$ .

WLOG start the elimination by eliminating  $y_3$ .

The inequalities containing  $y_3$  are

$$\begin{array}{rcl}
 -5x_1 & + & 6y_2 + 2y_3 \leq 0 \\
 10x_1 + 6x_2 + 2x_3 - 6y_2 - 2y_3 - 10 & \leq & 0 \\
 -x_1 & + & y_3 \leq 0 \\
 x_1 & + & x_3 - y_3 - 1 \leq 0 \\
 & & -y_3 \leq 0 \\
 & & -x_3 + y_3 \leq 0
 \end{array}
 \rightarrow
 \begin{array}{rcl}
 y_3 & \leq & \frac{1}{2}(5x_1 - 6y_2) \\
 y_3 & \leq & x_1 \\
 y_3 & \leq & x_3 \\
 \frac{1}{2}(10x_1 + 6x_2 + 2x_3 - 6y_2 - 10) & \leq & y_3 \\
 x_1 + x_3 - 1 & \leq & y_3 \\
 0 & \leq & y_3
 \end{array}$$

$$\rightarrow
 \begin{array}{l}
 y_3 \leq \min\{\frac{1}{2}(5x_1 - 6y_2), x_1, x_3\} \\
 y_3 \geq \max\{\frac{1}{2}(10x_1 + 6x_2 + 2x_3 - 6y_2 - 10), x_1 + x_3 - 1, 0\}
 \end{array}$$

Now we are going to write down all the possible inequalities that are implicated in  $\max\{\dots\} \leq \min\{\dots\}$

$$\begin{array}{rcl}
 x_1 \leq 1 & & 8x_1 + 6x_2 + 2x_3 - 6y_2 - 10 \leq 0 \\
 x_1 \geq 0 & & y_2 \leq x_1 \\
 x_3 \leq 1 & & 10x_1 + 6x_2 - 6y_2 - 10 \leq 0 \\
 x_3 \geq 0 & & y_2 \geq 0 \\
 & & 3x_1 - 2x_2 - 6y_2 + 2 \leq 0 \\
 & & x_1 + x_2 - y_2 - 1 \leq 0 \\
 5x_1 + 6x_2 + 2x_3 \leq 10 & & 5x_1 - 6y_2 \leq 0
 \end{array}$$

Eliminate the remaining  $y$ -variable  $y_2$  in the new system that we received by the  $\max\{\dots\} \leq \min\{\dots\}$  combinations like done above and adding system from step 3 without the inequalities containing  $y_3$

$$\begin{array}{rcl}
 8x_1 + 6x_2 + 2x_3 - 6y_2 - 10 & \leq & 0 \\
 10x_1 + 6x_2 - 6y_2 - 10 & \leq & 0 \\
 3x_1 - 2x_2 - 6y_2 + 2 & \leq & 0 \\
 x_1 + x_2 - y_2 - 1 & \leq & 0 \\
 5x_1 - 6y_2 & \leq & 0 \\
 x_1 & \leq & 1 \\
 x_1 & \geq & 0 \\
 x_3 & \leq & 1 \\
 x_3 & \geq & 0 \\
 5x_1 + 6x_2 + 2x_3 & \leq & 10 \\
 y_2 & \leq & x_1 \\
 y_2 & \geq & 0 \\
 y_2 & \leq & x_2
 \end{array}$$

leads to the polyhedron of the first projection  $P_1(K)$  that is described by

$$5x_1 + 6x_2 + 2x_3 - 10 \leq 0$$

$$x_i \geq 0 \quad \text{for } i \in \{1, 2, \dots, n\}$$

$$x_i \leq 1 \quad \text{for } i \in \{1, 2, \dots, n\}$$

$$-3x_1 + 2x_2 - 2 \leq 0$$

$$5x_1 + 8x_2 + 2x_3 - 12 \leq 0$$

$$3x_1 + 6x_2 + 2x_3 - 10 \leq 0$$

$$2x_1 + 6x_2 + 2x_3 - 10 \leq 0$$

$$8x_1 + 2x_3 - 10 \leq 0$$

$$7x_1 + 8x_2 - 12 \leq 0$$

$$5x_1 + 6x_2 - 10 \leq 0$$

$$4x_1 + 6x_2 - 10 \leq 0$$

$$10x_1 - 10 \leq 0$$

$$3x_1 + 8x_2 - 8 \leq 0$$

$$1x_1 + 6x_2 - 6 \leq 0.$$

#### 4.2 Lovász-Schrijver Construction

The construction of the LS method goes as follows:

Step 1: For every index  $j \in \{1, 2, \dots, n\}$  achieve the nonlinear system

$$(Ax - b) * x_j \leq 0$$

$$(Ax - b) * (1 - x_j) \leq 0$$

by multiplying the original system  $Ax \leq b$  by  $x_j$  and respectively  $1 - x_j$ .

Step 2: Substitute  $y_{ij}$  for  $x_i x_j$  ( $i=1, \dots, n, i \neq j$ ) and  $x_j$  for  $x_j^2$  to receive a linearized system of constraints. Use the fact that  $x_i x_j = x_j x_i$  to have  $y_{ji} = y_{ij}$ .

Step 3: Project the resulting polytope onto  $\mathbb{R}^n$  by eliminating  $y$  in the linearized system.

The projection  $N(K)$  satisfies  $P \subseteq N(K) \subseteq K$ .

Step 4: Define  $N^1(K) := N(K)$

$$N^t(K) := N(N^{t-1}(K)) \text{ for } t \geq 2$$

and iterate.

We know that  $N(K) \subseteq \text{conv}(K \cap \{x \mid x_j \in \{0, 1\}\})$ ,  $j = 1, \dots, n$ .

Again, after at most  $n$  iterations we receive the convex hull  $N^n(K) = P$

Instead of having a linear relaxation  $N^t(K)$  we can add the restriction that the matrix  $Y = (y_{ij})$  has to be positive semidefinite in order to receive a semidefinite relaxation.

### 4.3 Sherali-Adams Construction

The construction of the SA method goes as follows:

Step 1: Let  $t \in \{1, 2, \dots, n\}$ .

Step 2: Multiplying the original system  $Ax \leq b$  by products of the form  $(\prod_{j \in J_1} x_j) * (\prod_{j \in J_2} (1 - x_j))$  where  $J_1$  and  $J_2$  disjoint subsets of  $\{1, 2, \dots, n\}$  such that  $|J_1 \cup J_2| = t$ .

Step 3: Linearize the system by substituting  $x_j$  for  $x_j^2$  and  $y_J$  for  $\prod_{j \in J} x_j$  ( $J \subseteq \{1, 2, \dots, n\}$ )

Step 4: Project the resulting polytope onto  $\mathbb{R}^n$  by eliminating  $y$  in the linearized system and denote the projection  $S_t(K)$ .

The inclusions  $K \supseteq S_1(K) \supseteq S_2(K) \supseteq \dots \supseteq S_n(K) = P$

hold and after at most  $n$  iterations the convex hull is achieved.

To sum it up, the relations among the 3 lift-and-project methods (for the linear relaxations) are as follows:

$$S_t(K) \subseteq N^t(K) \subseteq P_{j_1, j_2, \dots, j_t}(K)$$

and we reach the convex hull  $P$  in at most  $n$  iterations. This bound is tight as we will demonstrate by 2 examples in the next section.

## 5. WORST CASE EXAMPLES

For the following two examples, the SA lift-and-project method has the worst (biggest) number of iterations, which is  $n$ .

*Example 1:*

$$K := \left\{ x \in [0, 1]^n \mid \sum_{i=1}^n x_i \leq \frac{1}{2} \right\}$$

*Example 2:*

$$K := \left\{ x \in [0, 1]^n \mid \sum_{i \in I} x_i + \sum_{i \notin I} (1 - x_i) \geq \frac{1}{2} \right\}$$

## **6. REFERENCES**

Monique Laurent, Franz Rendl: **Semidefinite Programming and Integer Programming**, April, 2003

Egon Balas, Sebastián Ceria, Gérard Cornuéjols: **A lift-and-project cutting plane algorithm for mixed 0-1 programs**, Journal Mathematical Programming Volume 58, Numbers 1-3, p. 295-324, Springer Berlin / Heidelberg, January, 1993, ISSN: 0025-5610 (Print) 1436-4646 (Online)

Egon Balas, Sebastián Ceria, Gérard Cornuéjols: **Solving mixed 0-1 programs by a lift-and-project method**, Symposium on Discrete Algorithms, Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms, p.232 – 242, Austin, Texas, USA, 1993, ISBN: 0-89871-313-7