CONSTRAINT CONVEXITY TOMOGRAPHY AND LAGRANGIAN APPROXIMATIONS

Levon Aslanyan, Artyom Hovsepyan, Hasmik Sahakyan

***Abstract**: This paper considers one particular problem of general type of discrete tomography problems and introduces an approximate algorithm for its solution based on Lagrangian relaxation. A software implementation is given as well.*

***Keywords:** discrete tomography, lagrangian relaxation.*

***ACM Classification Keywords**: F.2.2 Nonnumerical Algorithms and Problems: Computations on discrete structures.*

Introduction

Discrete tomography is a field which deals with problems of reconstructing objects from its projections. Usually in discrete tomography object T, represents a set of points in multidimensional lattice. Some measurements are performed on T, each of which contains projection, which calculates number of points of T along parallel directions. Given finite number of such measurements it is required to reconstruct object T, or if it is not possible to find unique reconstruction, construct an object which satisfies given projections. The object existence problem even by given 3 non-parallel projections is NP-complete [1].

In recent years discrete tomography draws huge attention because of the variety of mathematical formulations and applications. Theory of discrete tomography is widely used particularly in the field of medical image processing, which is based on so called computerized tomography.

Lets consider 2-dimensional lattice and horizontal and vertical projections only. Object T can be represented as a $m \times n$ $(0,1)$ matrix, where 1s corresponds to points in T. Vector of row sums corresponds to horizontal projection and vector of column sums to vertical projection. So the problem of reconstructing the object by given horizontal and vertical projections is equivalent to the $(0,1)$-matrix existence problem with given R and S row and column sums. The latter problem was solved independently by Gale and Ryser in 1957. They gave sufficient and necessary condition for such a matrix existence and also proposed an algorithm for the matrix construction. Same problem with condition of rows inequality was investigated in [6].

In many cases orthogonal projections does not contain enough information for the objects unique reconstruction.

That's why often we consider different classes of such problems, where we impose additional constraints, for instance of geometrical nature. Such constraints narrow the solutions set but at the same time could make the problem hard to solve. Typical examples of such constraints are convexity and connectivity.

We say that matrix has row (or horizontal) convexity feature if all ones in the row forms a continuous interval. Same way we define column (or vertical) convexity. Connectivity is the feature of moving between 1s in neighboring cells. In our case we consider only vertical and horizontal connectivity (not diagonal).

Existence problem for connected matrices is NP-complete [2]. Existence problems for horizontally or vertically convex, and for both horizontally and vertically convex matrices are also NP-complete [3].

Different authors proved that horizontally and vertically convex and connected matrices reconstruction problem can be solved in polynomial time. Given description shows how sensitive are this kind of problems to input conditions. We see that existence problem's complexity changes along with adding new constraints. At the same time there are a lot of other notations of the problem for those the complexity is not even known. Particularly that means that they also lacks easy solution algorithms.

So we consider several problems in the field of discrete tomography, propose ways for constructing such matrices that satisfy constraints (convex or nearly convex, satisfying given parameters or having values near to given parameters). Further we will formulate the problems as optimization problems and give ways for their approximation, based on the integer programming relaxation. The question is that integer programming model is known for being used to reformulate known NP complex optimization problems. This model's (precise or approximate algorithms construction) investigation is very important and often this model is used to approximate optimizations problems [4, 6]. Implemented algorithms and software package based on that algorithms give an ability to make calculations either for tomography problem or for similar problems, such that those calculations might guide us or give approximate or precise solutions.

In this paper we will consider one problem from the field of discrete tomography, horizontally convex matrix existence problem.

Horizontally convex matrix existence problem

Since 1's in the horizontally convex matrix are in neighboring position then if we count the number of 1's in the matrices rows, that number for convex matrices will be maximum for the ones with same parameters. That's why problems that are often considered are related to number of neighboring 1's, their constraints and optimization.

$R=(r_1,\cdots,r_m)$, $S=(s_1,\cdots,s_n)$, $R^{'}=(r_1^{'},\cdots,r_m^{'})$ vectors are given. Is there a $m\times n$ $X=\{x_{i,j}\}$ matrix such that R is row sum vector for that matrix and S is column sums vector, and number of neighboring 1's in row i is equal to $r_i^{'}$.

$$\begin{cases} \sum_{i=1}^{m} x_{i,j} = s_j, j = 1,\cdots,n \\ \sum_{j=1}^{n} x_{i,j} = r_i, i = 1,\cdots,m \\ \sum_{j=1}^{n-1} \min(x_{i,j}, x_{i,j+1}) = r_i^{'}, i = 1,\cdots,m \\ x_{i,j} \square \{0,1\} \end{cases}$$

In other words the problem is following, find the matrix with horizontal convexity in the class of $(0,1)$ matrices with given row and column sums. This problem is NP-complete, since for the case when $r_i^{'} = r_i - 1, i = 1,\cdots,m$ it's equivalent to the horizontally convex matrix existence problem. Given particular case just require the matrix to be horizontally convex by neighboring 1s in the rows.

As we already mentioned lot of combinatorial problems are suitable to represent as integer linear optimization problems. Lets reformulate our problem as integer programming problem.

Lets define $y_{i,j} \in \{0,1\}$ variables the way that it provides neighboring 1's in row i.

$$(y_{i,j} = 1) \Leftrightarrow (x_{i,j} = 1) \,\&\, (x_{i,j+1} = 1), i = 1,\cdots,m;\ j = 1,\cdots,n-1$$

This can be done by satisfying conditions

$$\begin{cases} y_{i,j} \le x_{i,j} \\ y_{i,j} \le x_{i,j+1} \\ y_{i,j} \ge x_{i,j} + x_{i,j+1} - 1 \end{cases}$$

So we reformulate the problem in the following way.

$R = (r_1,\cdots,r_m)$, $S = (s_1,\cdots,s_n)$, $R^{'} = (r_1^{'},\cdots,r_m^{'})$ vectors are given: Is there a $m \times n$ $X = \{x_{i,j}\}$ matrix such that

$$
\begin{cases}
(1)\sum_{i=1}^{m} x_{i,j} = s_j, j = 1,\cdots,n \\
(2)\sum_{j=1}^{n} x_{i,j} = r_i, i = 1,\cdots,m \\
(3)\begin{cases} y_{i,j} \le x_{i,j} \\ y_{i,j} \le x_{i,j+1} \\ y_{i,j} \ge x_{i,j} + x_{i,j+1} - 1 \end{cases} \quad i = 1,\cdots,m, j = 1,\cdots,n-1 \\
(4)\sum_{j=1}^{n-1} y_{i,j} = r_i^{'}, i = 1,\cdots,m \\
(5) x_{i,j} \square \{0,1\}, y_{i,j} \in \{0,1\}
\end{cases}
$$

Lagrangean relaxation and variable splitting

So we have horizontal row convex matrix existence problem, which is reformulated as linear integer programming problem I. We also know that problem I is NP-complete. To solve this problem we will use a method based on Lagrangian relaxation.

Obviously if we drop some of the constraints we will get problems relaxation. Assume that we can call one or several constraints hard in the since that by dropping those constraints we can solve resulted integer programming problem more easily. Constraints dropping could be embedded in more common method which is called Lagrangian relaxation. We can apply Lagrangian relaxation to given method in various ways. One of the ways, which we will use here is following, if the problem can be splitted to subproblems, which have common variables, first split those variables and then relax their equality constraint.

So, we take two set of variables $x_{i,j}^{h}$ and $x_{i,j}^{v}$ by duplicating $x_{i,j}$ variables, and reformulate our problem as

$$\begin{cases} (1)\sum_{i=1}^{m} x_{i,j}^{v} = s_j, j = 1,\cdots,n \\ (2)\sum_{j=1}^{n} x_{i,j}^{h} = r_i, i = 1,\cdots,m \\ (3)\begin{cases} y_{i,j} \le x_{i,j}^{h} \\ y_{i,j} \le x_{i,j+1}^{h} \\ y_{i,j} \ge x_{i,j}^{h} + x_{i,j+1}^{h} - 1 \end{cases} \quad i = 1,\cdots,m, j = 1,\cdots,n-1 \\ (4)\sum_{j=1}^{n-1} y_{i,j} = r_i^{'}, i = 1,\cdots,m \\ (5) x_{i,j}^{h}, x_{i,j}^{v} \square \{0,1\}, y_{i,j} \in \{0,1\} \\ (6) x_{i,j}^{h} = x_{i,j}^{v} \end{cases}$$

We split our original problem using variable splitting to two problems, each of which has its own variable set and which would be independent without constraint (6). From this point of view constraint (6) is the hardest one. We will relax constraint (6) using Lagrangian relaxation with coefficients $\lambda_{i,j}$.

We get following problem $VSI(\lambda)$, and its optimal value is $v^{VSI}(\lambda)$.

$$\begin{cases} \max(\sum_{i,j} \lambda_{i,j}(x_{i,j}^{h} - x_{i,j}^{v})) \\ (1)\sum_{i=1}^{m} x_{i,j}^{v} = s_j, j = 1,\cdots,n \\ (2)\sum_{j=1}^{n} x_{i,j}^{h} = r_i, i = 1,\cdots,m \\ (3)\begin{cases} y_{i,j} \le x_{i,j}^{h} \\ y_{i,j} \le x_{i,j+1}^{h} \\ y_{i,j} \ge x_{i,j}^{h} + x_{i,j+1}^{h} - 1 \end{cases} \quad i = 1,\cdots,m, j = 1,\cdots,n-1 \\ (4)\sum_{j=1}^{n-1} y_{i,j} = r_i^{'}, i = 1,\cdots,m \\ (5) x_{i,j}^{h}, x_{i,j}^{v} \square \{0,1\}, y_{i,j} \in \{0,1\} \end{cases}$$

Then using same method we can further split the problems into subproblems for rows and columns, which itself is reducing to the finding of simple path, with given number of edges and biggest weight on directed graph.

We can approach the problem in other way, by relaxing constraint (3) we would split the problem into two subproblems with $x_{i,j}$ and $y_{i,j}$ variables. But this paper is limited with first approach.

Obviously problem $VSI(\lambda)$ is relaxation of problem I, hence $v^{VSI}(\lambda)$ is upper limit for value of I. Find best upper limit means to solve Lagrangian dual problem which is

$$v^{VSD} = \min_{\lambda} v^{VSI}(\lambda)$$

This is convex non-differential optimization problem: There are different methods for solving this problem. One of them is subgradient optimization method. Subgradient optimization on each step calculates the value of $v^{VSI}(\lambda)$ for given $\lambda_{i,j}$, in this case that equals to solving following m independent problems

$$\begin{cases} \max(\sum_{j=1}^{n} c_j x_j) \\ \sum_{j=1}^{n} x_j = r \\ \begin{cases} y_j \le x_j \\ y_j \le x_{j+1} \qquad j = 1, \cdots, n-1 \\ y_j \ge x_j + x_{j+1} - 1 \end{cases} \\ \sum_{j=1}^{n-1} y_j = r' \\ x_j \in \{0,1\}, y_j \in \{0,1\} \end{cases} \qquad (*)$$

We will try to solve these problems using algorithm for finding simple path on acyclic directed graph with biggest cost and given number of edges.

Decomposed problem on graph and the solution

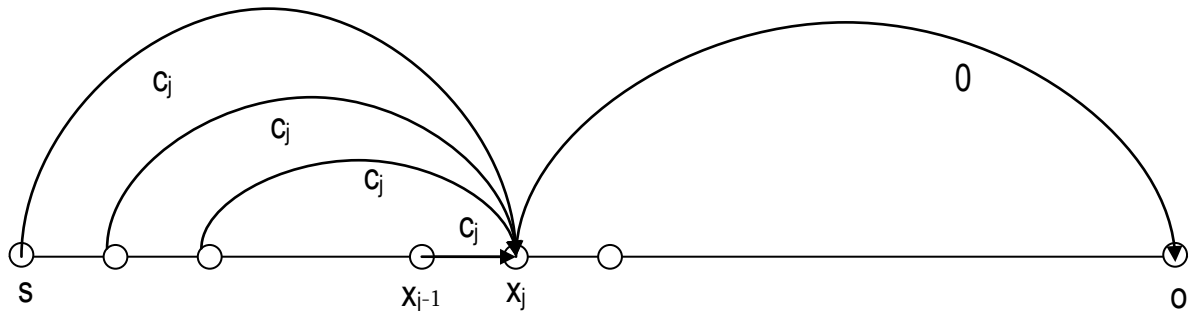
We consider directed graph $G = (V, E)$ which vertex set consists of vertexes for each x_j variable plus S source and O destination. We define edges in following way

(s, x_j) with weight c_j

$(x_i, x_j), i \leq j-1$ with weight c_j

(x_j, o) with weight 0

Consider the paths form S to O. Only r variables corresponding to x_j vertexes, are 1's according to (*) and among them r' is neighboring 1's. Hence we are interested only in those paths from S to O that have only r vertexes and there are only r' with neighboring 1's. We need to find among those paths, the one that has maximum weight. Now by assigning 1's to variables corresponding to vertexes we will get solution to the problem (*).



Now lets give algorithmic description.

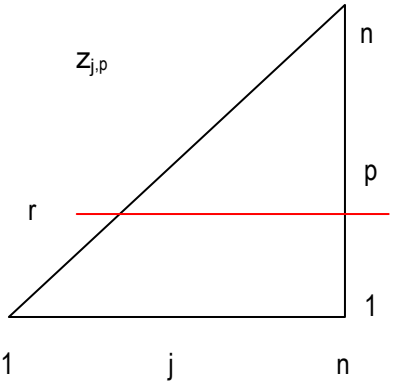
Let $z(j, p)$ is weight of the longest path from S to x_j vertex with p vertexes on it. Lets $w(j, p, q)$ is weight of the longest path from S to x_j which has p vertexes on it and there are q neighboring vertexes with corresponding variables equal to 1. In this case $z(j, p)$ and $w(j, p, q)$ can be calculated the following way. First of all consider $z(j, p)$

$$z(j,1) = c_j$$
$$z(j,p) = \max_{u<x_j}(z(u,p-1) + c_j)$$

And the optimal value we're looking for is $z(o,r) = \max_j z(x_j,r)$.

This part of the problem is solving in the following way.

For given n and r $Z_{j,p} = [z_{j,p}]$ array is constructed, where $j = 1,...,n$ and for j $p = 0,...,j-1$. In reality for fixed r its enough to consider $p = 1,...,r$ layers, but $p = 1,...,n$ will satisfy calculations needed for all r.



First of all $z_{1,1}$ value is calculated. That's equal to c_1. All values of row $p = 1$ are calculated in the same way $z_{j,1} = c_j$. To calculate $z_{j,p}$ by our formula we need to know values for $p-1$ and for all $1,...,j-1$ indexes. But in row $p-1$ first non-zero value is in $j = p-1$ position, which is on diagonal. So calculations can be done sequentially on $p = 1,...,r,...$ rows and in rows in order $j = p,...,n$. This constructs are needed for software implementation and these give ability to measure number of operations in calculation. It doesn't exceed n^3, which means polynomial complexity.

Maximal weight paths can be stored in a separate array. They can be stored as 0,1 vectors or as indexes of non zero elements which however won't significantly decrease number of computations.

Now lets calculate values of $w(j,p,q)$. First of all lets consider edge values. From $w(j,p,q)$ we have maximal weight path from s to x_j which has p vertexes and there are q pairs with neighboring 1's. $q \le p-1$ and lets p's are decreased up to $q+1$. $w(j,q+1,q)$'s can be non-zero starting from $j \ge q+1$. For bigger q's and smaller j's $w(j,p,q)$'s are equal to 0.

Interestingly q can't be very small. If $p > \left]\frac{j+1}{2}\right[$ then q can't be 0 (at least 2 vertexes must have neighboring indexes).

Let $\tau = \left[\frac{j+1}{3}\right]$. In that case τ vertex pairs still might not be neighbors, which gives 2τ vertexes. After that any new vertex addition would add 2 new pairs.

Now lets consider common case. For calculating $w(j,p,q)$ lets consider class where for j $p \le j$ and for j,p pairs $q \le p-1$. This class is larger than needed but in reality it doesn't differ much from the minimal class which is necessary for calculations. For slight transition of edge values class is zeroed before performing calculations. Lets investigate value of $w(j,p,q)$. We do chain calculations and on each step consider 2 cases $x_{j-1}=1$ and $x_{j-1}=0$. So we get following values

$w(j-1,p-1,q-1)+c_j$ and $\max_{u<x_j-1}(w(u,p-1,q)+c_j)$

We are interested in maximum of these values.

In $w(j-1,p-1,q-1)+c_j$ all indexes are less than preceding and we assume that this value is already calculated in previous steps. For calculating $\max_{u<x_j-1}(w(u,p-1,q)+c_j)$ we do next step in chain calculation.

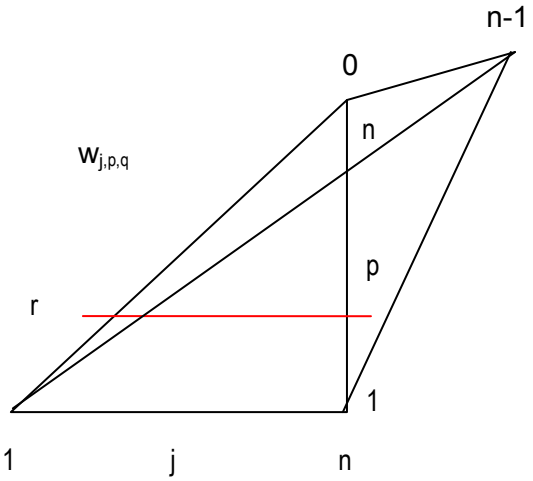
$$w(u-1, p-2, q-1) + c_j$$

$$\max_{v<u-1}(w(v, p-2, q) + c_j)$$

And the needed optimal value is $w(o, r, r') = \max_j w(x_j, r, r')$. This problem practically can be solved in following way.

For given n, r, r' we construct the class given above, array $W_{j,p,q} = [w_{j,p,q}]$, where for $j = 1,\ldots,n$ and j $p = 1,\ldots,j$ and for pair j, p $q = 0,\ldots,p-1$.

In reality for fixed r it's enough to consider $p = 1,\ldots,r$ layers and for q all values where $q \le r-1$. But calculations must be done in such sequence to be executable.



First $w_{1,1,0}$ values are calculated, $w_{1,1,0} = c_1$, all values in row $p = 1$ are calculated in the same way $w_{j,1,0} = c_j$. More, $q = 0$ values were already considered. To calculate $w_{j,p,q}$ based on our formula we need to know values for $p-1$ and all $1,\ldots,j-1$. But in layer $p-1$ with current q value is either 0 or already calculated. Then calculations can be done in layers $p = 1,\ldots,r,\ldots$ sequentially and in layers in order

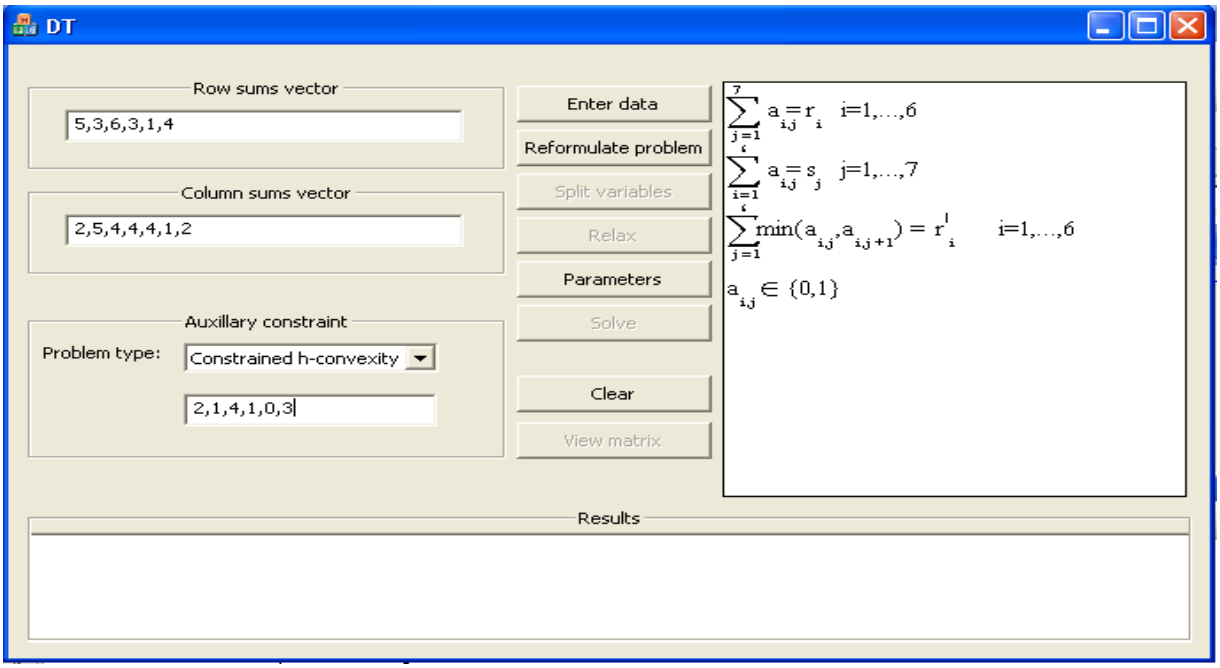
of $j = p,...,n$. Given constructions are needed for software implementation and give ability to measure number of calculations. Those are not more than n^4 which means polynomial complexity.

Maximal weight paths that we're looking for could be stored in separate array as 0,1 vectors or as array of indexes with non-zero values, which however won't significantly lower number of calculations.

Software implementation

Based on given methods a software system with an UI was implemented, which can be used to solve some problems from the field of discrete tomography based on Lagrangian relaxation.

There are several fields which are used for data input. Since we are solving problems in the field of discrete tomography so input data are projections, in our case row sums and column sums. Also we are giving specific problem description by additional constraints. So we have special fields for that purpose. Then there is special control which can be used to reformulate given problem as mathematical programming problem. Then we can choose one or several constraints which we want to relax. Also we can do variable splitting etc. And there is an output window which is used for displaying results. For example value of Lagrangian Dual or variables difference as a result of splitting.



In given example as a problem is considered horizontal row convexity existence problem.

Now lets describe one of the main classes in the implementation, ProblemBase abstract class. This class is base for all problem. Class encapsulates problem data. It also has several virtual functions which are used for problem solution. For example function which reformulates the problem as mathematical programming problem, chooses constraints for relaxation. Important function in ProblemBase is Solve method, which invokes the method for specific problem. Since most of the problems are reducing to relatively easy problems on graphs and methods for those solutions can be used for different problems we put those methods in a separate library.

Bibliography

1. Gardner R.J., Gritzmann P., Prangenberg D., On the computational complexity of reconstructing lattice sets from their X-rays. Technical Report (970-05012), Techn. Univ. Munchen, fak. f. math, 1997.

2. G.J. Woeginger. The reconstruction of polyominoes from their orthogonal projections. Inform. Process. Lett., 77:225-229, 2001.

3. E. Barcucci, A. Del Lungo, M. Nivat, and R. Pinzani. Reconstructing convex polyominoes from horizontal and vertical projections. Theoret. Comput. Sci., 155:321-347, 1996.

4. G. Dahl and T. Flatberg. Lagrangian decomposition for reconstructing hv-convex (0, 1) matrices, Report 303, University of Oslo, p. 1-13, 2002.

5. M. Guignard and S. Kim. Lagrangian decomposition: a model yielding stronger lagrangian bounds. Math. Prog., 39:215-228, 1987.

6. H.J. Ryser. Combinatorial properties of matrices of zeros and ones. Canad. J. Math., 9:371-377, 1957.

7. D. Gale. A theorem on flows in networks. Paci_c J. Math., 7:1073-1082, 1957.

Authors' Information

Levon Aslanyan *– Head of Department, Institute for Informatics and Automation Problems, P.Sevak St. 1, Yerevan 14, Armenia, e-mail: lasl@.sci.am*

Artyom Hovsepyan *– Researcher, Institute for Informatics and Automation Problems, P.Sevak St. 1, Yerevan 14, Armenia, e-mail:* artyom.hovsepyan@gmail.com

Hasmik Sahakyan *– Leading Researcher, Institute for Informatics and Automation Problems, NAS RA, P.Sevak St. 1, Yerevan 14, Armenia, e-mail: hasmik@ipia.sci.am*