A Solution Approach to Scheduling Problems with Generalized Variable Upper Bounds

Kwang Min Yang

Chung-Ang University

본 논문은 스케줄링 문제등 정수계획문제에서 흔히 찾아 볼 수 있는 변수상한 제약을 갖는 문제의 효율적인 해법을 위한 분석적인 틀을 마련하기 위한 연구이다. 본 연구에서는 이 문제에 관한 기존의 알고리즘적 해법과 달리 분해기법을 이용하여 일반변수상한 문제에 까지 확장 적용될 수 있음을 분석적으로 보였다.

I. Introduction

This paper proposes a solution approach to scheduling problems with many constraints of the following type:

$$\sum_j x_{ij} \leq y_k \quad , \quad i \in I, \; k \in K.$$

We call the above type of constraints generalized variable upper bound (GVUB) constraints. The variable y may appear in any number of constraints. Schrage [7] coined the name for a single variable type constraint, a variable upper bound constraint (VUB) and developed an algorithmic solution approach to the problem. We present a solution approach based on factorization and extend it for more generalized problems. The factorization approach has been successful in problems with generalized upper bound (GUB) type constraints [3]. This research will focus on the analysis why the VUB type constraints are amenable to the factorization approach and deal with implementation issues.

These types of constraints are frequently found in integer programming problems, such as location problems, job shop and priority scheduling problems. Since the VUB type constraints are most of the constraints in these problems, the success of any algorithm depends on the effectiveness of handling these constraints both in terms of storage space and solution time. Specific examples of these constraints in various models are listed in Section II.

II. Examples

Of the many models with VUB, the uncapacitated facility location problem is a representative. x_{ij} is the fraction of location $j \in J$'s demand supplied from facility $i \in I$, y_i is 1 if facility i is open and 0 otherwise. c_{ij} is the related variable cost for supplying j's demand from facility i; and $f_i \geq 0$ is the cost of opening the facility i.

The model formulation is:

$$\text{Minimize} \quad \sum_i \sum_j c_{ij} x_j + \sum_i f_i y_i$$

$$\text{subject to} \quad \sum_i x_{ij} = 1, \; j \in J$$

$$x_{ij} \leq y_i, \; i \in I, \; j \in J \quad \text{(VUB constraints)}$$

$$x_{ij} \geq 0, \quad i \in I, \; j \in J$$

$$y_i \in \{0,1\}, \; i \in I.$$

Note that the above model with VUB type constraints is a tighter formulation than the one below

$$\sum_j x_{ij} \leq n y_i, \quad i \in I.$$

Although the formulation gives us a strong bound but at the expense of solving larger problems. This makes us to solve these problems with efficiency. Erlenkotter [2] developed an efficient algorithm for this problem based on the dual characterization of the problem.

By introducing the following constraint into the model we generalize the model to p-median problem [1].

$$\sum_i y_i \leq p$$

where $p$is the maximum number of plants.

Another example can be found in a least deviation problem. In this problem we are to find the coefficients of a regression line that minimize the maximum deviation (error) of the fit. This also can be considered as a multi-attribute problem with g_i being individual targets.

$$\begin{aligned}
&\text{Minimize } y \\
&\text{subject to } \sum_j a_{ij} x_j = g_i + \delta_i^+ - \delta_i^- \\
&\qquad \delta_i^+ \leq y_i, \ i \in I \ \text{(VUB)} \\
&\qquad \delta_i^- \leq y_i, \ i \in I \ \text{(VUB)} \\
&\qquad y_i, x_{ij}, \delta_i^+, \delta_i^- \geq 0.
\end{aligned}$$

Committee scheduling problems also contain VUB [10]. x_{ijk} is 1 if member j meets in time slot $k$with topic i; otherwise 0. y_{ik} being logical variable needed to assure that the required number (n) of committee members must be met. Let p_{ijk} be preference of member i over j, k.

$$\begin{aligned}
&\text{Maximize } \sum_i \sum_j \sum_k p_{ijk} x_{ijk} \\
&\text{subject to } \sum_i x_{ijk} \leq 1, \ j \in J, k \in K \\
&\qquad \sum_j x_{ijk} = n y_{ik}, \ i \in I, \ k \in K \\
&\qquad \sum_k y_{ik} = 1, \ i \in I \\
&\qquad x_{ijk} \leq y_{ik}, \ i \in I, \ j \in J, \ k \in K \ \text{(VUB)} \\
&\qquad x_{ijk}, \ y_{ik} \in \{0,1\}.
\end{aligned}$$

See Schrage [7] for other examples with VUB constraints.

VUB and GVUB type constraints are mostly found in scheduling type problems where these constraints serve to enforce logical relations among variables or constraints. As shown in the examples, one characteristic of this type of model formulations is that most constraints are VUB type constraints. It is, therefore, imperative to take care of these VUB type constraints in an effective way if we can ever solve these large problems. We propose an approach to solve this type of problems using factorization which has shown good computational results in large scale problems with many GUB constraints.

III. Solution Approach

The solution approach adopted here is primarily based on Graves and McBride's [3] factorization approach. This approach has a merit of providing us a framework with which we can devise a way to exploit the particular structure of the problems on hand. Factorization has been successful in solving large scale LP problems with many GUB type problems [3]. We will show that the factorization can also be applied to VUB type problems.

III.1 Factorization

Consider the problem

$$\begin{aligned} &\text{Maximize } cx \\ &\text{subject to } Ux \le b \quad \text{(special constraints)} \\ &\qquad\qquad Lx \le r \quad \text{(general constraints)} \\ &\qquad\qquad -Ix \le 0, \end{aligned}$$

where U is $p \times n$ and L is $m \times b$. The U-type constraints are specially structured constraints. After row and column permuted partition of U, L, b, r, and c, we have

$$\begin{bmatrix} U_1 & U_2 & U_3 & b \\ L_1 & L_2 & L_3 & r \\ c_1 & c_2 & c_3 & 0 \end{bmatrix} \equiv \begin{bmatrix} U_{11} & U_{12} & U_{13} & b_1 \\ U_{21} & U_{22} & U_{23} & b_2 \\ L_{11} & L_{12} & L_{13} & r_1 \\ L_{21} & L_{22} & L_{23} & r_2 \\ c_1 & c_2 & c_3 & 0 \end{bmatrix} \quad (1)$$

After multiple exchange of constraints (block pivot), we obtain the factored tableau corresponding to any particular row basis. (Note that in Graves' algorithm row basis are used that is different from ordinary simplex method.) See Graves and McBride's [3] for details.

The complete factored tableau is

$$
\begin{array}{c} \text{(j)} \\ \text{(i)} \\ \text{(ii)} \end{array}
\begin{bmatrix}
[I+U_{11}^{-1}U_{12}\widetilde{A}_{11}^{-1}L_{11}]U_{11}^{-1} & -U_{11}^{-1}U_{12}\widetilde{A}_{11}^{-1} & U_{11}^{-1}[U_{13}-U_{12}\widetilde{A}_{11}^{-1}\widetilde{A}_{12}] & U_{11}^{-1}[b_1-U_{12}\widetilde{A}_{11}^{-1}\widetilde{r}_1] \\
[\widetilde{U}_{22}\widetilde{A}_{11}^{-1}L_{11}-U_{21}]U_{11}^{-1} & -\widetilde{U}_{22}\widetilde{A}_{11}^{-1} & \widetilde{U}_{23}-\widetilde{U}_{22}\widetilde{A}_{11}^{-1}\widetilde{A}_{12} & \widetilde{b}_2-\widetilde{U}_{22}\widetilde{A}_{11}^{-1}\widetilde{r}_1 \\
-\widetilde{A}_{11}^{-1}L_{11}U_{11}^{-1} & \widetilde{A}_{11}^{-1} & \widetilde{A}_{11}^{-1}\widetilde{A}_{12} & \widetilde{A}_{11}^{-1}\widetilde{r}_1 \\
[\widetilde{A}_{21}\widetilde{A}_{11}^{-1}L_{11}-L_{21}]U_{11}^{-1} & -\widetilde{A}_{21}\widetilde{A}_{11}^{-1} & \widetilde{A}_{22}-\widetilde{A}_{21}\widetilde{A}_{11}^{-1}\widetilde{A}_{12} & \widetilde{r}_2-\widetilde{A}_{21}\widetilde{A}_{11}^{-1}\widetilde{r}_1 \\
[\widetilde{c}_2\widetilde{A}_{11}^{-1}L_{11}-c_1]U_{11}^{-1} & -\widetilde{c}_2\widetilde{A}_{11}^{-1} & \widetilde{c}_3-\widetilde{c}_2\widetilde{A}_{11}^{-1}\widetilde{A}_{12} & -\widetilde{c}y^0
\end{bmatrix} \qquad (2)
$$

where

$$
\begin{array}{ll}
\widetilde{U}_{22}=U_{22}-U_{21}U_{11}^{-1}U_{12}, & \widetilde{U}_{23}=U_{23}-U_{21}U_{11}^{-1}U_{13}, \\
\widetilde{A}_{11}=L_{12}-L_{11}U_{11}^{-1}U_{12}, & \widetilde{A}_{12}=L_{13}-L_{11}U_{11}^{-1}U_{13}, \\
\widetilde{A}_{21}=L_{22}-L_{21}U_{11}^{-1}U_{12}, & \widetilde{A}_{22}=L_{23}-L_{21}U_{11}^{-1}U_{13}, \\
\widetilde{c}_2=c_2-w_1U_{11}^{-1}U_{12}, & \widetilde{c}_3=c_3-c_1U_{11}^{-1}U_{13}, \\
\widetilde{b}_2=b_2-U_{21}U_{11}^{-1}b_1, & \\
\widetilde{r}_1=r_1-L_{11}U_{11}^{-1}b_1, & \widetilde{r}_2=r_2-L_{21}U_{11}^{-1}b_1.
\end{array} \qquad (3)
$$

With knowledge of the partition in (1) and the original problem data, we can always reconstruct (2) from U_{11} and $\widetilde{A}_{11}^{-1}$.

The algebraic representation of the factored tableau looks very complex, but it becomes quite simple when the special structure of the U type constraints is exploited. For further details, refer to the Graves and McBride's original paper [3].

III.2 Factorization of VUB constraints

We now consider the case where the special constraints (U) are of variable upper bound type. That is,

$$x_{ij} \le y_i, \quad i \in I,\ j \in J.$$

In general,

$$\sum_{k \in K} x_{ijk} - y_i \le b_{ij}, \quad i \in I,\ j \in J.$$

If $K=\{1\}$, $b_{ij}=0$,

$$x_{ij} \le y_i, \quad i \in I,\ j \in J.$$

We prove theorems that will be used for developing a solution approach. Assume U type constraints are consisted of solely GVUB.

Theorem-1. U is totally unimodular.

Proof. 0,1 matrices with consecutive 1's are totally unimodular [9]. By using this theorem we indirectly prove that U is totally unimodular. Non-zero coefficients of y_i variables can always be arranged columnwise consecutively by varying j indices first. Variable x_{ijk} appears only once in the constraints. Therefore U type constraints matrix can always be rearranged as consecutive 1's or -1's. -1's cause no problem. This can always be handled through variable transformation. Q.E.D.

Proposition-1. U_{11} is either an identity matrix or an identity matrix with one or more columns replaced by coefficients of y_i.

Proof. U_{11} must be non-singular for U_{11} being existent. Rows of x_{ijk} are disjoint, that is, columns of x_{ijk} are singletons and not empty. When U_{11} consists of solely coefficients of x_{ij}, U_{11} must be an identity matrix, otherwise singular for having identical columns and null row(s). When y_i enters U_{11} and replaces a column, any one of x_{ijk} with same i in U_{11} must be replaced by it. Otherwise null row entries results, therefore singular. Q.E.D.

Theorem-2. Every row of U_{11} has at most two nonzeroes (either +1 or -1, or +1 and -1) elements.

Proof. Since x_{ijk} columns are singleton, no x_{ijk} with same i, k *indi*ces can enter U_{11} otherwise singular as proved in Proposition-1. y_i columns are disjoint among themselves. Therefore every row in U_{11} has at most two non-zero elements, one from an x_{ijk} column and the other from y_i column. Q.E.D.

Using Theorem-2 we prove the following main theorem. Without loss of generality we can assume that after column permutation we maintain no diagonal element in U_{11} is zero.

Theorem-3. U_{11} is periodic of period 2, that is $U_{11}^2 = I$.

Proof. Diagonal entries of U_{11} are either +1 or -1.
For diagonal elements in U_{11}^2 : For rows in U_{11} having only one element in their rows produce 1's in their diagonals of U_{11}^2 since they are with the same sign. For rows with two entries (+1 and -1), the diagonal element in U_{11} is always +1 with column singleton therefore the diagonal of U_{11}^2 is always 1.
For off-diagonal elements in U_{11}^2: Singleton rows do not produce off-diagonal elements in U_{11}^2 since these are in diagonal. Rows having two non-zeroes always have opposite signs. Columns having off-diagonal elements have all same (minus) signs. Therefore the resulting inner products for off-diagonals always vanish. Q.E.D.

Theorem-3 proves that we do not need to compute U_{11}^{-1}. Instead we can use U_{11} for U_{11}^{-1} in the factored tableau. Recall that U_{11} in GUB LP is an identity matrix, therefore the

tableau becomes simplified. This property is attributed to the effectiveness of the factorization approach to GUB.

Notice that in (2) and (3) U_{11}^{-1} always comes with the form of either $U_{11}^{-1}U_{12}$ or $U_{11}^{-1}U_{13}$. We now consider the way to generate these products efficiently. Let U_{1x} denote either U_{12} or U_{13}.

Theorem-4. $U_{11}^{-1}U_{1x}$ is constructed from U_{1x} by replacing every singleton column of U_{1x} by corresponding k-th column of U_{11} where $k$is the row index of 1 in the singleton column.

Proof. Post-multiplying U_{11} by a singleton column U_{1x} is equivalent to selecting a column of U_{11}. If the column of U_{1x} is not a singleton column then it must be the coefficients of y_i. Since coefficients of y_i are columnwise disjoint, rows of U_{11} corresponding to non-zero y_i rows have only one entry (i.e., 1) at their diagonal (Recall that U_{11} as permuted to have non-zero diagonal elements). Therefore pre-multiplying U_{11}^{-1} has no effect on U_{1x}. Q.E.D.

Due to Theorem-4 it is possible to construct $U_{11}^{-1}U_{1x}$ by maintaining column indices of U_{11} and U_{1x} without going through the actual computation.

IV. Implementation Considerations

IV.1 Reducing Explicit Part of the Tableau

Assuming all U-type constraints in (1) are transformed to equality constraints after adding slack variables, then the rows labeled (*ii*) in (2) will vanish. Upon block pivoting on these equality constraints at the very beginning of the algorithm, the columns labeled in (*j*) are no longer needed to carry since pivoting on these columns again would violate the equality constraints. The size of the factored tableau needed to carry out the algorithm, therefore, can be reduced.

Adopting a strategy of updating only $\widetilde{A}_{11}^{-1}$ and the rim (bottom row and RHS) data at each iteration and generating the other parts of the tableau as needed to execute the algorithm, the size of the tableau to be maintained will further be reduced.

IV.2 Generation of Implicit Part

Supposing a primal algorithm is executed, we need to generate implicit elements only columnwise to determine the pivot row. The tableau indicates that $U_{11}^{-1}U_{12}$ premultiplies $\widetilde{A}_{11}^{-1}$ and $U_{11}^{-1}U_{13}$ is premultiplied either by L_{11} or L_{12} to generate implicit elements. Since we assumed that the elements of $U_{11}^{-1}U_{1x}$ are generated columnwise, pre- and post-multiplying by $U_{11}^{-1}U_{1x}$ can be accomplished by outer product forms and by inner product forms respectively. All multiplications are executed from right to left.

Since only $\widetilde{A}_{11}^{-1}$ is maintained explicitly and other columns are generated as needed, the work per pivot is determined primarily by the size of $\widetilde{A}_{11}^{-1}$, not by the original problem size.

The size of $\widetilde{A}_{11}^{-1}$ is equal to the number of binding constraints in L-type constraints that is relatively small in comparison with the total number of the constraints. However maintaining A_{11}^{-1} requires to keep the indices of rows and columns of $\widetilde{A}_{11}$ dynamically, which takes extra time and complicates the implementation. Instead if we start with equality L-type constraints by introducing extra variables to the inequalities, an existing LP code such as XMP [5], SPLP [4], or LINDO [8] can readily be used with little modification. This requires more time to deal with larger $\widetilde{A}_{11}^{-1}$ but no time is spent to keep the dynamic changes of $\widetilde{A}_{11}^{-1}$ as we iterate.

V. Extension

The solution approach developed can be extended for the problems with following constraints.

Let $J_i \in \{j|a_{ij}^1 \neq 0\}$ be pairwise disjoint subset of the set $J \supseteq \bigcup_i J_i$ and $I_k \in \{i|a_{ik}^2 \neq 0\}$ be pairwise disjoint subset of the set $I = \bigcup_k I_k$. GVUB constraints are of the form,

$$\sum_j a_{ij}^1 x_j + \sum_k a_{ik}^2 y_k \leq b_i, \quad i \in I$$

$$x_j, y_k \in \{0,1\}.$$

Note that the sign of the coefficients, a, can be either plus or minus.

It is always possible to transform a^2 into -1's and a^1 into +1's by performing column scalings on a first and followed by variable transformations using $x_j' \equiv 1 - x_j$ if necessary.

VI. Conclusion

VUB type constraints are frequently found in ILP problems due to their tighter formulations which in turn provide good bounds in enumeration schemes. However the relatively large number of this type of constraints cause computational burden. Schrage developed an algorithmic solution approach based on the notion of carrying these constraints implicitly like in GUB algorithms.

The approach taken in this research is based on Graves and McBride's factorization approach. The factorization approach has been successful in problems with GUB, embedded network problems [6], and others. This paper showed that the problems with VUB are also amenable to this approach. The advantage of adopting the factorization lies not only in the computational efficiency but also in providing a framework of analyzing the underlying

problem structure that can be exploited in designing an algorithm. Generalization of this approach is possible due to this analytical framework.

The computational efficiency of the algorithm is attributed to obtaining $U_{11}^{-1}U_{1x}$. In GUB U_{11}^{-1} is an identity matrix, therefore, $U_{11}^{-1}U_{1x}$ becomes U_{1x}. The fact that U_{1x} is column singleton makes following computations effortless both in computational time and in data structure.

Although it is not dramatic as in GUB, the required work for constraints with VUB can be similarly reduced. $U_{11}^{-1}U_{1x}$ can be constructed from permuting columns of the original U instead of computing matrix inverse and multiplying matrices. This together with not carrying the specially structured constraints explicitly will result overall computational efficiency over other solution approaches. The computational burden is further reduced by carrying only $\widetilde{A}_{11}^{-1}$ explicitly and generating other elements of the tableau as needed.

The approach adopted in this research can be extended for problems with more generalized constraints such as the one shown in Section V.

Further research must include computational experiments on specific problems which will verify the effectiveness of the proposed approach.

References

1. Conn, A. R. and G. Cornuejols, "A Projection Method for the Uncapacitated Facility Location Problem," *Mathematical Programming 46* (1990), pp. 273-298.

2. Erlenkotter, Donald, "A Dual-Based Procedure for Uncapacitated Facility Location," *Operations Research*, Vol. 26, No.6 (November- December 1978), pp. 992-1009.

3. Graves, G. W. and R. D. McBride, "The Factorization Approach to Large-Scale Linear Programming," *Mathematical Programming*, Vol. 10 (1976), pp. 91-110.

4. Hanson, R. J. and K. L. Hiebert, *A Sparse Linear Programming Subprogram*, Sandia National Laboratory, Report SAND81-0297, 1981.

5. Marsten, Roy E., "The Design of the XMP Linear Programming Library," *ACM Transactions on Mathematical Software*, Vol. 7, No. 4 (December 1981), pp. 481-497.

6. McBride, Richard D., "Solving Embedded Generalized Network Problems," *European Journal of Operational Research*, Vol. 21 (1985), pp. 82-92.

7. Schrage, Linus, "Implicit Representation of Variable Upper Bounds in Linear Programming," *Mathematical Programming Study 4* (1975), pp. 118-132.

8. ------, *User's Manual: Linear, Integer, and Quadratic Programming with Lindo* (2nd ed.), The Scientific Press, 1985.

9. Veinott, A. F., Jr. and H. M. Wagner, "Optimal Capacity Scheduling - I and II," *Operations Research*, Vol. 10 (1962), pp. 518-546.

10. Yang, Kwang Min and Seung-Chul Shin, "A Thesis Committee Scheduling," *Journal of the Korean Operations Research and Management Science Society*, Vol. 15, No. 2 (December 1990), pp.17-31.