Combining Interior and Exterior Simplex Type Algorithms

Themistoklis Glavelis
Department of Applied Informatics,
University of Macedonia,
Economic and Social Sciences
156 Egnatia Str.,
Thessaloniki 54006, Greece
Tel.: (+30) 2310 891824
glavelis@uom.gr

Nikolaos Ploskas
Department of Applied Informatics,
University of Macedonia,
Economic and Social Sciences
156 Egnatia Str.,
Thessaloniki 54006, Greece
Tel.: (+30) 2310 891824
ploskas@uom.gr

Nikolaos Samaras
Department of Applied Informatics,
University of Macedonia,
Economic and Social Sciences
156 Egnatia Str.,
Thessaloniki 54006, Greece
Tel.: (+30) 2310 891866
samaras@uom.gr

ABSTRACT
Linear Programming (LP) is a significant research area in the field of operations research. The simplex algorithm is the most widely used and well-studied method for solving Linear Programming problems (LPs). Many algorithms have been proposed for the solution of LPs. The vast majority of these algorithms belong to three main categories: (i) Simplex-type or pivoting algorithms, (ii) interior-point methods (IPMs) and (iii) exterior point simplex type algorithms (EPSA). The aim of this paper is to present an implementation of a hybrid simplex algorithm that begins to solve the LP using an IPM and after a number of iterations continues with a primal-dual EPSA algorithm. This hybrid approach aims to take advantage of: (i) IPM strengths, which is the fast convergence in the first iterations, and (ii) EPSA strengths, i.e. the fast convergence when making steps in directions that are linear combinations of attractive directions. The interior point that is calculated by IPM after a number of iterations can lead to such attractive directions. In order to gain an insight into the practical behavior of the proposed algorithm, we have performed some computational experiments over sparse randomly generated optimal LPs. Finally, in the computational study that we have conducted, we investigate the adequate number of iterations that IPM should run in order to decrease the CPU time and the iterations of the proposed algorithm.

Categories and Subject Descriptors
G.1.6 [**Optimization**]: Linear Programming

General Terms
Algorithms, Operations Research, Linear Programming.

Keywords
Operations Research, Linear Programming, Simplex Algorithm, Interior Point Method, Exterior Point Simplex Type Algorithms, MATLAB, Computational Study.

1. INTRODUCTION
Linear Programming problem (LP) remains one of the most well-studied optimization problems. A large number of real world problems can be treated as linear problems [2] [3]. LP is an optimization problem that consists of maximizing or minimizing a linear function over a certain domain. The domain consists of a set of linear constraints.

The most well-known method for the optimization problem is the simplex algorithm developed by George B. Dantzig [4]. The simplex algorithm begins with a primal feasible basis and uses pivot operations until an optimum solution is computed. It also guarantees monotonicity of the objective value. Regarding the theoretical complexity of the simplex algorithm, it has been proved that the expected number of iterations in the solution of a linear problem is polynomial [1]. Furthermore, the worst case complexity has exponential behavior [9]. It has been observed that the simplex algorithm performs sufficiently well in practice, especially on small or medium sized LPs, but its performance is not satisfactory in large-scale LPs. This weakness of simplex algorithm stems from the stalling and cycling problem. Many anti-cycling pivoting rules have been introduced in the past [13].

Since Dantzig's initial contribution, researchers have made many efforts in order to enhance the performance of simplex algorithm. In the 1980s the monopoly of the simplex algorithm in the solution of the LP has been challenged. Interior Point Methods (IPM) were the result of subsequent research and their performance has been more than satisfactory compared to the simplex algorithm [8]. The main idea of IPMs is that the computation of the optimal solution can be achieved by moving inside the feasible region. The next research step was the attempt to combine the simplex algorithm and IPMs in order to enhance the computational behavior of software packages [5]. Consequently, researchers concluded that the most effective type of algorithms, in computational terms, were the primal-dual algorithms based on IPM and simplex method [7].

Beyond this, in the 1990s a totally different approach arose; namely Exterior Point Simplex Algorithm (EPSA). The first implementation of an EPSA was introduced for the assignment problem [10]. The main idea of EPSA is that it moves in the exterior of the feasible region and constructs basic infeasible solutions instead of feasible solutions calculated by the simplex algorithm. Although EPSA outperforms the original simplex algorithm [11], it has also some computational disadvantages. The main disadvantage is that in many LPs, EPSA can follow a path, which steps away from the optimal solution. This drawback can be avoided if the exterior path is replaced with a dual feasible simplex path. The most effective types of EPSA algorithms are the primal-dual versions. It has been observed that replacing the exterior path of an EPSA with a dual feasible simplex path results in an algorithm free from the computational disadvantages of EPSA [12].

A more effective approach is the Primal-Dual Exterior Point Simplex Algorithm (PDEPSA) [6]. PDEPSA can deal more effectively with the problems of stalling and cycling and as a result improves the performance of the primal dual exterior point algorithms. The advantage of PDEPSA stems from the fact that it uses an interior point in order to compute the leaving variable in contrast to primal dual exterior point algorithms which use a boundary point. For a full description of PDEPSA see [6].

This paper proposes a hybrid algorithm that combines an IPM and PDEPSA, which is the most effective exterior point algorithm. The advantage of this algorithm is that it exploits the strengths of both IPM and PDEPSA. Initially, IPM is used to compute an interior point and after a few iterations PDEPSA continues to calculate the optimal solution using a direction to the feasible region according to the interior point that was found by IPM. The main idea of the algorithm is that after a few iterations IPMs do not result in great enhancements of the objective function's value. Consequently, we can use PDEPSA in the second stage to compute the optimal solution in a few iterations.

The paper is organized as follows. Following introduction, in Section 2 PDEPSA is presented. Section 3 includes the description of the general framework of the proposed algorithm. In Section 4 the computational study is presented. Finally, in Section 5 there are the conclusions and possible enhancements of the proposed algorithm.

2. ALGORITHM DESCRIPTION

In this section we briefly describe PDEPSA. For a full description of PDEPSA see [6].

Consider now the following linear program in the standard form.

$$\begin{aligned} \min \quad & c^T x \\ \text{s. t.} \quad & Ax = b \qquad \text{(LP)} \\ & x \geq 0 \end{aligned}$$

where $A \in \Re^{mxn}$, $c,x \in \Re^n$, $b \in \Re^m$ and T denotes transpose. Assume that A has full rank, rank(A)=m (m<n).

The dual problem associated with (LP) is

$$\begin{aligned} \max \quad & b^T x \\ \text{s. t.} \quad & A^T w + s = c \qquad \text{(DP)} \\ & s \geq 0 \end{aligned}$$

where $w \in \Re^n$ and $s \in \Re^n$.

The steps that PDEPSA performs are presented below:

Step 0 (Initialization):

A) Start with a dual feasible basic partition (B, N) and an interior point y of (LP).

Set:

$$P = N, Q = \varnothing$$

and compute

$$x_B = (A_B)^{-1} b, w^T = (c_B)^T (A_B)^{-1}, (s_N) = (c_N)^T - w^T A_N$$

B) Compute the direction d_B from the relation: $d_B = y_B - x_B$

Step 1 (Test of optimality and choice of the leaving variable):

If $x \geq 0$, STOP. The (LP) is optimal. Else, choose the leaving variable $x_k = x_{B[r]}$ from the relation:

$$a_l = \frac{x_{B[r]}}{-d_{B[r]}} = \max\{\frac{x_{B[r]}}{-d_{B[r]}} : d_{B[i]} > 0 \wedge x_{B[i]} < 0\}$$

Step 2 (Computation of the next interior point):

Set:

$$a = \frac{a_1 + 1}{2}$$

Compute the interior point: $y_B = x_B + a d_B$

Step 3 (Choice of the entering variable):

Set: $H_{rN} = (B^{-1})_r . A_N$

Choose the entering variable x_l from the relation:

$$\frac{-s_l}{H_{rN}} = \min\{\frac{-s_j}{H_{rj}} : H_{rj} \wedge j \in N\}$$

Compute the pivoting column: $h_l = B^{-1} A_l$

If $l \in$ P,

$P \leftarrow P \backslash \{l\}$

Else

$Q \leftarrow Q \backslash \{l\}$

Step 4 (Pivoting):

Set:

$$B[r] = l \text{ and } Q \leftarrow Q \cup \{k\}$$

Using the new partition (B, N) where N = (P, Q), compute the new basis inverse B^{-1} and the variables:

$$x_B = (A_B)^{-1} b, w^T = (c_B)^T (A_B)^{-1}, (s_N) = (c_N)^T - w^T A_N$$

Go to step 0B.

3. COMBINING IPM-EPSA

In this paper a new algorithm is presented that uses an IPM to approach the solution of the LP and continues with PDEPSA to calculate the optimal solution. The IPM that was used in this implementation is MATLAB's large-scale linprog built-in function. The large-scale algorithm is based on Interior Point Solver [14], a primal-dual interior point algorithm. This hybrid approach takes advantage of: (i) IPM strengths, which is the fast convergence in the first iterations, and (ii) EPSA strengths, i.e. the fast convergence when making steps in directions that are linear combinations of attractive directions, like the interior point that is calculated by IPM after a number of iterations.

IPM solves an LP in few iterations with each iteration being expensive in computational terms, while PDEPSA needs more iterations with a relative small computational cost per iteration. The number of iterations that IPM should run is a matter that is related with the size and the density of the LP. In the computational experiments that follow in the next section, we investigate the adequate number of iterations that IPM should run in order to decrease the CPU time and the iterations of the proposed algorithm.

4. COMPUTATIONAL STUDY

Computational studies are useful tools in order to examine the practical efficiency of an algorithm, or even compare algorithms.

The computational comparison has been performed on a quad-processor Intel Core i7 3.4 GHz with 32 Gbyte of main memory running under Microsoft Windows 7 64-bit. The algorithms have been implemented using MATLAB Professional R2012b. MATLAB (MATrix LABoratory) is a powerful programming environment and is especially designed for matrix computations in general.

The test set used in the computational study was randomly generated. Problem instances have the same number of constraints and variables. The largest problem tested has 4000 constraints and 4000 variables. We have generated these instances with 10% and 20% density. For each problem type of a particular size, 10 instances were generated, using a different seed number. For each instance, we averaged times over 10 runs. All times in the following tables are measured in seconds.

In this computational study three algorithms were compared: (i) MATLAB's primal-dual interior point algorithm (IPM), (ii) PDEPSA, and (iii) the proposed hybrid algorithm. In order to gain an insight into the practical behavior of the proposed algorithm, we have experimented with three different values for the iterations that IPM should run, i.e. 5, 10 and 15.

Table 1 and Table 2 present the results from the execution of the above mentioned algorithms over randomly optimal generated problems with density 10% and 20%, respectively. Figures 1 - 4 are the graphical representation of the results shown in Tables 1 and 2. Figure 1 and Figure 3 present the execution time of the algorithms over randomly optimal generated problems with density 10% and 20%, respectively, while Figure 2 and Figure 4 present the total iterations of the algorithms over randomly optimal generated problems with density 10% and 20%, respectively.

Table 1. Results for Randomly Optimal Generated Problems with Dimension n x n and Density 10%

Density 10%	IPM		PDEPSA		HYBRID (5 iterations)		HYBRID (10 iterations)		HYBRID (15 iterations)	
	time (sec)	iterations	time (sec)	iterations	time (sec)	iterations	time (sec)	iterations	time (sec)	iterations
1000x1000	22.43	19.2	46.61	1,255.1	18.74	261.2	20.23	195.1	19.73	195.1
1500x1500	96.99	24.1	178.64	2,068.9	74.83	600.6	60.68	255.8	60.36	255.5
2000x2000	216.31	24.6	392.19	2,580.4	182.18	821.2	152.91	349.7	170.81	383.9
2500x2500	405.56	28.3	834.93	3,561.3	348.49	1,050.8	344.26	553.2	377.48	547.3
3000x3000	659.92	26.5	1,570.59	4,636.5	589.73	1,256.4	507.88	585.5	537.74	586.6
3500x3500	1,206.15	27.5	2,446.58	5,450.5	937.14	1,568.2	791.21	643.6	963.96	641.2
4000x4000	1,416.30	29.8	3,856.02	6,724.9	1,320.89	1,643.1	1,111.02	668.2	1,149.60	606.1
Average	574.81	25.7	1,332.22	3,753.9	496.00	1,028.8	426.89	464.4	468.52	459.4

Table 2. Results for Randomly Optimal Generated Problems with Dimension n x n and Density 20%

Density 20%	IPM		PDEPSA		HYBRID (5 iterations)		HYBRID (10 iterations)		HYBRID (15 iterations)	
	time (sec)	iterations	time (sec)	iterations	time (sec)	iterations	time (sec)	iterations	time (sec)	iterations
1000x1000	32.87	23.1	38.39	1,040.5	21.76	297.6	29.06	190.9	29.55	190.5
1500x1500	75.64	22.5	158.25	1,831.3	78.11	596.3	68.28	194.2	68.19	194.4
2000x2000	188.62	27.6	371.58	2,459.1	153.74	694.9	150.60	322.6	160.56	326.1
2500x2500	404.17	28.9	792.05	3,311.8	299.62	898.7	282.39	417.1	331.27	410.7
3000x3000	615.72	26.2	1,422.42	4,179.2	482.15	956.1	425.32	399.1	498.24	414.9
3500x3500	885.22	31.1	2,181.22	4,811.2	985.38	1,684.5	657.70	553.5	763.33	462.2
4000x4000	1,125.92	30.0	3,244.93	5,627.5	1,269.22	1,713.5	835.74	578.4	999.92	582.3
Average	475.45	27.1	1,172.69	3,322.9	470.00	977.4	349.87	379.4	407.29	368.7

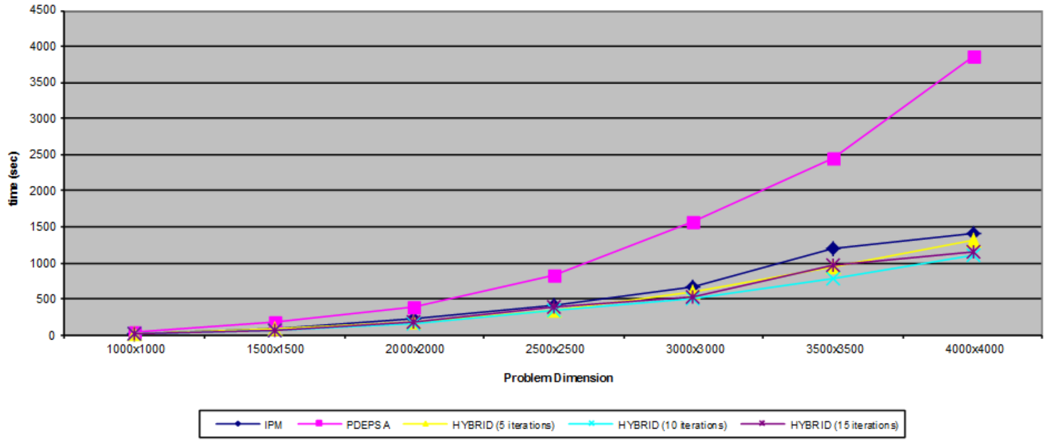

Figure 1. Execution Time of Randomly Generated Optimal LPs with 10% Density

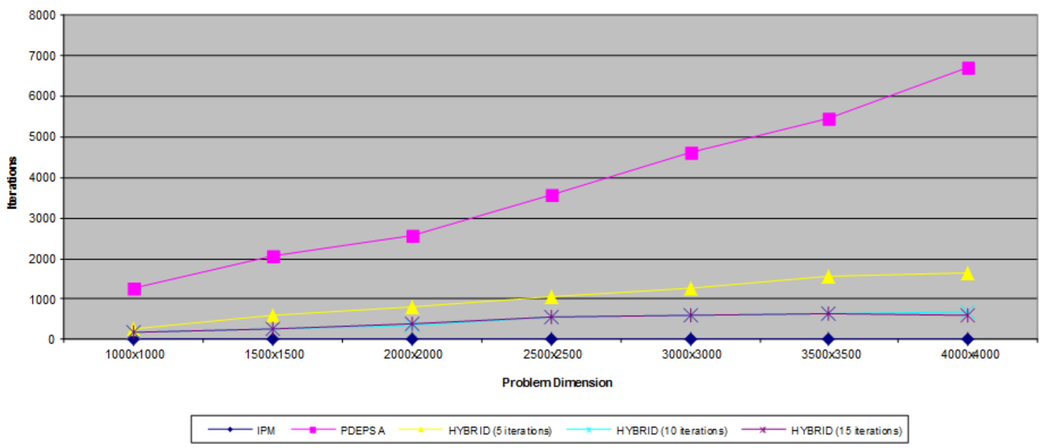

Figure 2. Number of Iterations for Randomly Generated Optimal LPs with 10% Density

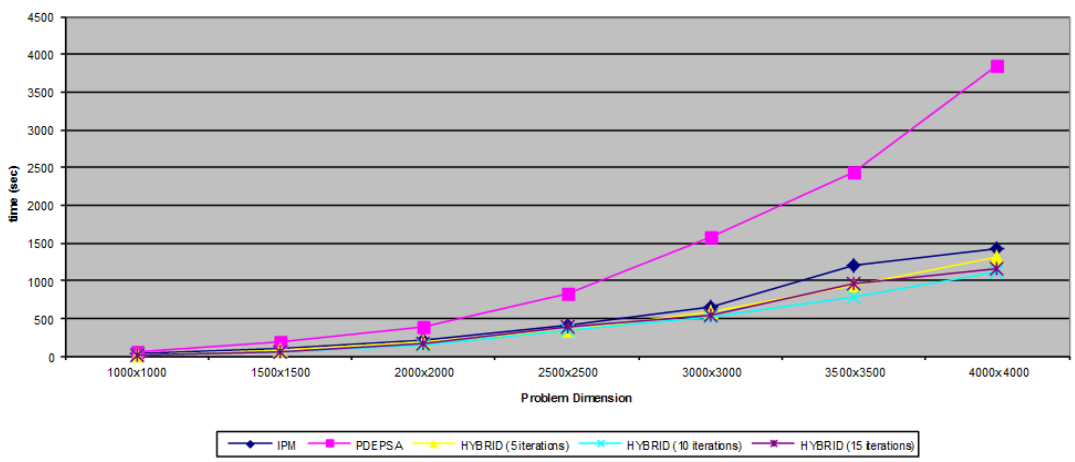

Figure 3. Execution Time of Randomly Generated Optimal LPs with 20% Density

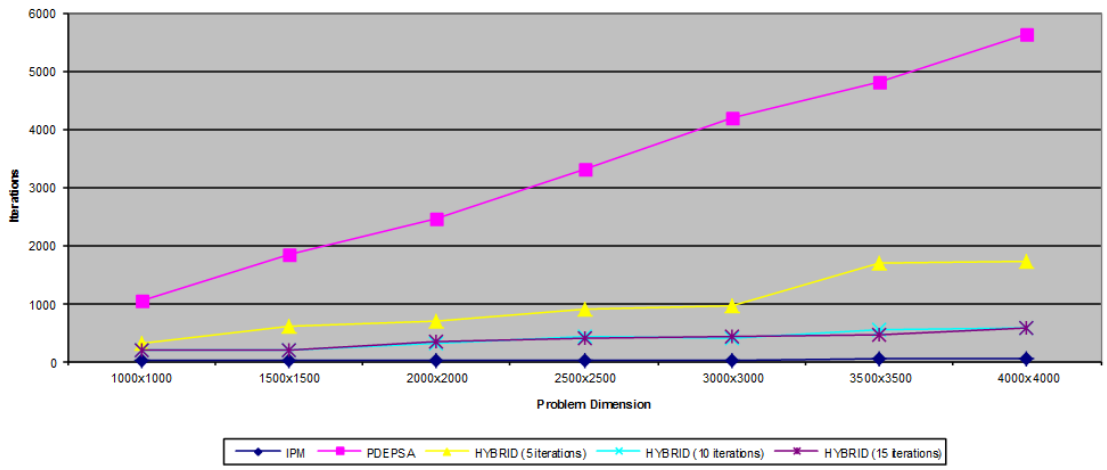

Figure 4. Number of Iterations for Randomly Generated Optimal LPs with 20% Density

We must highlight that the computational results observed for the proposed algorithm are very promising. From the above results, we observe that: (i) the proposed algorithm is on average faster than Matlab's IPM and PDEPSA for the different number of iterations (5, 10 and 15) that IPM was executed, (ii) as expected, being an interior point method, Matlab's IPM needs fewer iterations than the proposed algorithm and PDEPSA, and (iii) the proposed algorithm that begins with the IPM for 10 iterations is on average faster than the implementations that execute IPM for 5 and 15 iterations.

5. CONCLUSIONS

Linear Programming is a significant research area in the field of operations research. The algorithms, which have been proposed for the solution of a Linear Programming problem, belong to three main categories: (i) Simplex-type, (ii) interior-point methods, and (iii) exterior point simplex type algorithms. Interior point methods converge fast to the solution in their first iterations, while in latter iterations the improvement of the objective value is small. On the other hand, exterior point algorithms need more iterations than interior-point methods to solve an LP, but they can converge fast when making steps in directions that are linear combinations of attractive directions, like the interior point that can be calculated by an IPM after a number of iterations.

We proposed an implementation of a hybrid simplex algorithm that begins to solve an LP with an IPM and after a number of iterations continues with a primal-dual EPSA algorithm. This algorithm takes advantage of the aforementioned strengths of interior and exterior point methods. The results from the computational study that was conducted in this paper were very promising. Our proposed algorithm is faster than both interior and exterior point algorithms. Furthermore, we find that the adequate number of iterations that IPM should run in order to decrease the CPU time and the iterations of the proposed algorithm is 10.

In future work, we plan to test our algorithm on benchmark data set and compute more detailed statistics in order to determine the factors that will lead us to determine the exact number of iterations that IPM should run.

6. REFERENCES

[1] Borgwardt, H. K. 1982. The average number of pivot steps required by the simplex method is polynomial. *Zeitschrift fur Operational Research* 26, 1, 157-177. DOI= http://dx.doi.org/10.1007%2FBF01917108.

[2] Bazaraa, M. S., Jarvis, J. J., and Sherali, H. D. 2005. *Linear Programming and Network Flows*, 3rd ed.. John Wiley & Sons. DOI= http://dx.doi.org/10.1002/9780471703778.

[3] Bertsimas, D., and Tsitsiklis, J. N. 1997. *Introduction to Linear Optimization*. Athena Scientific.

[4] Dantzig, G. B. 1963. *Linear Programming and Extensions*. Princeton, University Press, Princeton, NJ.

[5] Andersen, E. D. and Ye, Y. 1996. Combining interior-point and pivoting algorithms for Linear Programming. *Management Science* 42, 12, 1719-1731.

[6] Glavelis, Th. and Samaras, N. 2013. An experimental investigation of a primal-dual exterior point simplex algorithm. *Optimization - A Journal of Mathematical Programming and Operations Research*, 1-10. DOI= http://dx.doi.org/10.1080/02331934.2013.765422

[7] Gondzio, J. 1996. Multiple centrality corrections in a primal-dual method for linear programming. *Computational Optimization and Applications* 6, 2, 137-156. DOI= http://dx.doi.org/10.1007/BF00249643

[8] Karmarkar, N. K. 1984. A new polynomial-time algorithm for linear programming, *Combinatorica*, 4, 4, 373–395. DOI= http://dx.doi.org/10.1007/BF02579150

[9] Klee, V. and Minty, G. J. 1992. *How good is the simplex algorithm?*. Inequalities III. New York: Academic Press, 159–175.

[10] Paparrizos, K. 1991. An infeasible exterior point simplex algorithm for assignment problems. *Mathematical Programming* 51, 1-3, 45–54. DOI= http://dx.doi.org/10.1007/BF01586925

[11] Paparrizos, K., Samaras, N., and Stephanides, G. 2003. A new efficient primal dual simplex algorithm. *Computers and Operations Research* 30, 9, 1383-1399. DOI= http://dx.doi.org/10.1016/S0305-0548(02)00077-1

[12] Paparrizos, K., Samaras, N., and Stephanides, G. 2003. An efficient simplex type algorithm for sparse and dense linear programs. *European Journal of Operational Research* 148, 2, 323-334. DOI= http://dx.doi.org/10.1016/S0377-2217(02)00400-9

[13] Terlaky, T. and Zhang, S. 1993. Pivot rules for linear programming – A survey. *Annals of Operations Research* 46-47, 1, 203-233. DOI= http://dx.doi.org/10.1007/BF02096264

[14] Zhang, Y. 1998. Solving Large-Scale Linear Programs by Interior-Point Methods under the MATLAB Environment. *Optimization Methods and Software* 10, 1, 1-31. DOI= http://dx.doi.org/10.1080/10556789808805699