Nonserial dynamic programming and local decomposition algorithms in discrete programming

Arnold Neumaier, Oleg Shcherbina

Fakultät für Mathematik
Universität Wien
Nordbergstrasse 15
A-1090 Wien
Austria
email: { Arnold.Neumaier, Oleg.Shcherbina }@univie.ac.at

Abstract.

One of perspective ways to exploit sparsity in the dependency graph of an optimization problem is nonserial dynamic programming (NSDP) which allows to compute solution in stages, each of them uses results from previous stages. The class of discrete optimization problems with the block-tree matrix of constraints is considered. Nonserial dynamic programming (NSDP) algorithm for the solution of this class of problems is proposed. Properties of NSDP algorithm for the solution of the problems of this class are investigated.

1 Introduction

Many questions in computer science, mathematics, and various applications can be expressed as or are strongly related to optimization problems ([8], [13], [16]). Among the most prominent classes of optimization problems we find the discrete programming problems (denoted by DP). Applications of combinatorial optimization are found in such areas as VLSI routing, scheduling, network optimization, routing traffic in communications networks, economical allocation, multi-dimensional smoothing for pattern recognition, theorem proving, game theory, and artificial intelligence. Popular benchmark applications from the literature include the integer knapsack problem, the traveling salesman problem, the 15 puzzle, and the n-queens problem. The tremendous attention that combinatorial optimization and DP particularly have received in the literature gives some indication of its importance in many research areas. Unfortunately, most of the interesting problems are in the complexity class NP-hard and may require searching a tree of exponential size (if $P \neq NP$) in the worst case. Today almost everyone expects that there is no polynomial time algorithm for solving an NP-hard problem ([9], p.1). Many real DP problems contain a huge number of variables and/or constraints that make the models intractable for currently available DP solvers. A well known drawback of exact algorithms of DP is that they have worst case exponential time performance. As noted in [9], one of the modern approaches is the development of practical exact algorithms for solving particular NP-hard problems.

Complexity theory proved that universality and effectiveness are contradictory requirements to algorithm complexity. But the complexity of some class of problems decreases if this

class may be divided into subsets and the special structure of these subsets can be used in algorithm construction. In recent years there has been emerging interest in the design and analysis of algorithms that are significantly faster than exhaustive search, though still not polynomial time (so called super-polynomial time algorithms that solve NP-complete problems to optimality), see the recent survey by WOEGINGER [25]. One of the promising approaches to cope with NP-hardness in solving DP problems is the construction of decomposition methods. The development of the decomposition methods in DP ([18], [24], [23], [17]) is very urgent due to the above-stated facts. Decomposition techniques usually determine subproblems which solutions can be combined to create a solution of the initial DP problem. Usually, DP problems from applications have a special structure, and the matrices of constraints for large-scale problems have a lot of zero elements (sparse matrices). The nonzero elements of the matrix often fall into a limited number of blocks. The block form of many DP problems is usually caused by the weak connectedness of subsystems of real-world systems. One of the promising ways to exploit sparsity in the dependency graph of an optimization problem is nonserial dynamic programming (NSDP) (BERTELE, BRIOSCHI [3], HOOKER [12]), which allows to compute a solution in stages such that each of them uses results from previous stages.
The search for graph structures appropriate for the application of dynamic programming caused a series of papers dedicated to tree decomposition research ([19], [1], [2], [14], [6], [4], [5], [10], [11]). Tree decomposition and the related notion of a treewidth (ROBERTSON, SEYMOUR [19]) play a very important role in algorithms, for many NP-complete problems on graphs that are otherwise intractable become polynomial time solvable when these graphs have a tree decomposition with restricted maximal size of cliques (or have a bounded treewidth [4], [5]).

But only few papers on applications of this powerful tool in the area of DP exist [15].

In this paper, the class of discrete optimization problems with a tree-like structure is considered and based on NSDP, algorithms are proposed for solving this class of problems.

2 Nonserial dynamic programming

A nonserial dynamic programming (NSDP) algorithm [3] consists of the elimination of variables according to a given ordering in such a way that after a variable v has been eliminated all the variables connected to it are joined together (making the neighborhood of v into a clique in the remaining graph, before deleting v), modifying the structure of the interaction graph.

2.1 Example. For example, consider the ILP problem:

$$
\begin{aligned}
& 2x_1 + 3x_2 + x_3 + 5x_4 + 4x_5 + 6x_6 + x_7 && \to \max \\
& 3x_1 + 4x_2 + x_3 && \le 6, \\
& \quad 2x_2 + 3x_3 + 3x_4 && \le 5, \\
& \quad 2x_2 \qquad + 3x_5 && \le 4, \\
& \qquad 2x_3 \qquad + 3x_6 + 2x_7 && \le 5, \\
& x_j = 0, 1, \ j = 1, \ldots, 7.
\end{aligned}
$$

An interaction graph $G = (V, E)$ (or augmented constraint graph [6]) of this optimization problem contains a vertex for each variable and an edge connecting any two variables that appear either in the same constraint or in the same component of the objective function. The interaction graph is shown in Fig 1a).

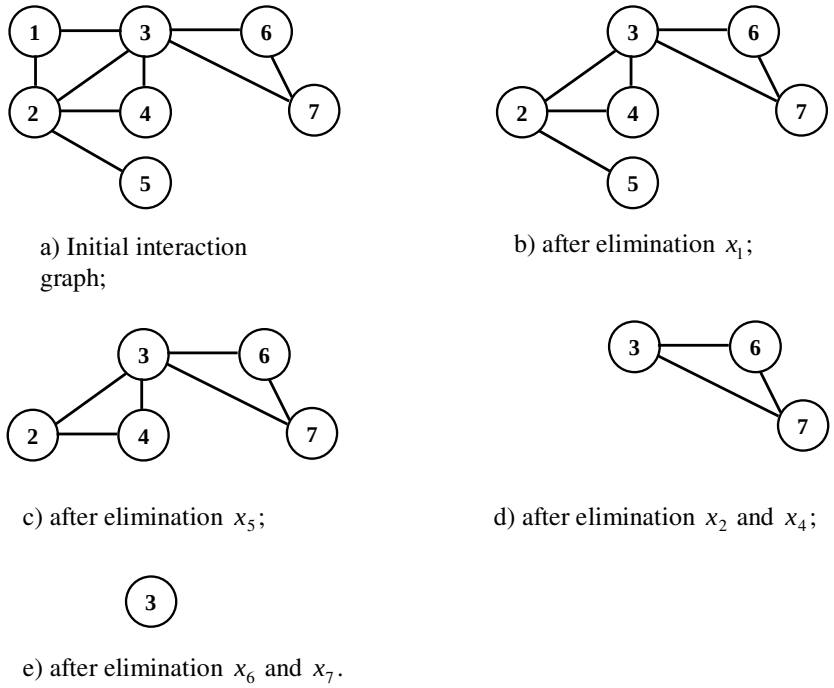

Figure 1: Interaction graph

To eliminate variable x_1 first, we consider with which variables x_1 interacts: x_2 and x_3. For every assignment to x_2 and x_3, compute the value of x_1 for which:
$h_1(x_2, x_3) = \max_{x_1}\{2x_1 \mid 3x_1+4x_2+x_3 \leq 6, x_j \in \{0, 1\}\}$ (note that x_1 is in the first constraint only); see table 1.

Table 1: Calculation of $h_1(x_2, x_3)$

x_2	x_3	h_1	x_1^*
0	0	2	1
0	1	2	1
1	0	0	0
1	1	0	0

Bellman's Principle of Optimality holds because, once the optimal values for x_2 and x_3 have been determined, the optimal value for x_1 is x_1^*. Therefore, we can consider a new problem,

in which x_1 has been eliminated:

$$\begin{aligned} h_1(x_2,x_3) + 3x_2 + \ x_3 + 5x_4 + 4x_5 + 6x_6 + \ x_7 \quad &\to \max \\ 2x_2 + 3x_3 + 3x_4 \quad &\le 5, \\ 2x_2 \qquad\quad + 3x_5 \quad &\le 4, \\ 2x_3 \qquad\quad + 3x_6 + 2x_7 \quad &\le 5, \\ x_j = 0,1,\ j = 2,\dots,7. \end{aligned}$$

The interaction graph for the new problem is shown in Fig 1 b).
Eliminate variable x_5. The neighbor of x_5 is x_2: $Nb(x_5) = \{x_2\}$. Solve the following problem containing x_5 in the objective and the constraints:
$h_2(x_2) = \max_{x_5}\{4x_5 \mid 2x_2 + 3x_5 \le 4, x_j \in \{0,1\}\}$
and build the table:

Table 2: Calculation of $h_2(x_2)$

x_2	h_2	x_5^*
0	4	1
1	0	0

The interaction graph after elimination of x_5 is shown in Fig 1c).

This leaves a new problem, in which x_5 does not appear (the constraint containing x_5 also has been eliminated):

$$\begin{aligned} h_1(x_2,x_3) + h_2(x_2) + \ x_3 + 5x_4 + 4x_5 + 6x_6 + \ x_7 \quad &\to \max \\ 2x_2 + 3x_3 + 3x_4 \quad &\le 5, \\ 2x_3 \qquad\quad + 3x_6 + 2x_7 \quad &\le 5, \\ x_j = 0,1,\ j = 2,3,4,6,7. \end{aligned}$$

Note that x_2 and x_4 interact only with x_3 (in the second constraint and the objective), so we can eliminate them together (in block) by solving $h_3(x_3) = \max_{x_2,x_4}\{h_1(x_2,x_3) + h_2(x_2) + 3x_2 + 5x_4 \mid 2x_2 + 3x_3 + 3x_4 \le 5, x_j \in \{0,1\}\}$ and build the following table

Table 3: Calculation of $h_3(x_3)$

x_3	h_3	x_2^*	x_4^*
0	11	0	1
1	6	0	0

The new problem left to be solved is

$$\begin{aligned} h_3(x_3) + \ x_3 \qquad & + 6x_6 + \ x_7 & \to \max \\ 2x_3 \qquad & + 3x_6 + 2x_7 & \le 5, \\ x_j = 0, 1, \ j = 3, 6, 7. & & \end{aligned}$$

The corresponding interaction graph is in Fig 1d).

Eliminate x_6 and x_7 in block. Now the problem to be solved is $h_4(x_3) = \max_{x_6,x_7}\{6x_6 + x_7 \mid 2x_3 + 3x_6 + 2x_7 \le 5, x_j \in \{0, 1\}\}$. Build the corresponding table:

Table 4: Calculation of $h_4(x_3)$

x_3	h_4	x_6^*	x_7^*
0	7	1	1
1	6	1	0

The corresponding interaction graph is in Fig. 1e). The new problem is: $\max_{x_3}\{h_3 + x_3 + h_4 | x_j \in \{0, 1\}\} = 18$, and the solution is x_3^*=0 and the maximal objective value is 18.

The order of elimination is $\alpha = \{x_1, x_5, (x_2, x_4), (x_6, x_7), x_3\}$. To find the optimal values of the variables, it is necessary to do backward step of the dynamic programming procedure: from the last table we have x_3^*=0. Considering the tables before we have for x_3^*=0: $x_6^* = 1, x_7^* = 1$, $x_2^* = 0, x_4^* = 1, x_5^* = 1, x_1^* = 1$. The solution is (1, 0, 0, 1, 1, 1, 1).

It is expedient to apply the NSDP procedure for elimination not only to separate variables but to sets of variables. Considering the quasiblock problem of integer optimization it is easy to see that application of the NSDP procedure leads to a local algorithm (defined in a section below) using a path for elimination blocks.

3 Local decomposition algorithms for discrete programming

The possibility of applying NSDP for the solution specific problems of discrete programming is caused by the weak connectedness of the subsets of real life complex systems being simulated.

During the study of complex objects it is not always possible (and expedient) to obtain (or to calculate) complete information about the object as a whole; therefore it is of interest to obtain information about the object, to examine it in parts, i.e., locally.

Yu.I. Zhuravlev [26] introduced and investigated local algorithms for calculation of the information about properties of objects. The local algorithm is characterized by two parameters: the volume of the neighborhood studied in each step of the algorithm, and the number

of stored characteristics of each element of the studied set (the last parameter is called a memory of the local algorithm).

Local decomposition algorithms (LA) [27],[7], [22] for the solution of DP problems use a concept of a neighborhood of the variables of the DP problem and finding an optimal solution of the DP problem is achieved with the aid of the study of neighborhoods and the decomposition of the initial DP problem on this basis.

Note. A special case of block-tree structure is a quasiblock structure [7] , for which the graph of the intersections of neighborhoods (or blocks) is a path (Fig 2).

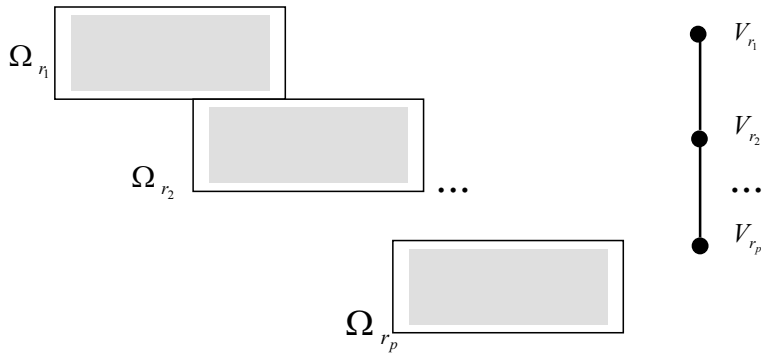

Figure 2: Quasi-block matrix and the corresponding path of blocks.

Consider the integer linear programming problem Z with binary variables:

$$z = c^T x \to \max \tag{1}$$

subject to the constraints

$$Ax \le b, \tag{2}$$

$$x_j = 0, 1,\ j = 1, \dots, n, \tag{3}$$

where $c = (c_j)_n,\ b = (b_i)_m,\ A = \|a_{ij}\|_{mn}$.

3.1 Definition. A set $S_1(j,\ Z) = \{x_k \mid a_{ik} \neq 0,\ i \in U_1\}$ is called the first order neighborhood of an index j; here $U_1 = \{i \mid a_{i1} \neq 0,\ i = 1, \dots, m\}$.

3.2 Definition. Row-neighborhood of an index set J:
$U(J) = \{i \mid a_{ik} \neq 0,\ k \in J\}$;
Column-neighborhood of a set I:
$S(I) = \{k \mid a_{lk} \neq 0,\ l \in I\}$, where $J \subseteq N = \{1, \dots, n\},\ I \subseteq M = \{1, \dots, m\}$.

The arbitrary order neighborhood of the index j is determined by induction. Let the pth order neighborhood $S_p(j,\ Z)$ be known. Let us determine the neighborhood of order $p+1$ as follows:

$$S_{p+1}(j,\, Z) = S(U(S_p(j,\, Z))).$$

Introduce the so-called external ring $Q(j, p, Z)$, i.e., the following set of indices:

$$Q(j,\, p,\, Z) = S(U(S_p(j,\, Z))) \setminus S_p(j,\, Z).$$

Each step of the LA consists in the change of neighborhoods and in replacing the index p with $p + 1$ (although it is possible to pass, also, from S_p to $S_{p+\rho}$); for each fixed collection of the variables of external ring the values of the variables of the corresponding neighborhood are stored. if $S = \{j_1,\, \ldots,\, j_q\}$, then we shall use the following notation: $X_S = \{x_{j_1},\, \ldots,\, x_{j_q}\}$;

Let $X_{Q^{(r)}}$ be a set of variables with indices in the external ring $Q^{(r)}$. Then the LA at the rth step is solving the optimization problem whose set of variables is determined by the neighborhood S_r, the set of constraints by the neighborhood U_r. The objective function is obtained from the objective function of the problem Z by the removal of all variables whose indices do not enter in S^r, and values $X_{Q^{(r)}}$ are fixed.

Let us note that upon transfer from S_r to S^{r+1} it is possible to use the information obtained during the solution of the ILP problem, which corresponds to the neighborhood S^r.

Let $Q^{(r-1)}$ be the external ring from the previous step of the LA, then passage from one neighborhood to the next one may be described using the relationship

$$f_r\left(X_{Q^{(r)}}\right) = \max_{Q^{(r-1)}} \left\{ f_{r-1}\left(X_{Q^{(r-1)}}\right) + z\left(X_{Q^{(r-1)}} \mid X_{Q^{(r)}}\right) + C_{Q^{(r-1)}} X_{Q^{(r-1)}} \right\},$$

where

- $z\left(X_{Q^{(r-1)}} \mid X_{Q^{(r)}}\right)$ is an optimum value of the objective function of the ILP problem obtained from the initial problem by fixing the variables in the external rings $Q^{(r-1)}$ and $Q^{(r)}$;
- $f_r\left(X_{Q^{(r)}}\right)$ is an optimum value of the objective function of the ILP problem which corresponds to neighborhood S_r with the fixed values of variables in the external ring $Q^{(r)}$.

The critical place of the LA is an enumeration over the external ring, since if the external rings of the neighborhoods are large, then the volume of the full enumeration over the rings will be exponentially large ("curse of dimensionality").

4 Local decomposition algorithm for solution of DP problems with a tree-like structure

Consider a LA for solution of DP problems with a tree-like structure, i.e., problems in which it is possible to find the set of the neighborhoods of different variables so that one variable

can belong to two neighborhoods only and the graph of intersections of these neighborhoods is a tree. It is clear that such a structure is a tree-decomposition and can be obtained with the aid of known tree decomposition algorithms ([6], [4], [5], [10], [11]). The LA solves this DP problem, moving bottom-up, i.e., from the neighborhoods corresponding to leaves of the tree, to the neighborhood corresponding to the root of the tree.

Let us introduce the necessary notions. Let $\Omega_1 = (S_1,\ U_1)\,,\quad \Omega_2 = (S_2,\ U_2)\,,\ldots,\quad \Omega_k = (S_k,\ U_k)$ be a set of the neighborhoods of some indices $j_1,\ldots,j_k$of some variables, where $S_r,\ U_r$ are, respectively, the sets of the indices of variables and constraints for the rth neighborhood, $r = 1,\ldots,k$, and

$$\bigcup_{r=1}^{k} U_r = M = \{1,\ldots,m\}\,, \tag{4}$$

$$\bigcup_{r=1}^{k} S_r = N = \{1,\ldots,n\}\,, \tag{5}$$

$$U_{r_1} \cap U_{r_2} = \emptyset,\ r_1 \neq r_2, \tag{6}$$

$$S_{r_1} \cap S_{r_2} \cap S_{r_3} = \emptyset \text{ for any triple of different indices } r_1, r_2, r_3. \tag{7}$$

Consider a vertex r of the tree D and introduce a tree D_r which consists of the vertex r and all its descendants.
Introduce the necessary notation:
S_r is the set of indices of variables which belong to block B_r;
$S_{rr'}$ is the set of indices of variables which belong simultaneously to blocks B_r and $B_{r'}$;
p_r is the vertex-ancestor for the vertex r;
J_r is the set of descendants of vertex r (see Fig. 3).

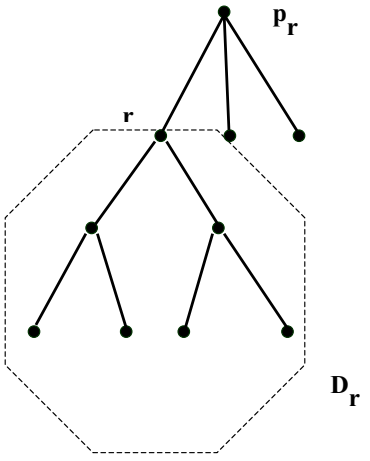

Figure 3: Tree D_r of descendants

In this notation $X_{S_{p_r r}}$ is the vector of the variables, common to the blocks B_{p_r} and B_r.

Let us designate as Z_{D_r} the following problem: for each vector $X_{S_{p_r r}}$ to find X_{S_r} and $X_{S_{rr'}}$ so that

$$f_{D_r}(X_{S_{p_r r}}) = \max\{C_{S_r} X_{S_r} + \sum_{r' \in J_r} [f_{D_{r'}}(X_{S_{rr'}}) + C_{S_{rr'}} X_{S_{rr'}}]\}$$

subject to the constraints

$$A_{S_r} X_{S_r} \leq b_r - \sum_{r' \in J_r} A_{S_{rr'}} X_{S_{rr'}} - A_{S_{p_r r}} X_{S_{p_r r}}.$$

Here $f_{D_r}(X_{S_{p_r r}})$ is an objective function value of the problem corresponding to the tree D_r.

The solution of the problem Z_{D_r} for the vertex r of the graph D with the fixed vector $X_{S_{rr'}}$ will be designated as:

$$X_{D_r}(X_{S_{p_r r}}) = \bigcup_{r' \in J_r} [X_{D_{r'}}(X_{S_{rr'}}) \bigcup X_{S_{rr'}}] \bigcup X_{S_r}.$$

It is clear that

$$f_{D_{r'}}(X_{S_{rr'}}) = C_{D_{r'}} X_{D_{r'}}.$$

It is easy to see that if we fix a vector $X_{S_{p_r r}}$, then the problem is decomposed into two problems: the first one corresponds to the tree D_r; and the second one to $D \setminus D_r$. An application of LA $\mathfrak{A}_{BT}$ for solving DP problems with a tree-like structure is based on this property.

5 Estimates of the efficiency of the local decomposition algorithm

An estimate of the efficiency of the LA when solving DP problems with a tree-like structure which is characterized by the tree D with weights of the vertices n_r and weights of the edges $n_{rr'}$, with binary variables and k blocks has the form

$$E(n, k, D) = \sum_r 2^{N_r} \varphi(n_r),$$

where $\varphi(n_r)$ is the estimate of the efficiency of the DP algorithm which solves DP problems corresponding to the blocks B_r; here

$$N_r = n_{p_r r} + \sum_{r' \in J_r} n_{rr'}.$$

When using a complete enumeration algorithm to solve the DP problems corresponding to the blocks, $\varphi(n_r) = 2^{n_r}$. Using this estimate it is possible to show that a transfer from a tree-like structure to a quasiblock one obtained by joining blocks is not expedient.

It is interesting to obtain the extremal and average values ($\langle E(n,k)\rangle = \frac{1}{\mid \{D\} \mid} \sum_{\{D\}} E(n,k,D)$) of the estimate of efficiency of the LA for the specified structure D and parameters n, k. Proofs of these propositions are in [22].

5.1 Proposition. *Writing* $d_{r^*} = \max d_r$, $d_{\bar{r}(r^*)} = \max_{r \in J_{r^*} \cup p_{r^*}} d_r$ *we have*

$$\max_{n_r, n_{rr'}} E(n,k,D) = 2^{n-2k+2}(2^{d_{r^*}} + 2^{d_{\bar{r}(r^*)}}) + \sum_{r \neq r^*, \bar{r}(r^*)} 2^{d_r+1},$$

5.2 Proposition. *If* $n > k \max d_r - k + 1$, *then*

$$\min E \geq k 2^{(n+k-1)/k}$$

.

5.3 Proposition. *If* $\varphi(n_r) = 2^{n_r}$, *then*

$$\langle E(n,k)\rangle \sim C_1(k) 2^n / n^{2k-2-\max d_r}, n \to \infty, k > 2$$

.

5.4 Proposition. *(necessary condition for the existence of a tree-like structure).*
If H *is an* $m \times n$ *matrix with* N_0 *zero elements, then for it to have a tree-like structure with* k *blocks it is necessary that* $n \geq 2k-1, m \geq k$ *and*

$$N_0 \geq (k-2)(2m + n - 2k + d^* + 2) - m(d^* - 2) - 3k + 2d^* + 4, k > 2.$$

6 Conclusion and future work

In this paper the class of discrete optimization problems with a tree-like structure is considered and based on nonserial dynamic programming (NSDP), and algorithms are proposed for solving this class of problems. Due to the fact that an NSDP algorithm is exact it has the known drawback of worst case exponential time performance. To cope with this difficulty, we plan to consider in the future questions of application of a postoptimality analysis in the NSDP (each step of the proposed algorithm consists of solving a series related integer programming problems), design of approximate versions of NSDP algorithms, and the use of relaxations and fathoming in the algorithmic scheme of NSDP.

References

[1] S. Arnborg, J. Lagergren, and D. Seese, Easy problems for tree-decomposable graphs, J. of Alg., 1991, 12: 308-340.

[2] J.R.S. Blair, B. Peyton, An introduction to chordal graphs and clique trees. In: Graph theory and sparse matrix computation. New York: Springer, 1993:1-29.

[3] U. Bertele and F. Brioschi, Nonserial Dynamic Programming, Academic Press. New York, 1972.

[4] H. L. Bodlaender, A tourist guide through treewidth, Acta Cybern. 11 (1993), No.1-2, 1-21 (1993).

[5] H. L. Bodlaender, Treewidth: Algorithmic techniques and results. Privara, L. (ed.) et al., Mathematical foundations of computer science 1997. 22nd international symposium, MFCS '97, Bratislava, Slovakia, August 25-29, 1997. Proceedings. Berlin: Springer. Lect. Notes Comput. Sci. 1295 (1997), 19–36 (1997)

[6] R. Dechter and Y. El Fattah, Topological parameters for time-space tradeoff. Artif. Intell. 125 (2001), No.1–2, 93–118 (2001).

[7] Finkel'shtein, Yu.Yu. On solving discrete programming problems of special form. - Economics and Math. Methods, 1965, 1, N 2, p.262–270 (Russian).

[8] C.A. Floudas, Nonlinear and mixed-integer optimization: fundamentals and applications. Oxford Univ.Press, Oxford, 1995.

[9] F. V. Fomin, I. Todinca and D.Kratsch. Exact (exponential) algorithms for treewidth and minimum fill-in. Report NO 268 May 2004, Department of Informatics, Bergen: University of Bergen.

[10] G. Gottlob, N. Leone and F.Scarcello, Hypertree decompositions: A survey. Sgall, Jir(ed.) et al., Mathematical foundations of computer science 2001. 26th international symposium, MFCS 2001, August 27-31, 2001. Proceedings. Berlin: Springer. Lect. Notes Comput. Sci. 2136, 37–57 (2001).

[11] P. Heggernes. Treewidth, partial k-trees, and chordal graphs. Internet document: http://www.ii.uib.no/ pinar/chordal.pdf

[12] J.N. Hooker, Logic-based methods for optimization: combining optimization and constraint satisfaction. John Wiley Sons, 2000. - 495 pp.

[13] J. Kallrath, ed., Modeling languages in mathematical optimization. Kluwer, Dordrecht, 2003.

[14] T.Kloks, Treewidth, Lecture Notes in Computer Science 842, Springer-Verlag, 1994.

[15] A.M.C.A. Koster, van Hoesel and A. W.J. Kolen, Solving frequency assignment problems via tree-decomposition, Broersma, H. J. (ed.) et al., 6th Twente workshop on graphs and combinatorial optimization. Univ. of Twente, Enschede, Netherlands, May 26-28, 1999. Extended abstracts. Amsterdam: Elsevier. Electron. Notes Discrete Math. 3, no pag., electronic only (1999)

[16] G. L. Nemhauser, L. A. Wolsey, Integer and Combinatorial Optimization, John Wiley Sons, Inc., 1988.

[17] I. Nowak, Lagrangian decomposition of block-separable mixed-integer all-quadratic programs,Math. Programming, 102, N2, 2005:295312.

[18] T.K. Ralphs, M.V. Galati, Decomposition in Integer Linear Programming, In: Integer Programming: Theory and Practice, John Karlof, ed., 2005.

[19] N. Robertson, P.D. Seymour. Graph minors. II. Algorithmic aspects of tree width. J.Algorithms, 7:309-322, 1986.

[20] O.A. Shcherbina, On local algorithms of solving discrete optimization problems, Problems of Cybernetics, Moscow, 1983, N 40, p. 171–200. (in Russian)

[21] O.A. Shcherbina, Application of local algorithms to the solution of problems of integer linear programming. (Russian) Vopr. Kibern., Mosk. 133 (1989), 19–34 (1989). (in Russian)

[22] O.A. Shcherbina, Local algorithms for block-tree problems of discrete programming, U.S.S.R. Comput. Math. Math. Phys. 25 (1985), No.4, 114–121.

[23] F. Vanderbeck and M.W.P. Savelsbergh (2005). A Generic View of Dantzig-Wolfe Decomposition for Integer Programming. To appear in Operations Research Letters.

[24] Tony J. VAN ROY. Cross decomposition for mixed integer programming with applications to facility location. In J.P. Brans (ed.), Operations Research 81. Amsterdam, North-Holland, 579-587, 1981.

[25] G.J. Woeginger. Exact algorithms for NP-hard problems: A survey, In: "Combinatorial Optimization - Eureka! You shrink!". M. Juenger, G. Reinelt and G. Rinaldi (eds.). LNCS 2570, Springer, 2003: 185-207.

[26] Yu.I. Zhuravlev, Local algorithm of information computation. I,II. Kibernetika, 1 (1965), N1, 12–19; 2 (1966), N2, 1–11 (in Russian).

[27] Yu.I. Zhuravlev and Yu.Yu. Finkelshtein, Local algorithm for integer linear programming problems, Problems of Cybernetics, 14 (1965), Moscow (in Russian).