International Journal of Pure and Applied Mathematics
Volume 32 No. 4 2006, 37-52

# A BIG-M FREE SOLUTION ALGORITHM FOR GENERAL LINEAR PROGRAMS

Hossein Arsham
Information and Quantitative Science Department
Merrick School of Business
University of Baltimore
1420 North Charles Street, Baltimore, Maryland, 21201-5779, USA
e-mail: harsham@ubalt.edu

**Abstract:** A three-phase simplex type solution algorithm is developed for solving general linear programs. In Phase 1, greater-than constraints are relaxed and the problem is solved starting at the origin. If it is unbounded; then the objective function is modified in order to have a bound feasible solution. Then, in Phase 2 the greater-than constraints are introduced in a dual simplex framework. Following this, in the case the original objective function was modified, Phase 3 restores optimality for the original objective function. The algorithm is numerically stable and works with the original decision variables. Our initial experimental study indicates that the proposed algorithm is more efficient than both the primal and the dual simplex in terms of the number of iterations.

**AMS Subject Classification:** 90C05, 90C06, 49M20, 65K05
**Key Words:** large-scale linear programs, artificial variable free, artificial constraint free, pivot algorithm, advance basis, non-Big-M method

## 1. Introduction

An active research area of linear programming is to construct a initial Simplex tableau which is close to the optimal solution, and to improve its pivoting selection strategies efficiently, see e.g. Vieira and Lins [17], and the references therein. In this paper, we present a new approach to the problem of initialization and pivoting rules: the algorithm is free from any artificial variables and

Received: October 20, 2006

artificial constraints, known as the Big-M methods. By supplying the Simplex method without using any large numbers, therefore the result is computationally stable and provides a better initial basis that reduces the number of pivoting iterations efficiency.

*Group of Simplex Methods.* The primal and the dual simplex methods follow the same general procedure. In the first step a full basis is established. In the proceeding steps we move from one basic solution to another while trying to reduce the degree of infeasibility of the primal and the dual problem respectively.

The primal simplex method starts with the initial basic solution $X = 0$, the first phase tries to find a feasible solution (an extreme point). If this search is unsuccessful, then the procedure terminates with "no feasible solution", otherwise the second phase is initialized with a basic feasibility. One version of the primal simplex known as the two-phase method introduces an *artificial objective function* which is the sum of artificial variables, while the other version adds the penalty terms which is the sum of *artificial variables with very large coefficients*. The latter approach is known as the Big-M method.

The second group of the simplex methods is that of the dual simplex algorithm. When some coefficients in the objective function are positive (i.e. not dual feasible) we must add to the tableau an *artificial constraint* of the type $\sum\limits_j X_j \leq M$ with $M \gg 0$ and the summation is taken over all decision variables with positive coefficients in the objective function.

Both groups require special tools to start up the algorithm when the respective starting conditions are violated. The primal simplex method starts with feasibility conditions satisfied and strives for the optimally condition. When primal feasibility conditions are not satisfied, one must introduce artificial variables in some constraints and penalty (i.e. Big-M) terms in the objective function, see e.g. [7].

For the dual simplex the opposite is true. The algorithm starts with optimally conditions satisfied and strives for feasibility. When the dual feasibility conditions are not satisfied, one must introduce an artificial constraint with a large positive number for its RHS. Both groups include all the constraints in their initial tableau. All these Big-M methods can cause serious error. Too small a value of $M$ could make an infeasible appear feasible. Too large a Big-M can create floating-point computational overflow which destroys the numerical accuracy.

Paparrizos [12] introduced an algorithm which avoids the use of artificial variables through a tedious evaluation of a series of extra objective functions besides the original objective function. The algorithm checks both the optimality and feasibility conditions after completion of iterations. He indicates: “We have not been able to construct such rules”, for entering and exiting variables to and from the basic variable set. The pivot and probe type algorithm suggested in [14] is much complicated. Spivey and Thrall [15] also made an attempt to come up with an algorithm; however it is not for general purpose. Arsham [2-5] introduce an artificial variable free algorithm for general linear program and the classical network problems.

This paper develops a simple alternative approach to solve general LP problems without using any artificial variables or the artificial constraint. Our goal is a unification of simplex and dual simplex which is efficient and eliminates the complexities of artificial variables, the artificial constraint, and artificial objective function. In most cases the proposed approach has a much smaller number of rows in its final tableau than either the simplex or the dual simplex. The current algorithm uses a concept of “active constraints” by relaxation-and-restoration of some constraints. At start all $k$ constraints are relaxed (i.e. temporarily ignored). After solving the sub-problem using only $j$ constraints, we bring in the $k$ constraints. Since the solution may not require all constraints to become active, this process frequently uses smaller tableaux with less number of iterations. Since, however, the complete final tableau may be required for post optimality analysis it can be obtained through an additional efficient procedure. Our approach only in a special case replaces and restores objective function, only once for the entire algorithm. This method does not introduce new rules, but in each phase it uses the customary rules of either the simplex or the dual simplex.

We present the new solution algorithm in Section 2 and give a proof of finiteness. Section 3 deals with the computational aspects of the algorithm with examples to illustrate all aspects of the algorithm. Section 4 presents concluding remarks and future research directions.

## 2. The New Solution Algorithm

The following proposed solution algorithm is a three-phase simplex-type method, which avoids the use of large numbers known as Big-M’s. It combines the primal and dual simplex method; however it uses parts of objective function as an auxiliary objective function whenever necessary to locate a corner point. In Phase 1 some constraints are relaxed to make the origin being feasible (if possible). After solving the relaxed problem using the current solution as the starting point, Phase 2 applies the dual simplex to solve the original problem while Phase 3 applies the primal simplex. Whenever the feasible region is unbounded, in Phase 1 the origin is served as the starting point for Phase 2 to generate a proper corner point by using a "temporary" objective function. Phase 3 restores the original objective function and applies the primal simplex rule to a conclusion. Our initial experimental study with some small size problems indicates that the proposed algorithm is more efficient than both the primal and the dual simplex in terms of the number of iterations.

We start with an LP problem in the following standard form:

$$\text{Max}\, Z = CX\,,$$

Subject to:

$$\begin{aligned} AX &\leq a\,,\\ BX &\geq b\,,\\ \text{all } X &\geq 0\,, \end{aligned}$$

where $b \geq 0$, $a > 0$.

Note that the main constraints have separated into two subgroups. $A$, and $B$ are the respective matrices of constraint coefficients, and $a$, and $b$ are the respective RHS vectors (all with appropriate dimensions). Without loss of generality we assume all the RHS elements are non-negative. We will not deal with the trivial cases such as when $A = B = 0$ (no constraints), or $a = b = 0$ (all boundaries pass through the origin). The customary notation is used: $Z$ for the objective, $C_j$ for the cost coefficients, and $X$ for the decision vector. Here, the word constraint means any constraint other than the non-negativity conditions. In this standard form, whenever the RHS is zero, this constraint should be considered as since elements of vector $b$ is allowed to be 0 but vector $a$ are not. We depart from the usual convention by separating the constraints into two groups according to $\leq$ and $\geq$.

To arrive at this form any general problem some of the following preliminaries may be necessary.

## 2.1. Preliminaries

- Convert unrestricted variables (if any) to non-negative.
- Make all RHS non-negative.
- Convert equality constraints (if any) into a pair of $\geq$ and $\leq$ constraints.

- Remove $\leq$ constraints for, which all coefficients are non-positive. This eliminates constraints, which are redundant with the first quadrant constraint.

- The existence of any $\geq$ constraints for, which all coefficients are non-positive and RHS positive, indicates that the problem is infeasible.

- Convert all $\leq$ inequality constraints with RHS $= 0$ into $\geq 0$. This may avoid degeneracy at the origin.

The following provides an overview of the algorithm's strategy considering the most interesting common case (a mixture of $\geq$ and $\leq$ constraints). This establishes a framework to, which we then extend to complete generality. We proceed as follows:

*Phase 1.* Relax the $\geq$ constraints and solve the reduced problem with the usual simplex. In most cases we obtain an optimal simplex tableau, i.e. a tableau, which satisfies both optimality and feasibility conditions (the discussion of several special cases is deferred for the moment).

*Phase 2.* If the solution satisfies all the relaxed constraints, we are done. Otherwise we bring the most "violated" constraint into the tableau and use dual simplex to restore feasibility. This process is repeated until all constraints are satisfied. This step involves several special cases to be discussed after the primary case.

*Phase 3.* Restore the original objective function (if needed) as follows. Replace the last row in the current tableau to its original values and catch up. Perform the usual simplex. If this is not successful, the problem is unbounded; otherwise the current tableau contains the optimal solution. Extract the optimal solution and multiple solutions (if any).

## 2.2. Special Cases

Three special cases arise in Phase 1, which may require special treatment. If all constraints are $\geq$ and the objective function coefficients satisfy the optimality condition, the origin satisfies the optimality condition but is infeasible. Relax all constraints and start with origin. Some cases may not prove optimal (the relaxed problem has an unbounded feasible region). In Phase 1 the current objective function is changed in a prescribed manner. If all constraints are $\geq$, the origin is not feasible, and if there is any positive $C_j$ we have to change it to 0. This requires restoration of the original objective function, which is done in Phase 3.

Notice that unlike simplex and dual simplex, we focus on the most active constraints rather than carrying all constraints in the initial and all subsequent tableaux. We bring $\geq$ constraints one by one into our tables. This may also

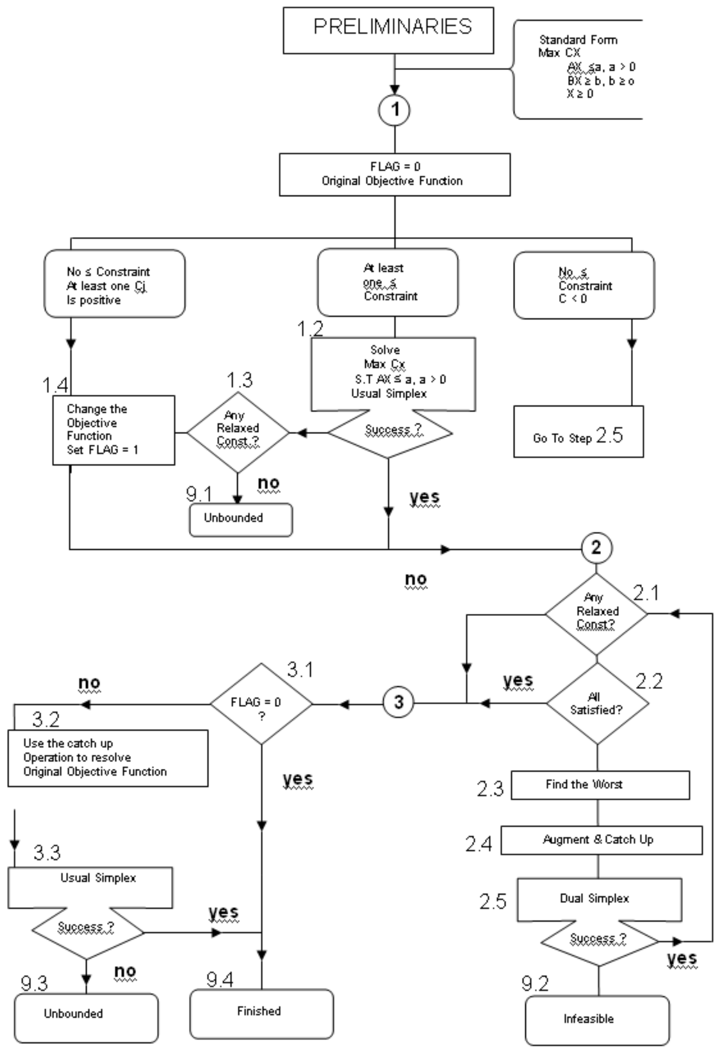

Figure 1:

reduce computational effort.

An overview of the new solution algorithm is outlined in the flowchart presented on Figure 1.

### 2.3. An Overview of the New Solution Algorithm.

**Detailed Process**

*Phase 1.* Recognize three cases as follows:

*Step 1.1.* Set FLAG=0. This indicates the original objective function is still in use. Check the existence of $\leq$ constraints and the signs of the $C_j$s.

Distinguish there, three possibilities:

a) There is at least one $\leq$ constraint. Go to Step 1.2.

b) There are no $\leq$ constraints, and no $C_j$ is positive. Go to Step 2.5.

c) There are no $\leq$ constraints, and at least one $C_j$ is positive (i.e. $A = 0$, $C_j > 0$ for at least one $j$). Go to Step 1.4.

*Step 1.2.* From the standard problem relax temporarily all $\geq$ constraints with positive RHS. Solve the sub-problem using the usual Simplex. If successful go to Phase 2. Otherwise, go to Step 1.3.

*Step 1.3.* Are there any $\geq$ constraints? If yes, go to Step 1.4. Otherwise go to Step 9.1.

*Step 1.4.* Change all $C_j > 0$ elements to 0. Set the FLAG=1. The origin is the starting point. Go to the Phase 2.

*Step 9.1.* Original problem is unbounded.

*Phase 2.* Bring in relaxed constraints (if needed) as follows:

Step 2.1 If there are any remaining relaxed $\geq$ constraint(s), go to Step 2.2. Otherwise, go to the Phase 3.

*Step 2.2.* Does the current optimal solution *satisfy* all the still relaxed constraints? If yes, go to the Phase 3. Otherwise, go to Step 2.3.

*Step 2.3.* Find the most violated relaxed constraint (substitute the current solution, $X$, into each constraint and select the constraint with the largest absolute difference between the right and left sides (arbitrary tie-breaker)). Continue with Step 2.4.

*Step 2.4.* Bring the chosen constraint into the current tableau.

a) Convert the chosen constraint into a $\leq$ constraint and add a slack variable.

b) Use the Gauss operation to transform the new constraint into the current nonbasic variables. (i.e. multiply the row for each basic variable by the coefficient of the variable in the new constraint; sum these product rows and subtract this computed row from the new row). Go to Step 2.5

*Step 2.5.* Iterate dual simplex rules to a conclusion: The dual simplex rule: Use the most negative RHS to select the pivot row (if none is successful conclusion for Step 2.5). The pivot column has a negative coefficient in the pivot row. If a tie results, choose the column with smallest R/R (i.e. row $C_j$ / pivot row element wherever the pivot row element is negative). If there is still a tie, choose one arbitrary (if there is no positive RR the problem is infeasible i.e. Step 9.2). Generate the next tableau by using the Gauss pivot operation and repeat the above. If successful (RHS is nonnegative), go to Step 2.1. Otherwise, the problem proves infeasible (no negative element in the outgoing row), go to Step 9.2.

*Step 9.2.* Original problem is infeasible.

*Phase 3.* Restore the original objective function (if needed) as follows:

*Step 3.1.* If FLAG=0, go to Step 4.1; otherwise go to Step 3.2.

*Step 3.2.* Replace the $C_j$ in the current tableau to their original values and catch up. Go to Step 3.3.

*Step 3.3.* Perform the usual simplex pivoting; if not successful the problem is unbounded go to Step 9.3. Otherwise, go to Step 4.1.

*Step 9.3.* Original problem is unbounded.

*Step 9.4. Finished.* The current tableau contains the optimal solution. Extract the optimal solution and multiple solutions (if any, i.e. whenever the number of $C_j$=0 in the final tableau exceeds the number of basic variables). In order to perform the usual post-optimality analysis we can augment the optimal tableau, whenever there are some unused constraints, to obtain the usual final tableau. This can be done easily and efficiently by applying operations a and b of Step 2.4 in turn to each of the non-active constraints.

Notice that the unboundedness can be checked when the selected pivot column has no positive element. Infeasibility is checked whenever the selected pivot row has no negative element.

*Proof.* The general LP problem is deformed into a sequence of sub-problems, each of which is handled by either simplex or dual simplex as appropriate. The proof that these algorithms terminate successfully is well known. The details of the deformation and reformation of the problem is given in sufficient detail that the proof that the algorithm works and terminates successfully for all cases becomes obvious. □

Notice that the proposed algorithm starts with a basic solution to the sub-problem and then reduces the dual and primal infeasibility in Phase 2 and Phase 3 respectively. The algorithm takes advantage of the optimal solution available from the previous iteration (i.e. re-optimization rather that solving each problem from any scratches). The initial active set includes all constraints with $a > 0$. As an alternative to the Phase 1, one may use the simple bounds $X \times 0$ as the active constraints.

## 3. Applications and Illustrative Numerical Example

This section deals with the computational aspects of the algorithm with numerical examples to illustrate all aspects of the algorithm.

**Example 1.** Consider the following problem:

$$\begin{aligned} & \text{Max } -X1-4X2-2X3\,, \\ \text{subject to: } & X1-X2+X3=0\,, \\ & -X2+4X3=1\,, \\ & \text{and } X1,X2,X3\geq 0. \end{aligned}$$

Converting the equality constraints to inequality constraints, we have:

$$\begin{aligned} & \text{Max } -X1-4X2-2X3\,, \\ \text{subject to: } & X1-X2+X3\leq 0\,, \\ & -X2+4X3\leq 1\,, \\ & X1-X2+X3\geq 0\,, \\ & -X2+4X3\geq 1\,, \\ & \text{and } X1,X2,X3\geq 0. \end{aligned}$$

Relaxing all $\geq$ constraints, we have:

$$\begin{aligned} & \text{Max } -X1-4X2-2X3\,, \\ \text{subject to: } & X1-X2+X3\leq 0\,, \\ & -X2+4X3\leq 1\,, \\ & \text{and } X1,X2,X3\geq 0. \end{aligned}$$

*Phase 1.* The initial tableau with the two slack variables as its BV is:

| | **X1** | **X2** | **X3** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|---|
| **S1** | **1** | **-1** | **1** | **1** | **0** | **0** |
| **S2** | **0** | **-1** | **4** | **0** | **1** | **1** |
| $C_j$ | **-1** | **-4** | **-2** | **0** | **0** | |

The origin is the solution to the relaxed problem.

*Phase 2.* Bringing in the violated constraint, we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **S2** | **S3** | **RHS** |
|---|---|---|---|---|---|---|---|
| **S1** | **1** | **-1** | **1** | **1** | **0** | **0** | **0** |
| **S2** | **0** | **-1** | **4** | **0** | **1** | **0** | **1** |
| **S3** | **0** | **1** | **-4** | **0** | **0** | **1** | **-1** |
| $C_j$ | **-1** | **-4** | **-2** | **0** | **0** | **0** | |
| **R/R** | – | – | **1/2** | – | – | | |

The variable to go out is S3 and the variable to come in is X3. After pivoting we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **S2** | **S3** | **RHS** |
|---|---|---|---|---|---|---|---|
| **S1** | **1** | **-3/4** | **0** | **1** | **0** | **1/4** | **-1/4** |
| **S2** | **0** | **0** | **0** | **0** | **1** | **1** | **0** |
| **X3** | **0** | **-1/4** | **1** | **0** | **0** | **-1/4** | **1/4** |
| $C_j$ | **-1** | **-9/2** | **0** | **0** | **0** | **-1/2** | |
| **R/R** | – | **6** | – | – | – | – | |

The variable to go out is S1 and the variable to come in is X2. After pivoting we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **S2** | **S3** | **RHS** |
|---|---|---|---|---|---|---|---|
| **X2** | **-4/3** | **1** | **0** | **-4/3** | **0** | **-1/3** | **1/3** |
| **S2** | **0** | **0** | **0** | **0** | **1** | **1** | **0** |
| **X3** | **-1/3** | **0** | **1** | **-1/3** | **0** | **-1/3** | **1/3** |
| $C_j$ | **-7** | **0** | **0** | **-6** | **0** | **-2** | |

This is the end of Phase 2. The optimal solution for this tableau is X1 = 0, X2 = 1/3, X3 = 1/3, which satisfies all the constraints. Therefore, it must be the optimal solution to the problem. Notice that since all constraints are in equality form, all slack/surplus variables are all equal to zero, as expected. The optimal value, which is found by plugging the optimal decision into the original objective function, we get the optimal value = -2.

**Example 2.** Now consider the he following problem:

$$\begin{aligned} &\text{Maximize } 3X1 + X2 - 4X3\,, \\ \text{subject to: } & X1 + X2 - X3 = 1\,, \\ & X2 \geq 2\,, \\ & X1 \geq 0. \end{aligned}$$

Converting the equality constraints to inequality constraints, we have:

$$\text{Maximize } 3X1 + X2 - 4X3\,,$$

$$\text{subject to:}\quad X1 + X2 - X3 \le 1\,,$$
$$X1 + X2 - X3 \ge 1\,,$$
$$X2 \ge 2\,,$$
$$X1 \ge 0\,.$$

*Phase 1.* Relaxing all $\ge$ constraints, we have:

$$\text{Maximize}\, 3X1 + X2 - 4X3\,,$$
$$\text{subject to:}\quad X1 + X2 - X3 \le 1\,.$$

The initial tableau with the one slack variable as its BV is:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **RHS** |
|---|---|---|---|---|---|
| **S1** | **1** | **1** | **-1** | **1** | **1** |
| $C_j$ | **3** | **1** | **-4** | **0** | |

The variable to come in is X1, after pivoting we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **RHS** |
|---|---|---|---|---|---|
| **X1** | **1** | **1** | **-1** | **1** | **1** |
| $C_j$ | **0** | **-2** | **-1** | **-3** | |

The optimal solution is X1 = 1, X2 = 0, X3 = 0. This solution does not satisfy the constraint X2 $\ge$ 2.

*Phase 2.* Bringing in the violated constraint, we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|---|
| **X1** | **1** | **1** | **-1** | **1** | **0** | **1** |
| **S2** | **0** | **-1** | **0** | **0** | **1** | **-2** |
| $C_j$ | **0** | **-2** | **-1** | **-3** | **0** | |
| **R/R** | – | **2** | – | – | **-** | **-** |

The variable to go out is S2, and the variable to come in is X2. After pivoting, we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|---|
| **X1** | **1** | **0** | **-1** | **1** | **1** | **-1** |
| **X2** | **0** | **1** | **0** | **0** | **-1** | **2** |
| $C_j$ | **0** | **0** | **-1** | **-3** | **-2** | |
| **R/R** | – | – | **1** | – | – | |

The variable to go out is X1, and the variable to come in is X3. After pivoting, we have:

| **BVS** | **X1** | **X2** | **X3** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|---|
| **X3** | **-1** | **0** | **1** | **-1** | **-1** | **1** |
| **X2** | **0** | **1** | **0** | **0** | **-1** | **2** |
| $C_j$ | **-1** | **0** | **0** | **-4** | **-3** | |

The optimal solution for the relaxed problem is X1 = 0, X2 = 2, X3 = 1. This solution satisfies all the constraints; therefore, it must be the optimal solution to the original problem.

**Example 3.** Consider the following problem:

$$\begin{aligned} &\text{Max } -40X1 - 50X2\,, \\ \text{subject to: } &\ 2X1 + X2 \geq 5\,, \\ &X1 + 2X2 \geq 3\,, \\ &\text{and } X1, X2 \geq 0. \end{aligned}$$

*Phase 1.* There are no $\leq$ constraints, and no $C_j$ is positive. Go to Step 2.5. To perform the dual simplex, we convert the problem into the dual standard form:

$$\begin{aligned} &\text{Max } -40X1 - 50X2\,, \\ \text{subject to: } &\ -2X1 - X2 + S1 = -5\,, \\ &-X1 - 2X2 + S2 = -3\,, \\ &\text{and } X1, X2 \geq 0. \end{aligned}$$

The initial tableau with the two slack variables as its BV is:

| | **X1** | **X2** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|
| **S1** | **-2** | **-1** | **1** | **0** | **-5** |
| **S2** | **-1** | **-2** | **0** | **1** | **-3** |
| $C_j$ | **-40** | **-50** | **0** | **0** | |

S1 goes out and X1 comes in, after pivoting we have:

| | **X1** | **X2** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|
| **X1** | **1** | **1/2** | **-1/2** | **0** | **5/2** |
| **S2** | **0** | **-3/2** | **-1/2** | **1** | **-1/2** |
| $C_j$ | **0** | **-30** | **-20** | **0** | |

Now, S2 goes out and X2 comes in, after pivoting we have:

| | **X1** | **X2** | **S1** | **S2** | **RHS** |
|---|---|---|---|---|---|
| **X1** | **1** | **0** | **-2/3** | **1/3** | **7/3** |
| **X2** | **0** | **1** | **1/3** | **-2/3** | **1/3** |
| $C_j$ | **0** | **0** | **-10** | **-20** | |

The optimal solution for this problem is X1 = 7/3, X2 = 1/3.

**Example 4.** As out last numerical example, consider the following problem:

$$\begin{aligned} & \text{Max } -X1+2X2\,, \\ \text{subject to: } & X1+X2 \geq 2\,, \\ & -X1+X2 \geq 1\,, \\ & X2 \leq 3\,, \\ & \text{and } X1, X2 \geq 0. \end{aligned}$$

*Phase 1.* Relaxing the $\geq$ constraints, the initial tableau is:

| **BVS** | **X1** | **X2** | **S3** | **RHS** | **C/R** |
|---|---|---|---|---|---|
| **S3** | **0** | **1** | **1** | **3** | **3/1** |
| $C_j$ | **-1** | **2** | **0** | **0** | |

The variable to come in is X2, after pivoting we have:

| **BVS** | **X1** | **X2** | **S3** | **RHS** |
|---|---|---|---|---|
| **X2** | **0** | **1** | **1** | **3** |
| $C_j$ | **-1** | **0** | **-2** | |

The optimal solution for the relaxed problem is X1 = 0, X2 = 3. The solution satisfies all the constraints; therefore, it must be the optimal solution to the original problem.

## 4. Concluding Remarks

A new solution algorithm for the general linear programs has been presented. It favorably compares with both primal and dual simplex methods. The classical primal simplex and the dual simplex include all of the constraints in the first and all the subsequent tableaux. We start with only $j$ constraints and bring $k$ constraints into the tables one by one. This frequently reduces computational effort. The proposed algorithm has simplicity, potential for wide adaptation, and deals with all cases.

It is well known that the initial basic solution by the usual primal simplex and dual simplex could be far away from the optimal solution, see e.g. [13]. This motivated relaxation of some of the constraints. Clearly, the application of relaxation to large scale problems lowers complexity see e.g. [1], but the disadvantage is that the rate of convergence could be very slow. As an alternative, one may adapt similar ideas given in [11] for constraint selection criteria. However, his approach is limited to constraints with all RHS $\geq k_0$.

A degenerate vertex can cause computational cycling which is difficult to detect. In the event of degeneracy one must use some anti-degeneracy efficient devices such as ordering, see e.g. [8]. The current algorithm is less likely to cycle since some constraints are brought into the tableau one by one. Similarly redundant constraints create difficulties in most existing solution algorithms. Again since some constraints are brought into the tableau one by one, redundancy is less likely to occur in the algorithm presented.

In comparison with primal and dual simplex methods the new algorithm in tested examples generates the final tableau with a fewer iteration. Our proposed approach is a general purpose method for solving LP problems. Clearly, more specialized methods exist that can be used for solving LP problems with specific properties [6, 10, 16]. To reduce the number of iterations in the proposed algorithm even further one may select the pivot element so as to maximize "total increase" rather than the "rate of increase" in the classical simplex and dual simplex, see [9].

An area for immediate future research is the development of efficient code to implement this approach and perform an extensive study of computational efficiency compared with existing methods. The implementation should use the state-of-the-art practice instead of tableau-type. At this stage, a convincing evaluation for the reader would be the application of this approach to a problem he/she solved by any other methods.

## Acknowledgments

I am most appreciative to the reviewers for their useful comments and suggestions. This work is supported by NSF Grant CCR-9505732.

## References

[1] I. Adler, R. Karp R., R. Shamir, A family of simplex variants solving an mxd linear program in expected number of pivot steps depending on d only, *Mathematics of Operations Research,* **11** (1986), 570-589.

[2] H. Arsham, An algorithm for Simplex tableau reduction: The push-to-pull solution strategy, *Applied Mathematics and Computation*, **137** (2003), 525-547.

[3] H. Arsham, Initialization of the simplex algorithm: An artificial-free approach, *SIAM Review*, **39** (1997), 736-744.

[4] H. Arsham, Affine geometric method for linear programs, *Journal of Scientific Computing*, **12** (1997), 289-303.

[5] H. Arsham, A comprehensive simplex-like algorithm for network optimization and perturbation analysis, *Optimization*, **32** (1995), 211-267.

[6] J. Barnes, R. Crisp JR., Linear programming: A survey of general purpose algorithms, *AIIE Transactions*, **7** (1975), 212-221.

[7] J. Camm, A. Raturi, A. Tsubakitani, Cutting Big-M down to sizes, *Interfaces*, **20** (1990), 61-66.

[8] G. Dantzig, Making progress during a stall in the simplex algorithm, *Linear Algebra and Its Applications*, **114** (1989), 251-259.

[9] J. Forrest, D. Goldfarb, Steepest-edge simplex algorithm for linear programming, *Mathematical Programming*, **57** (1992), 341-374

[10] J. More, S. Wright, *Optimization Software Guide*, SIAM, Philadelphia, PA., (1993).

[11] D. Myers, A dual simplex implementation of a constraint selection algorithm for linear programming, *Journal of the Operational Research Society*, **43** (1992), 177-180.

[12] K. Papparrizos, The two phase simplex without artificial variables, *Methods of Operations Research*, **61** (1990), 73-83.

[13] A. Ravindran, A comparison of the primal-simplex and complementary pivot methods for linear programming, *Naval Research Logistic Quarterly*, **20** (1973), 95-100.

[14] A. Sethi, G. Thompson, The pivot and probe algorithm for solving a linear program, *Mathematical Programming*, **29** (1984), 219-233.

[15] A. Spivey, W. Thrall, *Linear Optimization*, Holt, Rinehart and Winston, London, (1970).

[16] T. Terlaky, S. Zhang, Pivot rules for linear programming: A survey on recent theoretical developments, *Annals of Operations Research*, **46** (1993), 203-233.

[17] H. Vieira Jr., M. Lins, An improved initial basis for the simplex algorithm, *Computers and Operations Research*, **32** (2005), 1983-1993.