On Mehrotra-Type Predictor-Corrector Algorithms

M. Salahi*†, J. Peng†, T. Terlaky†

October 10, 2006 (Revised)

Abstract

In this paper we discuss the polynomiality of a feasible version of Mehrotra's predictor-corrector algorithm whose variants have been widely used in several IPM based optimization packages. A numerical example is given that shows that the adaptive choice of centering parameter and correction terms in this algorithm may lead to small steps being taken in order to keep the iterates in a large neighborhood of the central path, which is important to proving polynomial complexity properties of this method.

Motivated by this example, we introduce a safeguard in Mehrtora's algorithm that keeps the iterates in the prescribed neighborhood and allows us to obtain a positive lower bound on the step size. This safeguard strategy is also used when the affine scaling direction performs poorly. We prove that the safeguarded algorithm will terminate after at most $\mathcal{O}(n^2 \log \frac{(x^0)^T s^0}{\epsilon})$ iteration. By modestly modifying the corrector direction, we reduce the iteration complexity to $\mathcal{O}(n \log \frac{(x^0)^T s^0}{\epsilon})$. To ensure fast asymptotic convergence of the algorithm, we changed Mehrotra's updating scheme of the centering parameter slightly while keeping the safeguard. The new algorithms have the same order of iteration complexity as the safeguarded algorithms, but enjoy superlinear convergence as well. Numerical results using the McIPM and LIPSOL software packages are reported.

Keywords: Linear Optimization, Predictor-Corrector Method, Interior-Point-Methods, Mehrotra-Type Algorithm, Polynomial Complexity, Superlinear Convergence.

AMS Subject Classification: 90C05, 90C51

*Department of Mathematics, The University of Guilan, Rasht, Iran, msalahi@optlab.mcmaster.ca

†Advanced Optimization Lab, Department of Computing and Software, McMaster University, Hamilton, Ontario, Canada, L8S 4L7, pengj, terlaky@mcmaster.ca,

1 Introduction

Since Karmarkar's landmark paper [10], the study on Interior Point Methods (IPMs) has become one of the most active research areas in the field of optimization. Many IPMs have been proposed and analyzed [22, 26, 27] and several powerful IPM-based software packages have been developed and successfully applied to numerous applications [1, 5, 29, 30]. Among various variants of IPMs, the so-called predictor-corrector methods have attracted much attention in the IPM community due to its high efficiency and have become the backbones of several optimization packages. It should be mentioned that most implementations of predictor-corrector IPMs adopted a heuristics proposed first by Mehrotra in his remarkable paper [12]. The practical importance of Mehrotra's algorithm motivated us to investigate its theoretical properties. Before going in the details of the algorithm, we briefly review the basics and unique results of IPMs and predictor-corrector IPMs.

We consider primal-dual IPMs for solving the following *Linear Optimization* (LO) problem

$$(P) \qquad \min \{c^T x : Ax = b,\ x \geq 0\},$$

where $A \in R^{m\times n}$ satisfies $\text{rank}(A) = m$, $b \in R^m$, $c \in R^n$, and its dual problem

$$(D) \qquad \max \{b^T y : A^T y + s = c,\ s \geq 0\}.$$

It is common in IPMs theory to assume that both (P) and (D) satisfy the interior point condition (IPC) [22], i.e., there exists an (x^0, y^0, s^0) such that

$$Ax^0 = b,\ x^0 > 0, \quad A^T y^0 + s^0 = c,\ s^0 > 0.$$

Finding optimal solutions of (P) and (D) is equivalent to solving the following system:

$$\begin{aligned} Ax &= b, \quad x \geq 0, \\ A^T y + s &= c, \quad s \geq 0, \\ Xs &= 0, \end{aligned} \tag{1}$$

where $X = \text{diag}\,(x)$. The basic idea of primal-dual IPMs is to replace the third equation in (1) by the parameterized equation $Xs = \mu e$, where e is the all one vector. This leads to the following system:

$$\begin{aligned} Ax &= b, \quad x \geq 0, \\ A^T y + s &= c, \quad s \geq 0, \\ Xs &= \mu e. \end{aligned} \tag{2}$$

If the IPC holds, then for each $\mu > 0$, system (2) has a unique solution. This solution, denoted by $(x(\mu), y(\mu), s(\mu))$, is called the μ-center of the primal-dual pair (P) and (D).

The set of μ-centers with all $\mu > 0$ gives *the central path* of (P) and (D) [11, 24]. It has been shown that the limit of the central path (as μ goes to zero) exists and it is a optimal solution of (P) and (D) [22].

Applying Newton's method to (2) for a given feasible point (x, y, s) gives the following linear system of equations

$$\begin{aligned} A\Delta x &= 0, \\ A^T\Delta y + \Delta s &= 0, \\ x\Delta s + s\Delta x &= \mu e - xs, \end{aligned} \tag{3}$$

where $(\Delta x,\ \Delta y,\ \Delta s)$ give the Newton step.

Predictor-corrector algorithms use (3) with different values of μ in the predictor and corrector steps. The predictor-corrector algorithm with best iteration complexity is the Mizuno-Todd-Ye (MTY) algorithm for LO, operates in two small neighborhoods of the central path [15]. In the predictor step the MTY algorithm uses the so-called primal-dual affine scaling step with $\mu = 0$ in (3) and it moves to a slightly larger neighborhood. Then, in the corrector step, it uses $\mu = \mu_g = \frac{x^T s}{n}$, proportional to the duality gap to bring the iterate towards the central path, back to the smaller neighborhood. In spite of its strong theoretical results for LO and conic linear optimization problems, the algorithm has not been used in developing IPMs based software packages. Several variants of MTY type predictor-corrector algorithms operating in both small and large neighborhoods have been proposed in the past decade [2, 7, 8, 17, 18, 19, 20, 21, 23] and most of them follow a similar theoretical framework as in [15].

In what follows we describe in details a feasible[1] version of Mehrotra's original predictor-corrector algorithm that has been widely used in implementations [1, 12, 30]. In the predictor step Mehrotra's algorithm computes the affine scaling search direction, i.e.,

$$\begin{aligned} A\Delta x^{\mathrm{a}} &= 0, \\ A^T\Delta y^{\mathrm{a}} + \Delta s^{\mathrm{a}} &= 0, \\ s\Delta x^{\mathrm{a}} + x\Delta s^{\mathrm{a}} &= -xs, \end{aligned} \tag{4}$$

then it computes the maximum feasible step size that ensures

$$(x + \alpha_{\mathrm{a}}\Delta x^{\mathrm{a}}, s + \alpha_{\mathrm{a}}\Delta s^{\mathrm{a}}) \geq 0.$$

However, the algorithm does not take such a step right away. It is worth mentioning that Mehrotra's original algorithm allows different step sizes in both primal and dual spaces while here for simplicity of the analysis we consider only the case when they are equal.

[1] The original Mehrotra's algorithm is an infeasible algorithm. However, the self-dual embedding model [22] can be used to construct a slightly bigger LO problem that has an obvious starting point on the central path.

Mehrotra's algorithm then uses the information from the predictor step to compute the corrector direction that is defined as follows:

$$\begin{aligned} A\Delta x &= 0, \\ A^T\Delta y + \Delta s &= 0, \\ s\Delta x + x\Delta s &= \mu e - xs - \Delta x^{\mathrm{a}}\Delta s^{\mathrm{a}}, \end{aligned} \tag{5}$$

where μ is defined adaptively by

$$\mu = \left(\frac{g_{\mathrm{a}}}{g}\right)^2 \frac{g_{\mathrm{a}}}{n},$$

where $g_a = (x + \alpha_{\mathrm{a}}\Delta x^{\mathrm{a}})^T(s + \alpha_{\mathrm{a}}\Delta s^{\mathrm{a}})$ and $g = x^T s$. Since $(\Delta x^{\mathrm{a}})^T\Delta s^{\mathrm{a}} = 0$, the previous relation implies

$$\mu = (1 - \alpha_{\mathrm{a}})^3\mu_g. \tag{6}$$

From (6) it is obvious that if only a small step in the affine scaling direction can be made, then we only improve the centrality of the iterate.

Finally, Mehrotra's algorithm makes a step in the $(\Delta x, \Delta y, \Delta s)$ direction by an appropriate step size and let us denote the new iterate by:

$$x(\alpha) := x + \alpha\Delta x,\ y(\alpha) := y + \alpha\Delta y,\ s(\alpha) := s + \alpha\Delta s.$$

We note that several variants of the previous algorithm have been well-studied in the literature. For example, Mehrotra proposed an infeasible second order predictor corrector IPM [13] based on the power series extension of Monteiro et al. [16]. In his infeasible variant, Mehrotra combined the adaptive scheme with a safeguard technique to stabilize the convergence of the algorithm. Zhang and Zhang [28] have analyzed this second order algorithm without using the adaptive update of the centrality parameter. In [9], the authors suggested to generate several corrector directions first and then use the generated directions to construct a new search direction along which a step can be taken. Gondzio [6] proposed to use multiple centrality steps to bring the iterates back to the vicinity of the central path. Significant improvements have been reported for solving several challenge NETLIB test problems and problems arising from real applications. In a recent work [4], Gondzio further combined the idea of multiple centering with a symmetric neighborhood to avoid potential ill-behaviors of Mehrotra's predictor corrector algorithm. More recently, Mehrotra and Li [14] considered a Krylov subspace based P-C method and established its global convergence. Promising numerical results are reported as well.

Different from the above-mentioned results, in this paper we first explore the potential flaws in the feasible version of Mehrotra's original algorithm. By a numerical example

we show that Mehrotra's algorithm may result in very small steps in order to keep the iterate in a certain neighborhood of the central path, which is essential to prove the polynomiality of the algorithm. To avoid such a trap, we propose to incorporate a safeguard in the algorithm so that we can guarantee a positive lower bound for the step size and subsequently the polynomial complexity. Further, to ensure the superlinear convergence of the algorithm we changed the updating scheme of centering parameter so that the new scheme preserves the same iteration complexity with stronger asymptotic convergence result. It is worthwhile mentioning that our simple safeguard strategy is different from the most recent results by Gondzio et al. [4] and Mehrotra et al. [14], where they employ multiple centering with symmetric neighborhood and Krylov subspace-base corrections, respectively. Most recently, in [3] also the author provided another example that shows that Mehrotra-type predictor-corrector algorithms may fail to converge to an optimal solution. However, the author did not address how to improve Mehrotra's algorithm.

The rest of the paper is organized as follows. First, in Section 2, we present a numerical example that motivates the introduction of a safeguard in Mehrotra's algorithm. Then, in Section 3 we present the safeguard based algorithm and establish its worst case iteration complexity. For readability of the paper we moved some technical lemmas that are used in Section 3 to the Appendix. In Section 4, we further modify the algorithm of Section 3 and discuss its iteration complexity. In Section 5, Mehrotra's updating scheme of the centering parameter is slightly modified to ensure the superlinear convergence of both algorithms in Sections 3 and 4. Some illustrative numerical results using the NETLIB and Kennington test problems are reported in Section 6, and finally we conclude the paper by few remarks in Section 7.

Conventions: Throughout the paper $\|\cdot\|$ denotes the 2-norm of vectors. We denote by $\mathcal{I}$ the index set $\{1, 2, \ldots, n\}$. For any two vectors x and s, xs denotes the componentwise product of the two vectors and e denotes the vector with all components equal to one. For simplicity of notation we remove the iteration index in the coming sections. We also use the notations

$$\mathcal{I}_+ = \{i \in \mathcal{I} \mid \Delta x_i^a \Delta s_i^a > 0\},\ \mathcal{I}_- = \{i \in \mathcal{I} \mid \Delta x_i^a \Delta s_i^a < 0\},$$

$$\mathcal{F} = \{(x, y, s) \in R^n \times R^m \times R^n \mid (x, s) \geq 0,\ Ax = b,\ A^T y + s = c\}$$

and

$$\mathcal{F}^0 = \{(x, y, s) \in R^n \times R^m \times R^n \mid (x, s) > 0,\ Ax = b,\ A^T y + s = c\}.$$

2 Motivation

In this section first we introduce the neighborhood of the central path that the algorithms will operate throughout the paper. Then, we give a numerical example showing that

using the strategy described in the introduction might force the algorithm to make very small steps to keep the iterate in a certain neighborhood of the central path, which further implies the algorithm needs to take many iterations to convergence. The example indicates that Mehrotra's adaptive updating scheme of the centering parameter has to be combined with certain safeguards to get a warranted step size at each iteration.

Most efficient IPM solvers work in the negative infinity norm neighborhood defined by

$$\mathcal{N}_\infty^-(\gamma) := \{(x, y, s) \in \mathcal{F}^0 : \ x_i s_i \geq \gamma \mu_g \ \forall i = 1, \cdots, n\}, \tag{7}$$

where $\gamma \in (0, 1)$ is a constant independent of n and $\mu_g = \frac{x^T s}{n}$. In this paper, we consider algorithms that are working in $\mathcal{N}_\infty^-(\gamma)$ (called large neighborhood).

Let us consider the following simple LO.

$$\begin{aligned} \min \quad & -x_2 \\ \text{s.t.} \quad & 0 \leq x_1 \leq 1, \\ & 0 \leq x_2 \leq 1 + \delta x_1, \end{aligned} \tag{8}$$

where $\delta = 0.1$. Let the algorithm start with from the following feasible points in the neighborhood $\mathcal{N}_\infty^-(0.1)$:

$$x^0 = (0.03; 0.9), \ s^0 = (6.8; 1; 7; 2), \ y^0 = (-7, -2).$$

For the given starting point, if we use identical step sizes for both primal and dual problems, in the third iteration the maximum step size in the predictor step will be $\alpha_{\text{a}} = 0.96$ while the maximum step size in the corrector step is $\mathcal{O}(10^{-4})$ and this value is getting worse for later iterations. To explain what we observed, we examine the constraints

$$x_i(\alpha) s_i(\alpha) \geq \gamma \mu_g(\alpha) \ \forall i \in \mathcal{I}, \tag{9}$$

for $\gamma = 0.1$ that keeps the next iterate in $\mathcal{N}_\infty^-(0.1)$ neighborhood, where

$$\mu_g(\alpha) = \frac{x(\alpha)^T s(\alpha)}{n} = \left(1 - \alpha + \alpha \frac{\mu}{\mu_g}\right) \mu_g. \tag{10}$$

By expanding inequality (9) and reordering one has

$$(1 - \alpha) x_i s_i + \alpha(1 - 0.1)\mu - \alpha \Delta x_i^{\text{a}} \Delta s_i^{\text{a}} + \alpha^2 \Delta x_i \Delta s_i \geq \gamma(1 - \alpha)\mu_g \ \forall i \in \mathcal{I}.$$

Note that for the given staring point, $x_1 s_1 - 0.1\mu_g$ is a very small nonnegative number, while $\Delta x_1^{\text{a}} \Delta s_1^{\text{a}}$ and $\Delta x_1 \Delta s_1$ are both negative numbers whose absolute values are dominated by $x_1 s_1$, and finally $\mu = \mathcal{O}(10^{-5})$ due to a big affine scaling step size. Incorporating all these information into (9) implies that the algorithm requires a very small step to satisfy (9). This phenomenon might be the result of following:

- An aggressive update of centering parameter μ using (6).
- The usage of the correction terms in the corrector system of equations.

To alleviate the abovementioned phenomenon, we propose the following remedies:

- Using a fix fraction of μ_g, for example $\mu = \frac{\mu_g}{10}$ rather than an adaptive update.
- Cutting the maximum step size in the predictor step if it is greater than a certain threshold. This might prevent the algorithm from an aggressive update.
- Modifying the correction terms in the corrector system of equations.

For this specific example, these ideas help us in solving the difficulty that might arise. However, in general modifying the second order correction terms is not as effective as using a simple large update of the centering parameter.

These observations motivate us to introduce a safeguard strategy that will help us to have control on the minimal warranted step size both theoretically and practically. In our safeguard we simply use a fix fraction of μ_g as the μ value. It is worthwhile mentioning that when the affine scaling step size is very small, for example $\alpha_{\mathrm{a}} < 0.1$, which implies marginal reduction of the complementarity gap, we also employ the same large update safeguard.

3 A Safeguard-Based Algorithm

In this section we first discuss the step size estimation of the algorithm and then outline the safeguard based algorithm. Finally, we establish its worst case iteration complexity.

The following technical lemma will be used in the next theorem, which estimates the maximum step size in the corrector step.

Lemma 3.1 *Suppose that the current iterate is* $(x, y, s) \in \mathcal{N}_\infty^-(\gamma)$ *and let* $(\Delta x, \Delta y, \Delta s)$ *be the solution of (5), where* $\mu \geq 0$. *Then we have*

$$\|\Delta x \Delta s\| \leq 2^{\frac{-3}{2}} \left(\frac{1}{\gamma} \left(\frac{\mu}{\mu_g} \right)^2 - \left(2 - \frac{1}{2\gamma} \right) \frac{\mu}{\mu_g} + \frac{17\gamma + n}{16\gamma} \right) n\mu_g.$$

Proof: If we multiply the third equation of (5) by $(XS)^{-\frac{1}{2}}$, then by Lemma 5.3 of [26] we have

$$\|\Delta x \Delta s\| \leq 2^{\frac{-3}{2}} \left\| \mu (XSe)^{-\frac{1}{2}} - (XSe)^{\frac{1}{2}} - (XS)^{-\frac{1}{2}} \Delta x^{\mathrm{a}} \Delta s^{\mathrm{a}} \right\|^2$$

$$= 2^{\frac{-3}{2}} \left(\mu^2 \sum_{i\in\mathcal{I}} \frac{1}{x_i s_i} + \sum_{i\in\mathcal{I}} x_i s_i + \sum_{i\in\mathcal{I}} \frac{(\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}})^2}{x_i s_i} - 2n\mu - 2\mu \sum_{i\in\mathcal{I}} \frac{\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}}}{x_i s_i} \right)$$
$$\leq 2^{\frac{-3}{2}} \left(\frac{n\mu^2}{\gamma\mu_g} + n\mu_g + \frac{n\mu_g}{16} + \frac{n^2\mu_g}{16\gamma} - 2n\mu + \frac{n\mu}{2\gamma} \right),$$

where the last inequality follows from Lemma A.2 and the assumption that the previous iterate is in $\mathcal{N}_\infty^-(\gamma)$. By reordering and factorizing we get the statement of the lemma. □

Motivated from the computational practice, we use $\mu = \frac{\beta}{1-\beta}\mu_g$ as the value of safeguard, where $\gamma \leq \beta < \frac{1}{3}$ rather than using $\mu = \frac{\gamma}{1-\gamma}\mu_g$. Since the later choice for small values of γ might imply an aggressive update of barrier parameter. The following corollary, which follows from Lemma 3.1, gives an explicit upper bound for this specific value of μ.

Corollary 3.2 *If $\mu = \frac{\beta}{1-\beta}\mu_g$, where $\gamma \leq \beta < \frac{1}{3}$ and $\gamma \in (0, \frac{1}{3})$, then*

$$\|\Delta x \Delta s\| \leq \frac{1}{2\gamma} n^2 \mu_g.$$

Theorem 3.3 *Suppose that the current iterate $(x, y, s) \in \mathcal{N}_\infty^-(\gamma)$, where $\gamma \in (0, \frac{1}{3})$ and let $(\Delta x, \Delta y, \Delta s)$ be the solution of (5) with*

$$\mu = \frac{\beta}{1-\beta}\mu_g,$$

where $\gamma \leq \beta < \frac{1}{3}$. Then the maximum step size α_{c}, that keeps $(x(\alpha_{\mathrm{c}}), y(\alpha_{\mathrm{c}}), s(\alpha_{\mathrm{c}}))$ in $\mathcal{N}_\infty^-(\gamma)$, satisfies

$$\alpha_{\mathrm{c}} \geq \frac{3\gamma^2}{2n^2}.$$

Proof: The goal is to find the maximum nonnegative α for which the relation (9) holds. To do so, first let use define

$$t = \max_{i\in\mathcal{I}_+} \left\{ \frac{\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}}}{x_i s_i} \right\}. \tag{11}$$

Since $(\Delta x^a)^T \Delta s^a = 0$, then $\mathcal{I}_+ \neq \emptyset$. Now it is sufficient to prove (9) for $\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}} > 0$. To do so, we have

$$\begin{aligned} x_i(\alpha) s_i(\alpha) &= x_i s_i + \alpha(\mu - x_i s_i - \Delta x^{\mathrm{a}}{}_i \Delta s^{\mathrm{a}}{}_i) + \alpha^2 \Delta x_i \Delta s_i \\ &\geq (1-\alpha) x_i s_i + \alpha\mu - \alpha t x_i s_i - \frac{\alpha^2 n^2 \mu_g}{2\gamma} \\ &= (1 - (1+t)\alpha) x_i s_i + \alpha\mu - \frac{\alpha^2 n^2 \mu_g}{2\gamma}, \end{aligned}$$

where the first inequality follows from $\alpha \geq 0$, (11), and Corollary 3.2. Moreover, from Lemma A.1 we have that $t \leq \frac{1}{4}$, which implies $\frac{1}{1+t} \geq \frac{4}{5}$. Thus we further deduce that for $\alpha \in [0, \frac{4}{5}]$, we have

$$x_i(\alpha)s_i(\alpha) \geq (1-(1+t)\alpha)\gamma\mu_g + \alpha\mu - \frac{\alpha^2 n^2 \mu_g}{2\gamma}.$$

Now using (10), the next iterate belongs to $\mathcal{N}_\infty^-(\gamma)$ provided

$$(1-(1+t)\alpha)\gamma\mu_g + \alpha\mu - \frac{\alpha^2 n^2 \mu_g}{2\gamma} \geq \gamma\left(1-\alpha+\alpha\frac{\mu}{\mu_g}\right)\mu_g,$$

which is equivalent to

$$(1-\gamma)\mu - \gamma t\mu_g \geq \frac{\alpha n^2 \mu_g}{2\gamma}. \tag{12}$$

Using Lemma A.1, and the definition of μ one has

$$(1-\gamma)\mu - \gamma t\mu_g \geq \frac{(1-\gamma)\beta}{(1-\beta)}\mu_g - \frac{\gamma\mu_g}{4} \geq \frac{3\gamma\mu_g}{4}.$$

Therefore, inequality (12) holds if

$$\frac{3\gamma\mu_g}{4} \geq \frac{\alpha n^2 \mu_g}{2\gamma}.$$

This inequality definitely holds for $\alpha = \frac{3\gamma^2}{2n^2}$. Now, we can conclude that

$$\alpha_c \geq \min\left(\frac{4}{5}, \frac{3\gamma^2}{2n^2}\right) = \frac{3\gamma^2}{2n^2}.$$

□

We remind the readers that we use this safeguard when the affine scaling performs poorly, for example $\alpha_{\mathrm{a}} < 0.1$.

Now, after all the previous discussions we may outline our new safeguard based algorithm as follows.

Algorithm 1

Input:

A proximity parameters $\gamma \in (0, \frac{1}{3})$;

a safeguard parameter $\beta \in [\gamma, \frac{1}{3})$;

an accuracy parameter $\epsilon > 0$;

$(x^0, y^0, s^0) \in \mathcal{N}_\infty^-(\gamma)$.

begin

while $x^T s \geq \epsilon$ **do**

begin

Predictor Step

Solve (4) and compute the maximum step size $\alpha_{\rm a}$ such that $(x(\alpha_{\rm a}), y(\alpha_{\rm a}), s(\alpha_{\rm a})) \in \mathcal{F}$;

end

begin

Corrector step

If $\alpha_{\rm a} \geq 0.1$, then solve (5) with $\mu = (1-\alpha_{\rm a})^3 \mu_g$ and compute the maximum step size $\alpha_{\rm c}$ such that $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$;

If $\alpha_{\rm c} < \frac{3\gamma^2}{2n^2}$, then solve (5) with $\mu = \frac{\beta}{1-\beta}\mu_g$ and compute the maximum step size $\alpha_{\rm c}$ such that $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$;

end

else

Solve (5) with $\mu = \frac{\beta}{1-\beta}\mu_g$ and compute the maximum step size $\alpha_{\rm c}$ such that $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$;

end

Set $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) = (x + \alpha_{\rm c}\Delta x, y + \alpha_{\rm c}\Delta y, s + \alpha_{\rm c}\Delta s)$.

end

end

Remark 3.4 *By using an identical step sizes for both the primal and dual problems, comparing with the Mehrotra's algorithm, our new algorithm requires at most an extra backsolve to make a better step.*

The following theorem gives an upper bound for the number of iterations in which Algorithm 1 stops with an $\epsilon-$approximate solution.

Theorem 3.5 *Algorithm 1 stops after at most*

$$O\left(n^2 \log \frac{(x^0)^T s^0}{\epsilon}\right)$$

iterations with a solution for which $x^T s \leq \epsilon$.

Proof: If $\alpha_{\rm a} < 0.1$ or $\alpha_{\rm c} < \frac{3\gamma^2}{2n^2}$, then the algorithm uses the safeguard strategy. It follows from Theorem 3.3 and (10) that

$$\mu_g(\alpha) \leq \left(1 - \frac{3\gamma^2(1-2\beta)}{2(1-\beta)n^2}\right)\mu_g.$$

If $\alpha_{\rm a} \geq 0.1$ and $\alpha_{\rm c} \geq \frac{3\gamma^2}{2n^2}$, then the algorithm uses the Mehrotra's updating strategy, and thus one has

$$\mu_g(\alpha) \leq \left(1 - \frac{2\gamma^2}{5n^2}\right)\mu_g,$$

which completes the proof conforming to Theorem 3.2 of [26]. □

4 A Modified Version of Algorithm 1

In this section we propose a slightly modified version of Algorithm 1 (Algorithm 2) that enjoys much better iteration complexity than Algorithm 1 and computationally also more appealing. The improvement in the iteration complexity is the result of the following lemma and modified corrector step that allow us to strengthen the bound in Lemma 3.1.

Lemma 4.1 *For all* $i \in \mathcal{I}_-$ *one has*

$$-\Delta x_i^{\rm a}\Delta s_i^{\rm a} \leq \frac{1}{\alpha_{\rm a}}\left(\frac{1}{\alpha_{\rm a}} - 1\right) x_i s_i. \tag{13}$$

Proof: For the maximum step size in the predictor step, $\alpha_{\rm a}$, one has

$$x_i(\alpha_{\rm a})s_i(\alpha_{\rm a}) \geq 0,\ \ i = 1, \cdots, n.$$

This is equivalent to

$$(1-\alpha_{\rm a})x_i s_i + \alpha_{\rm a}^2 \Delta x_i^{\rm a}\Delta s_i^{\rm a} \geq 0,\ \ i = 1, \cdots, n,$$

which the statement of the lemma follows. □

The motivation for modifying the Newton system in the corrector step is the following observation. If the affine scaling step size is good, then the classical corrector direction should also be a good choice. However, if the affine scaling step size is not that good, then possibly we should try to bring the iterate back to the vicinity of the central path. In such a case, the second order correction terms in system (5) might not be a good choice since it might lead to a search direction moving towards the boundary of the feasible region.

Therefore, we propose to change the second order correction terms in the corrector step proportional to the affine scaling step size when it does not perform good, for example $\alpha_{\rm a} < 0.1$.

The new corrector system of equations when $\alpha_{\rm a} < 0.1$, which by using Lemma 4.1 enable us to improve on the iteration complexity of Algorithm 1, is

$$\begin{aligned} A\Delta x &= 0, \\ A^T\Delta y + \Delta s &= 0, \\ s\Delta x + x\Delta s &= \mu e - xs - \alpha_{\rm a}\Delta x^{\rm a}\Delta s^{\rm a}, \end{aligned} \tag{14}$$

where the centering parameter μ is defined as in the previous section. Changing corrector system of equation to (14) when $\alpha_{\rm a} < 0.1$ helps us avoid the potential ill behaviors of Mehrotra's original algorithm without sacrificing its practical efficiency (see Section 6). Thus, in the sequel we consider this variant for further analysis.

Analogous to Lemma 3.1 we have the following bound for $\|\Delta x\Delta s\|$ when $\alpha_{\rm a} \in (0, 0.1)$.

Lemma 4.2 *Suppose that the current iterate $(x, y, s) \in \mathcal{N}_\infty^-(\gamma)$, $\alpha_{\rm a} \in (0, 0.1)$ and $(\Delta x, \Delta y, \Delta s)$ be the solution of (14). Then we have*

$$\|\Delta x\Delta s\| \le 2^{\frac{-3}{2}} \left(\frac{1}{\gamma}\left(\frac{\mu}{\mu_g}\right)^2 - \left(2 - \frac{\alpha_{\rm a}}{2\gamma}\right)\frac{\mu}{\mu_g} + \frac{20 - 4\alpha_{\rm a} + \alpha_{\rm a}^2}{16} \right) n\mu_g.$$

Proof: Since $(\Delta x^a)^T\Delta s^a = 0$, both $\mathcal{I}_+$ and $\mathcal{I}_-$ are nonempty. If we multiply the third equation of (14) by $(XS)^{-\frac{1}{2}}$, then by Lemma 5.3 of [26] we have

$$\begin{aligned} \|\Delta x\Delta s\| &\le 2^{\frac{-3}{2}} \left\| \mu(XSe)^{-\frac{1}{2}} - (XSe)^{\frac{1}{2}} - \alpha_{\rm a}(XS)^{-\frac{1}{2}}\Delta x^a\Delta s^a \right\|^2 \\ &= 2^{\frac{-3}{2}} \left(\mu^2 \sum_{i\in\mathcal{I}} \frac{1}{x_i s_i} + x^T s + \alpha_{\rm a}^2 \sum_{i\in\mathcal{I}} \frac{(\Delta x_i^{\rm a}\Delta s_i^{\rm a})^2}{x_i s_i} - 2n\mu - 2\alpha_{\rm a}\mu \sum_{i\in\mathcal{I}} \frac{\Delta x_i^{\rm a}\Delta s_i^{\rm a}}{x_i s_i} \right) \\ &\le 2^{\frac{-3}{2}} \left(\frac{n\mu^2}{\gamma\mu_g} + n\mu_g + \frac{\alpha_{\rm a}^2 n\mu_g}{16} - (1-\alpha_{\rm a}) \sum_{i\in\mathcal{I}_-} \Delta x_i^{\rm a}\Delta s_i^{\rm a} - 2n\mu + \frac{\alpha_{\rm a} n\mu}{2\gamma} \right) \\ &\le 2^{\frac{-3}{2}} \left(\frac{1}{\gamma}\left(\frac{\mu}{\mu_g}\right)^2 + 1 + \frac{\alpha_{\rm a}^2}{16} + \frac{(1-\alpha_{\rm a})}{4} - 2\frac{\mu}{\mu_g} + \frac{\alpha_{\rm a}}{2\gamma}\frac{\mu}{\mu_g} \right) n\mu_g \\ &= 2^{\frac{-3}{2}} \left(\frac{1}{\gamma}\left(\frac{\mu}{\mu_g}\right)^2 - \left(2 - \frac{\alpha_{\rm a}}{2\gamma}\right)\frac{\mu}{\mu_g} + \frac{20 - 4\alpha_{\rm a} + \alpha_{\rm a}^2}{16} \right) n\mu_g, \end{aligned}$$

where the second inequality follows from (13), Lemmas A.1 and A.2, and the assumption that the previous iterate is in $\mathcal{N}_\infty^-(\gamma)$. The third inequality also follows from Lemma A.2. □

The following corollary gives an explicit upper bound for a specific μ.

Corollary 4.3 *Let* $\mu = \frac{\beta}{1-\beta}\mu_g$, *where* $\beta \in [\gamma, \frac{1}{3})$, $\gamma \in (0, \frac{1}{3})$, *and* $\alpha_{\rm a} \in (0, 0.1)$, *then*

$$\|\Delta x \Delta s\| \leq \frac{\beta}{\sqrt{2}\gamma(1-\beta)} n\mu_g.$$

In the following theorem we estimate the maximum step size in the corrector step of the modified algorithm defined by (14) for $\alpha_{\rm a} \in (0, .01)$.

Theorem 4.4 *Suppose that the current iterate* $(x, y, s) \in \mathcal{N}_\infty^-(\gamma)$, *where* $\gamma \in (0, \frac{1}{3})$, $\beta \in [\gamma, \frac{1}{3})$, $\alpha_{\rm a} \in (0, 0.1)$, *and* $(\Delta x, \Delta y, \Delta s)$ *is the solution of (14) with* $\mu = \dfrac{\beta}{1-\beta}\mu_g$. *Then the maximum step size* $\alpha_{\rm c}$, *such that* $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$, *satisfies*

$$\alpha_{\rm c} \geq \frac{39\sqrt{2}\gamma(1-\gamma)}{40n}.$$

Proof: We need to estimate the maximum nonnegative α for which (9) holds. We know that $(\Delta x^{\rm a})^T \Delta s^{\rm a} = 0$, then $\mathcal{I}_+ \neq \emptyset$. It suffices to consider only the case when $\Delta x_i^{\rm a} \Delta s_i^{\rm a} > 0$. Therefore, we have

$$\begin{aligned} x_i(\alpha)s_i(\alpha) &= x_i s_i + \alpha(\mu - x_i s_i - \alpha_{\rm a} \Delta x^{\rm a}{}_i \Delta s^{\rm a}{}_i) + \alpha^2 \Delta x_i \Delta s_i \\ &\geq (1-\alpha) x_i s_i + \alpha\mu - \alpha\alpha_{\rm a} t x_i s_i - \frac{\beta}{\sqrt{2}\gamma(1-\beta)}\alpha^2 n\mu_g \\ &= (1 - \alpha(1+\alpha_{\rm a} t))\, x_i s_i + \alpha\mu - \frac{\beta}{\sqrt{2}\gamma(1-\beta)}\alpha^2 n\mu_g \end{aligned}$$

where the first inequality follows from α being nonnegative, (11), and Corollary 4.3. Moreover, from Lemma A.1 we have that $t \leq \frac{1}{4}$, which implies $\frac{1}{1+\alpha_{\rm a} t} \geq \frac{4}{5}$. Thus we further deduce that for $\alpha \in [0, \frac{4}{5}]$, we have

$$x_i(\alpha)s_i(\alpha) \geq (1 - (1+\alpha_{\rm a} t)\alpha)\gamma\mu_g + \alpha\mu - \frac{\beta}{\sqrt{2}\gamma(1-\beta)}\alpha^2 n\mu_g.$$

By using (10), the new iterate is in $\mathcal{N}_\infty^-(\gamma)$ whenever

$$\gamma\left(1 - \alpha(1+\alpha_{\rm a} t)\right) + \frac{\mu}{\mu_g}\alpha - \frac{\beta}{\sqrt{2}\gamma(1-\beta)}\alpha^2 n \geq \gamma\left(1 - \alpha + \frac{\mu}{\mu_g}\alpha\right).$$

Analogous to Theorem 3.3, one can easily verify that this inequality holds for

$$\alpha = \frac{39\sqrt{2}\gamma(1-\gamma)}{40n}.$$

Therefore, we have

$$\alpha_c \geq \min\left(\frac{4}{5}, \frac{39\sqrt{2}\gamma(1-\gamma)}{40n}\right) = \frac{39\sqrt{2}\gamma(1-\gamma)}{40n} := \hat{\alpha}_c.$$

□

Now, we can outline Algorithm 2 as follows:

Algorithm 2

Input:

A proximity parameters $\gamma \in (0, \frac{1}{3})$;
a safeguard parameter $\beta \in [\gamma, \frac{1}{3})$;
an accuracy parameter $\epsilon > 0$;
$(x^0, y^0, s^0) \in \mathcal{N}_\infty^-(\gamma)$.

begin
while $x^T s \geq \epsilon$ **do**
begin
Predictor Step
Solve (4) and compute the maximum step size $\alpha_{\rm a}$ such that $(x(\alpha_{\rm a}), y(\alpha_{\rm a}), s(\alpha_{\rm a})) \in \mathcal{F}$;
end
begin
Corrector step
If $\alpha_{\rm a} \geq 0.1$, **then** solve (5) with $\mu = (1-\alpha_{\rm a})^3 \mu_g$ and compute the maximum step size $\alpha_{\rm c}$ such that $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$;
If $\alpha_{\rm c} < \hat{\alpha}_c$, **then** solve (5) with $\mu = \frac{\beta}{1-\beta}\mu_g$ and compute the maximum step size $\alpha_{\rm c}$ such that $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$;
end
else
Solve (14) with $\mu = \frac{\beta}{1-\beta}\mu_g$ and compute the maximum step size $\alpha_{\rm c}$ such that $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) \in \mathcal{N}_\infty^-(\gamma)$;
end
Set $(x(\alpha_{\rm c}), y(\alpha_{\rm c}), s(\alpha_{\rm c})) = (x + \alpha_{\rm c}\Delta x, y + \alpha_{\rm c}\Delta y, s + \alpha_{\rm c}\Delta s)$.
end
end

The following theorem gives an upper bound for the number of iterations in which Algorithm 2 stops with an $\epsilon-$ approximate solution.

Theorem 4.5 *Algorithm 2, the modified version of Algorithm 1, stops after at most*

$$\mathcal{O}\left(n \log \frac{(x^0)^T s^0}{\epsilon}\right)$$

iterations with a solution for which $x^T s \leq \epsilon$.

Proof: If $\alpha_{\rm a} < 0.1$ or $\alpha_{\rm c} < \hat{\alpha}_c$, then the algorithm uses the safeguard strategy. Then by (10) and Theorem 4.4 one has

$$\mu_g(\alpha) \leq \left(1 - \frac{39\sqrt{2}\gamma(1-\gamma)(1-2\beta)}{40(1-\beta)n}\right)\mu_g.$$

If $\alpha_{\mathrm{a}} \geq 0.1$ and $\alpha_{\mathrm{c}} \geq \hat{\alpha}_c$, then the algorithm uses the Mehrotra's updating strategy, and thus one has

$$\mu_g(\alpha) \leq \left(1 - \frac{37\gamma(1-\gamma)}{100n}\right)\mu_g,$$

which completes the proof conforming to Theorem 3.2 of [26]. □

5 Superlinear Convergence

In this section we analyze the asymptotic behavior of the previous algorithms using a modification of the centering parameter μ rather than using (6) due to the following observations.

We note that by Theorem 7.4 of [26] for $(x, s) \in \mathcal{N}_\infty^-(\gamma)$ the relations

$$|\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}}| \leq \mathcal{O}(\mu_g^2), \ i = 1, \cdots, n \tag{15}$$

hold. This further implies that $\alpha_{\mathrm{a}} \geq 1 - \mathcal{O}(\mu_g)$. Now, for the asymptotic case, one has to estimate α that satisfies the following inequalities for each $i \in \mathcal{I}$:

$$(1-\alpha)x_i^k s_i^k + (1-\gamma)\alpha\mu - \alpha\Delta x_i^{\mathrm{a}}\Delta s_i^{\mathrm{a}} + \alpha^2 \Delta x_i \Delta s_i \geq \gamma(1-\alpha)\mu_g^k.$$

By using (6) one also has $\mu \leq \mathcal{O}\left((\mu_g^k)^4\right)$. The worst asymptotic value for α might be the result of the case when $x_i s_i = \gamma\mu_g$ and $\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}} > 0$. Assuming this, it is not clear whether $\Delta x_i \Delta s_i$ is nonnegative or negative. In case of nonnegativity the previous inequality holds for a positive value of α, but if $\Delta x_i \Delta s_i < 0$ this inequality might not hold due to the very small μ value which is the result of sufficiently small μ_g. Therefore, modifying Mehrotra's heuristic might be cast as a way to achieve the superlinear convergence. The new adaptive updating strategy is defined by

$$\mu = \frac{\gamma t + \gamma \min(\mu_g^{\frac{1}{2}}, 1)}{1-\gamma}\mu_g, \tag{16}$$

where t is given by (11) and $0 < \gamma < \frac{1}{3}$. The 'γt' term in this definition guarantees the existence of a positive step size following the proof of Theorems 3.3 and 4.4. However, the second term enable us to prove the superlinear convergence as it will be proven in the sequel, since for small μ_g it is not as aggressive as (6). The '$1-\gamma$' term in the denominator of (16) is used for simplicity of the theoretical analysis which follows.

Following the analysis of Sections 3 and 4, changing Mehrotra's updating scheme to this updating strategy in Algorithms 1 and 2, while preserving the large update safeguard, do not change the order of the iteration complexity of them. Since the large update safeguard gives us a positive lower bound for the maximum step size in the corrector step. For simplicity those complexity proofs are omitted here.

Theorem 5.1 *Let the iterate (x^k, y^k, s^k) be generated by Algorithm 1 or 2, where μ is given by (16). When μ_g is sufficiently small, Algorithms 1 and 2 are superlinearly convergent in the sense that $\mu_g^{k+1} = \mathcal{O}\left((\mu_g^k)^{1+r}\right)$, for some $r \in (0,1)$.*

Proof: Since $\left|(\Delta x^{\mathrm{a}})_i^k (\Delta s^{\mathrm{a}})_i^k\right| \leq \mathcal{O}((\mu_g^k)^2)\ \forall i \in \mathcal{I}$, then similar to the proof of Theorem 7.4 of [26] one can show that

$$\left|(\Delta x)_i^k (\Delta s)_i^k\right| \leq \mathcal{O}\left((\mu_g^k)^2\right).$$

By the new definition of μ, the next iterate is in the neighborhood $\mathcal{N}_\infty^-(\gamma)$ if for each $i \in \mathcal{I}$

$$(1-\alpha)x_i^k s_i^k + \alpha(1-\gamma)\mu^k - \alpha \Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}} + \alpha^2 \Delta x_i \Delta s_i \geq \gamma(1-\alpha)\mu_g^k. \tag{17}$$

Our goal is to find $\alpha \in (0,1]$ for which (17) holds. For this, it is sufficient to prove (17) for the case where $(\Delta x^{\mathrm{a}})_i^k (\Delta s^{\mathrm{a}})_i^k > 0$ and $(\Delta x)_i^k (\Delta s)_i^k < 0$. Using the definition of t, for positive component of $\Delta x^a \Delta s^a$ one also has $\Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}} \leq t x_i s_i$. Therefore, it suffices to find $\alpha \in (0,1]$ for which the following inequality holds:

$$(1-\alpha(1+t))x_i^k s_i^k + \alpha(1-\gamma)\mu^k - \alpha^2 \mathcal{O}((\mu_g^k)^2) \geq \gamma(1-\alpha)\mu_g^k. \tag{18}$$

If (18) holds for $\alpha \geq \frac{1}{1+t}$, then $\alpha \geq 1 - \mathcal{O}(\mu_g^k)$, since $\dfrac{1}{1+t} = \dfrac{1}{1+\mathcal{O}(\mu_g^k)} \geq 1 - \mathcal{O}(\mu_g^k)$. Now let us assume that $\alpha < \frac{1}{1+t}$. In order to have (18), using the fact that $x_i^k s_i^k \geq \gamma \mu_g^k$, it suffices to have

$$\begin{aligned}(1-\alpha(1+t))\gamma\mu_g^k + \alpha\gamma t\mu_g^k + \ & \alpha\gamma \min(\mu_g^{\frac{1}{2}}, 1)\mu_g - \alpha^2 \mathcal{O}((\mu_g^k)^2) \\ & \geq \gamma(1-\alpha)\mu_g^k\end{aligned}$$

for some $\alpha \in (0,1]$, which is equivalent to

$$\gamma \mu_g^{\frac{3}{2}} - \alpha \mathcal{O}((\mu_g^k)^2) \geq 0. \tag{19}$$

Inequality (19) definitely holds for $\alpha \geq 1 - \mathcal{O}\left((\mu_g^k)^r\right)$ where $r \in (0,1)$.

Now, by using (10) one further has

$$\mu_g^k(\alpha_{\mathrm{c}}^k) = \left(1 - \alpha_c^k(1 - \mathcal{O}(\mu_g^k))\right)\mu_g^k \leq \left(1 - (1 - \mathcal{O}((\mu_g^k)^r))(1 - \mathcal{O}(\mu_g^k))\right)\mu_g^k \leq \mathcal{O}\left((\mu_g^k)^{1+r}\right).$$

This gives the superlinear convergence of Algorithm 1 with the new choice of the parameter μ. The superlinear convergence of Algorithm 2 also can be proved analogously. □

6 Numerical Results

In this section we report some illustrative numerical results for different variants of the Algorithm 2 presented in Section 4 due to its better computational performance than Algorithm 1 for few problems. The results are obtained by modifying some of the subroutines of the McIPM (a self-dual embedding model based implementation) and the LIPSOL (an infeasible IPMs implementation for LO problems), two Matlab based software packages [29, 30]. Our computational experiments are done on a Pentium 4 machine with 2.53 GHZ and 512 MB ram. Numerical results are reported for all feasible NETLIB and Kennington test problems. For each problem we report the number of iterations, the time it takes to solve the problem, and the number of exact digits, respectively.

For the McIPM package we use the following abbreviations for the different implementations of Mehrotra's algorithm that are follow:

- **PMMcIPM:** Mehrotra's original algorithm presented in Section 1.
- **HMcIPM:** Mehrotra's original algorithm presented in Section 1 combined with heuristics in the definition of the centering parameter. The interested reader can consult the McIPM package for heuristics that are used there [30].
- **NMcIPMI:** Algorithm 2.
- **NMcIPMII:** Algorithm 2 with the new definition of the centering parameter (16) instead of using (6).

For all the abovementioned variants we set $\gamma = 10^{-4}$ and $\mu = \frac{\mu_g}{10}$ as the safeguard. In the implementation of our new definition of μ given by (16), we use $\mu = \frac{1}{5}(t + \min(\mu_g^{\frac{1}{2}}, 1))\mu_g$ rather than using (16) which is introduced for theoretical easiness.

Tables 1 to 3 show that for 36 problems (total number of problems is 112), the number of iterations for PMcIPM are higher than NMcIPMI, for 63 problems are higher than NMcIPMII, and for 65 problems are higher than HMcIPM implementations. As one can notice from Tables 1 to 3, for some problems PMcIPM is doing better than the other implementations and for the rest they all perform equally. Significant difference in time occurs when the number of iterations are significantly different, for example see 'dfl001' and 'degen3' in Table 1 and 'osa-60' and 'pds-20' in Table 3. The comparison between our two new algorithms (NMcIPMI and NMcIPMII) shows that NMcIPMII is better than NMcIPMI on 56 problems and NMcIPMI is doing better only for 22 problems and they perform equally on the rest of the problems. Therefore, in overall NMcIPMII performs better than NMcIPMI. Finally the comparison between NMcIPMII and HMcIPM shows that HMcIPM is doing better on 26 problems while HMcIPM is better on 27 problems

and they perform equally on the rest of the problems. This comparison also shows that our simple safeguard based algorithm is at least as effective as the heuristics used in the package and sometimes over performs HMcIPM on difficult problems as given in Table 3.

The following abbreviations is also used for different implementations of Mehrotra's algorithm in LIPSOL.

- **PLIPSOL:** Infeasible variant of Mehrotra's algorithm presented in Section 1.
- **HLIPSOL:** Infeasible variant of Mehrotra's original algorithm presented in Section 1 with heuristics that are used in the definition of the centering parameter by LIPOSL. One should consult the LIPSOL package for the details of the heuristics [29].
- **SLIPSOL:** Infeasible variant of Algorithm 2.

For all the abovementioned versions of LIPSOL we use $\gamma = 10^{-4}$ and $\frac{\beta}{1-\beta} = 10^{-1}$.

It is worthwhile mentioning that we do not have the second modification of the LIPSOL, namely the new definition of the centering parameter, because it requires a detailed analysis of the infeasible Mehrotra's algorithm that are left for future research.

In Tables 4 to 6 we report the numerical results using the abovementioned variants of LIPSOL. The comparison of iterations numbers show that for 66 problems SLIPSOL and HLIPSOL are doing better than PLIPSOL, while PLIPSOL is better only for few problems. The comparison between SLIPSOL and HLIPSOL shows that in overall they perform equally. Finally, a significant difference in time occurs when the number of iterations dramatically differ, for example see 'dfl001' in Table 4 and 'cre-d', 'osa-14', 'pds-10' and 'pds-20' in Table 6.

7 Final Remarks

In this paper we have discussed the polynomiality of Mehrotra's original predictor-corrector algorithm. By a numerical example we have shown that Mehrotra's algorithm might lead to an inefficient algorithm while keeping the iterate in $\mathcal{N}_\infty^-(\gamma)$ neighborhood which is essential to prove the polynomial iteration complexity. This motivated us to combine his idea with a safeguard strategy that allows us to get a positive lower bound for the step size in the corrector step. Further, by slightly changing the Newton system, the iteration complexity of the algorithm is significantly reduced. This also led us to superior computational performance of the algorithm. To ensure the superlinear convergence of the algorithm we changed Mehrotra's updating scheme of the centering parameter so that the new algorithms, preserve the iterations complexity and exhibit stronger asymptotic

Table 1: Comparison of Iterations Number For the NETLIB Test Problems

Problem	PMMcIPM	NMcIPM I	NMcIPM II	HMcIPM	Problem	PMMcIPM	NMcIPM I	NMcIPM II	HMcIPM
25fv47	(28,4.17,7)	(28,4.07,7)	(28,4.5,7)	(27,4.42,6)	e226	(21,1,7)	(21,0.96,7)	(21,0.98,9)	(21,0.9,9)
80bau3b	(43,18.42,5)	(43,17.56,5)	(41,16.6,5)	(42,18.43,5)	etamacro	(26,1.9,8)	(25,1.7,8)	(24,1.5,7)	(25,1.7,7)
afiro	(10,0.34,8)	(10,0.27,8)	(10,0.31,8)	(10,0.3,9)	ffff800	(28,3,6)	(28,2.7,6)	(27,2.5,6)	(27,3,6)
adlittle	(14,0.33,8)	(14,0.37,8)	(14,0.33,8)	(14,0.42,9)	finnis	(30,2.1,5)	(27,2.1,5)	(28,2.1,5)	(28,2.1,5)
agg	(22,1.87,9)	(22,1.65,9)	(22,1.86,9)	(22,1.83,9)	fit1d	(26,3.13,6)	(24,2.67,6)	(24,3,8)	(26,2.94,8)
agg2	(19,2.31,8)	(19,1.72,8)	(19,1.76,8)	(19,2.11,11)	fit1p	(17,8.5,9)	(17,8.1,9)	(15,7.3,8)	(16,7.3,9)
agg3	(22,2.39,10)	(22,1.71,10)	(20,2.1,9)	(20,2.15,9)	fit2d	(24,23.6,9)	(22, 20.5,8)	(24,21.8,9)	(23,21.3,8)
bandm	(17,0.55,5)	(17,0.55,5)	(18,0.75,8)	(18,0.76,8)	fit2p	(21,23.5,10)	(20,21.8,8)	(21,22.9,8)	(21,23.8,8)
beaconfd	(12,0.61,7)	(12,0.5,7)	(12,0.51,7)	(12,0.71,7)	forplan	(30,1.8,4)	(29.1.1,5)	(30,1.53,4)	(31,1.8,5)
blend	(10,0.37,7)	(10,0.3,7)	(11,0.39,8)	(11,0.39,8)	ganges	(21,3.1,5)	(21,2.5,5)	(20,2.4,5)	(20,2.81,5)
bnl1	(33,2.71,7)	(33,2.6,7)	(33,2.5,7)	(32,2.35,7)	gfrd-pnc	(17,1.5,6)	(17,1.1,6)	(18,1.2,10)	(17,1.33,6)
bnl2	(41,11.86,7)	(41,11.86,7)	(38,10.72,7)	(37,10.91,7)	greenbea	(49,21,2)	(48,20,2)	(47,20.2,2)	(46,20,2)
boeing1	(25,1.94,7)	(25,1.98,7)	(24,1.8,7)	(24,1.94,7)	greenbeb	(48,17.15,4)	(47,17.5,4)	(46,16.2,4)	(48,17,4)
boeing2	(21,0.9,8)	(21,0.8,8)	(20,0.74,6)	(20,0.78,6)	grow15	(18,1.99,9)	(18,1.6,9)	(18,1.56,8)	(17,1.71,7)
bore3d	(21,0.84,9)	(20,0.82,9)	(18,0.74,9)	(17,0.72,7)	grow22	(19,2.5,8)	(19,2.38,8)	(19,2.6,8)	(18,2.35,8)
brandy	(18,1,7)	(18,0.8,7)	(17,0.6,7)	(17,0.63,7)	grow7	(18,1,8)	(18,1,8)	(18,1.2,7)	(17,1.1,7)
capri	(18,1.28,6)	(18,0.92,6)	(19,1.16,6)	(19,1.08,6)	israel	(22,1.9,8)	(22,1.74,6)	(22,1.6,7)	(22,1.88,6)
cycle	(40,12.23,6)	(38,11.72,6)	(39,11.63,6)	(39,11.77,6)	kb2	(18,0.57,10)	(18,0.38,10)	(18,0.49,10)	(17,0.35,8)
czprob	(32,3.71,9)	(32,3.4,8)	(32,3.7,8)	(32,3.75,11)	lotfi	(23,0.93,6)	(23,0.65,6)	(22,0.8,6)	(22,0.74,5)
d2q06c	(46,25.3,7)	(44,24.65,7)	(45,23.54,8)	(45,24.7,8)	maros-r7	(16,91,8)	(16,91,8)	(15,86,9)	(16,91,8)
d6cube	(20,6.35,9)	(20,6.73,10)	(20,6.5,10)	(20,6.1,10)	maros	(31,3.8,5)	(31,3.85,5)	(31,3.5,5)	(32,4.1,5)
degen2	(13,1.2,10)	(13,1.55,9)	(13,1.4,11)	(12,1.2,9)	modszk1	(29,2,4)	(29,2,4)	(29,2.1,6)	(29,2.2,6)
degen3	(43,23,8)	(22,13,8)	(14,9.5,8)	(14,9.5,8)	nesm	(34,6.3,6)	(34,6,6)	(32,5.94,6)	(31,5.8,6)
df1001	(49,607,6)	(47,584,,6)	(45,551,6)	(46,571,6)	perold	(48,5,5)	(48,5.6,6)	(42,4,6)	(42,5.1,5)

Table 2: Comparison of Iterations Number for the NETLIB Test Problems

Problem	PMMcIPM	NMcIPM I	NMcIPM II	HMcIPM	Problem	PMMcIPM	NMcIPM I	NMcIPM II	HMcIPM
pilot	(54,44.52,4)	(65,52.55,4)	(49,39,4)	(48,39.51,4)	sctap3	(14,2.4,9)	(14,2.4,9)	(14,2.4,10)	(14,1.8,10)
pilot4	(39,4.61,6)	(37,3.98,6)	(35,3.46,6)	(37,4,6)	seba	(31,4.36,8)	(31,4.14,9)	(23,3.58,9)	(24,3.63,9)
pilotja	(49,10.25,5)	(44,9.83,5)	(42,8.75,6)	(43,9.25,6)	share1b	(28,0.7,5)	(28,0.7,5)	(27,0.83,5)	(27,0.8,7)
pilotnov	(28,5.9,9)	(28,6.15,10)	(26,5,9)	(26,5.57,9)	share2b	(12,0.45,7)	(11,0.42,7)	(11,0.53)	(11,0.33,6)
pilotwe	(46,5.63,6)	(44,5.29,6)	(42,5.4,6)	(42,5.33,6)	shell	(24,1.7,9)	(23,1.5,8)	(25,2,9)	(24,1.99,9)
pilot87	(77,176,5)	(71,161,5)	(71,158,5)	(79,178,5)	ship04l	(17,1.33,9)	(17,1.34,9)	(16,1.3,9)	(16,1.33,9)
recipe	(12,0.5,9)	(12,0.6,9)	(12,0.5,9)	(12,0.4,9)	ship04s	(17,1,7)	(18,1,7)	(17,0.9,7)	(16,1.1,7)
sc105	(12,0.4,6)	(12,0.4,6)	(11,0.31,6)	(12,0.28,6)	ship08l	(20,2.7,8)	(20,3.2,8)	(19,2.7,10)	(19,2.8,8)
sc205	(12,0.36,6)	(12,0.5,6)	(12,0.31,6)	(12,0.33,6)	ship08s	(19,1.3,8)	(19,1.7,10)	(17,1.34,8)	(17,1.35,8)
sc50a	(11,0.23,7)	(11,0.3,7)	(10,0.27,6)	(11.0.3,7)	ship12l	(28,4.9,8)	(27,3.7,9)	(27,3.86,9)	(27,4.3,8)
sc50b	(10.0.2,8)	(10,0.3,8)	(9,0.24,7)	(10,0.25,8)	ship12s	(23,1.86,9)	(23,1.5,9)	(21,1.82,7)	(21,1.8,7)
scagr25	(17,0.86,9)	(17,0.74,9)	(16,0.83,8)	(15,0.86,7)	sierra	(20,3.6,10)	(19,2.9,9)	(18,2.9,9)	(18,3.16,8)
scagr7	(14,0.43,7)	(14,0.36,7)	(13,0.39,7)	(13,0.44,7)	stair	(17,1.48,6)	(17,1,6)	(18,1.42,7)	(18,1.4,6)
scfxm1	(24,1.07,6)	(23,1.01,7)	(25,1.41,8)	(24,1.27,7)	standata	(17,0.98,10)	(17,0.96,10)	(18,0.86,9)	(18,1.22,9)
scfxm2	(27,1.96,8)	(27,1.91,8)	(26,1.96,8)	(25,2.07,7)	standmps	(20,1.36,10)	(19,1,9)	(20,0.86,10)	(19,1.6,10)
scfxm3	(27,2.85,7)	(26,2.4,7)	(26,2.4,7)	(26,2.75,7)	stocfor1	(14,0.49,8)	(14,0.38,8)	(15,0.56,8)	(14,0.48,7)
scorpion	(14,0.5,8)	(14,0.55,8)	(14,0.6,8)	(14,0.55,8)	stocfor2	(31,4.49,8)	(31,3.58,8)	(32,3.8,8)	(33,4.82,7)
scrs8	(26,1.8,5)	(25,1.83,5)	(24,1.48,5)	(24,1.4,5)	stocfo3	(50,48.7,5)	(50,48.3,5)	(49,47,5)	(51,50.4,5)
scsd1	(11,0.5,9)	(11,0.65,9)	(10,0.43,7)	(10,0.47,7)	truss	(21,5.5,8)	(21,5.94,8)	(20,4.93,8)	(20,5.27,8)
scsd6	(12,0.8,8)	(12,0.85,8)	(13,0.8,8)	(12,0.6,8)	tuff	(20,1.52,6)	(20,1.25,6)	(19,1.13,6)	(19,1.58,7)
scsd8	(10,1.3,10)	(10,1.4,10)	(11,1.3,10)	(10,1.1,10)	vtpbase	(17,0.8,9)	(20,0.7,9)	(17,0.58,8)	(17,0.6,8)
sctap1	(19,0.7,9)	(19,0.92,9)	(19,0.7,10)	(18,0.57,10)	woodlp	(16,5.4,9)	(15,5.3,9)	(15,4.95,9)	(15,5.5,10)
sctap2	(14,1.9,9)	(14,2,9)	(13,1.7,8)	(13,1.4,8)	woodw	(25,7.7,10)	(25,6.5,10)	(25,6.93,9)	(24,7.5,8)

Table 3: Comparison of Iterations Number for the Kennington Test Problems

Problem	MMcIPM	NMcIPM I	NMcIPM II	HMcIPM
cre-a	(28,8.3,8)	(29,8.3,8)	(27,7.1,8)	(29,8.3,8)
cre-b	(37,205,8)	(36,200,8)	(34,188.7,8)	(34,190,8)
cre-c	(31,7.9,8)	(31,7.7,8)	(32,7.5,8)	(32,7.9,8)
cre-d	(35,177.5,8)	(34,173.20,8)	(33,169,8)	(32,163.4,8)
ken-07	(17,4,8)	(17,3.2,8)	(17,4,7)	(17,3.6,7)
ken-11	(21,29.7,7)	(21,29.5,7)	(20,29.6,7)	(20,30,7)
ken-13	(29,91,7)	(28,88,7)	(27,88.5,7)	(26,83,7)
ken-18	(37,637.6,8)	(36,621.6,8)	(34,590.6,8)	(35,621.6,8)
osa-07	(31,31,8)	(31,31,8)	(34,33,8)	(40,39,7)
osa-14	(39,91,8)	(39,87.5,8)	(45,95.2,7)	(52,114,8)
osa-30	(44,,209.5,7)	(41,197.3,7)	(44,206,7)	(44,207,7)
osa-60	(51,612,7)	(48,580,8)	(47,573,8)	(57,668,8)
pds-02	(34,12,7)	(33,11,8)	(32,10.8,88)	(32,11,7)
pds-06	(51,164,7)	(50,160,7)	(43,136.5,7)	(45,146,7)
pds-10	(70,864,7)	(69,854,8)	(56,698,8)	(58,725,8)
pds-20	(96,8164.5,7)	(85,7552.6,7)	(79,6763,7)	(81,6926.4,7)

Table 4: Comparison of Iterations Number for the Kennington Test Problems

Problem	PLIPSOL	SLIPSOL	HLIPSOL	Problem	PLIPSOL	SLIPSOL	HLIPSOL
25fv47	(27,3.1,10)	(24,4,11)	(25,3.7,10)	e226	(23,1.2,7)	(20,0.98,7)	(21,1.2,11)
80bau3b	(46,12.1,4)	(40,10.8,4)	(39,11.3,4)	etamacro	(27,1.8,7)	(25,1.88,7)	(25,1.6,7)
afiro	(8,0.34,10)	(8,0.35,10)	(8,0.2,10)	ffff800	(31,2.95,6)	(26,2.5,6)	(26,2.75,6)
adlittle	(13,0.57,11)	(13,0.57,11)	(13,0.36,11)	finnis	(36,1.8,5)	(28,1.8,5)	(30,1.5,5)
agg	(21,1.4,10)	(21,1.64,10)	(21,1.45,10)	fit1d	(20,1.63,11)	(18,1.74,11)	(19,1.7,11)
agg2	(20,1.9,10)	(19,1.82,10)	(18,2.1,10)	fit1p	(17,12.6,10)	(16,12.3,10)	(16,11.7,10)
agg3	(19,1.86,11)	(18,1.77,11)	(17,1.82,11)	fit2d	(23,11.7,11)	(22,12,11)	(22,11.5,9)
bandm	(18,0.86,9)	(18,0.85,8)	(18,1,10)	fit2p	(21,28.5,9)	(21,30,9),	(21,33,9)
beaconfd	(13,0.7,11)	(13,0.68,11)	(13,0.78,11)	forplan	(25,1.25,6)	(24,1.1,6)	(22,1.2,6)
blend	(12,0.44,10)	(12,0.42,10)	(12,0.5,10)	ganges	(19,2,5)	(18,2,5)	(18,2,5)
bnl1	(33,2.14,7)	(30,2,7)	(26,2,7)	gfrd-pnc	(21,1,11)	(20,0.7,11)	(21,1,11)
bnl2	(38,12.3,9)	(31,10.1,9)	(31,10.4,9)	greenbea	(43,18,3)	(48,19.9,3)	4(43,18.8,2)
boeing1	(24,1.6,10)	(22,1.4,10)	(21,1.6,9)	greenbeb	(42,14.4,4)	(37,12.6,4)	(38,13.6,4)
boeing2	(20,0.83,10)	(17,0.65,8)	(19,1.17,8)	grow15	(18,1.44,11)	(17,1.3,11)	(17,1.4,11)
bore3d	(17,0.83,11)	(18,0.8,10)	(18,1,11)	groww22	(19,2,11)	(18,1.8,11)	(19,1.95,11)
brandy	(18,0.88,10)	(15,0.75,10)	(17,1.2,10)	grow7	(17,0.88,10)	(16,0.86,10)	(16,0.9,10)
capri	(18,1,9)	(18,1,10)	(20,1.2,10)	israel	(25,1.76,11)	(22,1.6,11)	(23,1.3,11)
cycle	(26,8.9,0)	(25,8.4,6)	(24,8.6,6)	kb2	(14,0.45,10)	(14,0.4,10)	(15,0.55,11)
czprob	(37,2.9,11)	(38,3,11)	(36,3,11)	lotfi	(18,0.9,7)	(18,1.1,10)	(18,0.8,10)
d2q06c	(35,25.5,7)	(33,23.9,7)	(32,24,7)	maros-r7	(16,154,8)	(15,151,11)	(15,144,11)
d6cube	(27,8.1,9)	(24,7.5,7)	(23,7.2,9)	maros	(31,7,5)	(33,4.4,11)	(33,4.2,11)
degen3	(26,20.4,9)	(20,16.9,11)	(20,16.4,8)	modszk1	(24,2,9)	(24,1.8,9)	(24,1.8,9)
degen2	(14,1.48,9)	(14,1.46,9)	(14,1.5,9)	nesm	(35,4.9,6)	(33,5,6)	(33,4.7,6)
df1001	(81,1883.33,6)	(59,1452.6,6)	(73,1810,6)	perold	(38,3.85,5)	(33,3.46,5)	(31,2.7,5)

Table 5: Comparison of Iterations Number for the NETLIB Test Problems

Problem	PLIPSOL	SLIPSOL	HLIPSOL	Problem	PLIPSOL	SLIPSOL	HLIPSOL
pilot	(39,43.6,4)	(30,33.6,4)	(31,37,4)	sctap3	(19,1.92,11)	(18,2.3,12)	(18,2.3,12)
pilot4	(31,2.95,8)	(34,3.2,8)	(30,2.65,8)	seba	(20,3.64,12)	(22,4.8,12)	(22,4.3,12)
pilotja	(34,7.4,6)	(32,7,6)	(31,6.7,6)	share1b	(23,0.7,11)	(22,0.83,11)	(22,0.88,11)
pilotnov	(21,4.1,11)	(19,3.94,11)	(20,4.4,11)	share2b	(13,0.47,11)	(12,0.4,11)	(13,0.55,11)
pilotwe	(40,3.5,6)	(35,3,6)	(37,3.8,6)	shell	(23,1.1,11)	(19,0.78,11)	(21,1.27,11)
pilot87	(42,159,6)	(38,151,6)	(38,149,6)	ship04l	(14,0.97,9)	(14,0.9,9)	(14,1,9)
recipe	(9,0.45,11)	(9,0.53,11)	(9,0.58,11)	ship04s	(15,0.86,11)	(14,0.75,11)	(14,0.5,11)
sc105	(10,0.5,11)	(10,0.4,11)	(10,0.4,11)	ship08l	(16,1.9,11)	(16,1.8,11)	(16,1.8,11)
sc205	(11,0.5,11)	(11,0.4,11)	(11,0.4,11)	ship08s	(15,1.14,11)	(15,1.1,11)	(15,1,11)
sc50a	(10,0.33,10)	(10,0.45,10)	(10,0.4,10)	ship12l	(17,1.9,1)	(19,2,11)	(18,1.6,11)
sc50b	(7,0.38,8)	(7,0.33,8)	(7,0.27,8)	ship12s	(18,1,11)	(18,1,11)	(18,0.8,11)
scagr25	(17,0.8,)	(17,0.7,)	(17,0.95,)	sierra	(18,1.95,10)	(18,1.72,10)	(17,2,10)
scagr7	(14,0.52,7)	(13,0.44,7)	(14,0.63,7)	stair	(16,1.4,11)	(14,1.15,10)	(14,1.4,10)
scfxm1	(20,0.92,11)	(19,0.78,10)	(19,1.1,11)	standata	(18,0.92,11)	(16,0.75,11)	(17,1,11)
scfxm2	(22,1.55,11)	(20,1.4,11)	(21,1.7,10)	standmps	(25,1.25,11)	(23,1.2,9)	(24,1.46,11)
scfxm3	(23,2,10)	(20,1.72,10)	(21,2.1,10)	stocfor1	(15,0.53,11)	(17,0.42,11)	(16,0.67,11)
scorpion	(15,0.64,11)	(15,0.65,11)	(15,0.6,11)	stocfor2	(22,2.75,10)	(23,2.85,10)	(21,2.94,10)
scrs8	(25,1.4,5)	(25,1.55,5)	(24,1.35,5)	stocfor3	(33,30.3,5)	(33,30.3,5)	(33,30.3,5)
scsd1	(10,0.52,11)	(10,0.63,7)	(9,0.7,5)	truss	(20,4.1,10)	(18,3.8,10)	(19,4.1,10)
scsd6	(12,0.66,10)	(11,0.78,6)	(12,0.8,10)	tuff	(23,1.44,6)	(20,0.95,6)	(20,1.5,4)
scsd8	(11,0.91,10)	(11,1.1,10)	(11,1.16,10)	vtpbase	(19,0.72,11)	(27,0.81,11)	(23,1,11)
sctap1	(19,0.7,12)	(17,0.8,12)	(17,0.7,12)	woodlp	(24,6.15,10)	(19,5.3,10)	(19,5.3,6)
sctap2	(17,1.55,11)	(16,1.8,10)	(19,1.6,10)	woodw	(30,7.3,5)	(26,6,6)	(28,7.1,8)

Table 6: Comparison of Iterations Number for the Kennington Test Problems

Problem	PLIPSOL	SLIPSOL	HLIPSOL
cre-a	(33,7.18,8)	(29,6.3,8)	(30,6.8,8)
cre-b	(45,352,8)	(37,304.4)	(42,332.4,8)
cre-c	(32,6,8)	(31,6.1,8)	(30,6,8)
cre-d	(47,310.5,8)	(38,264.6,8)	(38,271,8)
ken-07	(16,2.2,8)	(15,2.2,8)	(16,2.2,8)
ken-11	(23,18.8,7)	(21,18.1,7)	(22,18.3,7)
ken-13	(30,60.7,10)	(28,60,10)	(27,55.3,10)
ken-18	(42,650,8)	(42,632,8)	(38,574,8)
osa-07	(29,25.5,7)	(27,25.7,7)	(27,24.4,7)
osa-14	(30,63.5,8)	(34,74,8)	(37,76,8)
osa-30	(37,160.2,8)	(39,183,8)	(36,157,8)
osa-60	(39,461,8)	(38,451,8)	(34,422,8)
pds-02	(29,6.75,7)	(28,7.2,7)	(29,7.1,7)
pds-06	(44,200,7)	(45,211,7)	(42,191,7)
pds-10	(58,1226,8)	(55,1183,8)	(52,1104,8)
pds-20	(69,10975.2,7)	(63,10028,7)	(67,10645,7)

convergence properties. Our illustrative numerical results show that our new safeguard based algorithms are competitive with the two state of the art software packages that are using heuristics to stabilize the convergence of the implemented algorithms.

There are several interesting questions regarding the proposed safeguard strategy. For example, one can analyze the infeasible variant of this prototype. It is also possible to extend this approach to other classes of optimization problems, such as SDO, SOCO and convex nonlinear optimization.

Acknowledgments: The research of the first and last authors is supported by the NSERC Discovery Grant DG:5-48923, the Canada Research Chair program, and by MITACS. The research of the second author is supported by the NSERC Discovery Grant DG:249635-02, a PREA award, and by MITACS. Finally, the authors thank the associate editor and two anonymous referees for their valuable comments on the earlier versions of this paper.

References

[1] E.D. Andersen. and K. D. Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, High Performance Optimization, Kluwer Academic Publishers, pp. 197–232, 2000.

[2] K.M. Anstreicher and R.A. Bosch. A new infinity-norm path following algorithm for linear programming. *SIAM Journal on Optimization*, 5(2), pp. 236–246, 1995.

[3] C. Cartis. Some disadvantages of a Mehrotra-tpye primal-dual corrector interior point algorithm for linear programming. `http://www.optimization-online.org`, 2005.

[4] M. Colombo and J. Gondzio. Further Development of Multiple Centrality Correctors for Interior Point Methods. `http://www.optimization-online.org`, 2005.

[5] J. Czyzyk, S. Mehrtotra, M. Wagner, and S.J. Wright. PCx: An interior-point code for linear programming. *Optimization Methods and Software*, 11/12, pp. 397–430, 1999.

[6] J. Gondzio. Multiple centrality corrections in a primal-dual method for linear programming. *Computational Optimization and Applications,* 6, pp. 137–156, 1996.

[7] C.C. Gonzaga. Complexity of predictor-corrector algorithms for LCP based on a large neighborhood of the central path. *SIAM Journal on Optimization,* 10, pp. 183–194, 1999.

[8] P. Hung and Y. Ye. An asymptotically $O(\sqrt{n}L)$-iteration path-following linear programming algorithm that uses long steps. *SIAM Journal on Optimization*, 6, pp. 570–586, 1996.

[9] F. Jarre and M. Wechs. Extending Mehrotra's corrector for linear programs. *Advanced Modeling and Optimization*, 1(2), pp. 38–60, 1999.

[10] N.K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4, pp. 373–395, 1984.

[11] N. Megiddo. Pathways to the optimal set in linear programming. In: N. Megiddo, editor, *Progress in Mathematical Programming: Interior Point and Related Methods*, pp. 131–158. Springer Verlag, New York. Identical version in: *Proceedings of the 6th Mathematical Programming Symposium of Japan, Nagoya, Japan*, 1986, pp. 1–35.

[12] S. Mehrotra. On finding a vertex solution using inteior-point methods. *Linear Algebra and its Applications*, 152, pp. 233–253, 1991.

[13] S. Mehrotra. On the implementation of a (primal-dual) interior point method. *SIAM Journal on Optimization*, 2, pp. 575–601, 1992.

[14] S. Mehrotra and Z. Li. Convergence conditions and krylov subspace-based corrections for primal-dual interior point method. *SIAM Journal on Optimization*, 15, pp. 635–653, 2005

[15] S. Mizuno. M.J. Todd, and Y. Ye. On adaptive step primal-dual interior-point algorithms for linear programming. *Mathematics of Operations Research*, 18, pp. 964–981, 1993.

[16] R.D.C. Monteiro, I. Adler, and M.G.C. Resende. A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extensions. *Mathematics of Operations Research*, **15**, pp. 191–214, 1990.

[17] J. Peng, T. Terlaky, and Y.B. Zhao. A predictor-corrector algorithm for linear optimization based on a specific self-regular proximity function. *SIAM Journal on Optimization,* 15(4), pp. 1105–1127, 2005.

[18] J. Ji, F.A. Potra, and S. Huang. A predictor-corrector method for linear complementarity problems with polynomial complexity and superlinear convergence. *Journal of Optimization Theory and Applications,* 84(1), pp. 187–199, 1995.

[19] F.A. Potra. The Mizuno-Todd-Ye algorithm in a large neiborhood of the central path, *European Journal of Operation Researh,* 143, pp. 257–267, 2002.

[20] F.A. Potra. A superlinearly convergent predictor-corrector method for degenrate LCP in a wide neighborhood of the central path, *Mathematical Programming.* 100(2), pp. 317–337, 2004.

[21] F.A. Potra and X. Liu. Predictor-corrector methods for sufficient linear complementarity problems in a wide neighborhood of the central path. *Technical Report, Department of Mathematics and Statistics, University of Maryland, USA,* 2003.

[22] C. Roos, T. Terlaky, and J.-Ph.Vial. *Theory and Algorithms for Linear Optimization: An Interior Point Approach.* John Wiley & Sons, Chichester, UK, 1997.

[23] M. Salahi and T. Terlaky. Adaptive large neighborhood self-regular predictor-corrector IPMs for LO. *Technical Report* 2004/7*, Advanced Optimization Lab. Department of Computing and Software, McMaster University,* `http://www.cas.mcmaster.ca/~oplab/publication.` To appear in *Journal of Optimization Theory and Applications.*

[24] G. Sonnevend. An "analytic center" for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming. In: A. Prékopa, J. Szelezsán, and B. Strazicky, editors, *System Modeling and Optimization: Proceeding of the 12th IFIP Conference*, Budapest, Hungary, September 1985, Volume 84, Lecture Notes in Control and Information Sciences, pp. 866–876. Springer Verlag, Berlin, 1986.

[25] R.A. Tapia, Y. Zhang, M.J. Saltzman, and A Weiser.The Mehrotra predictor-corrector interior-point method as a perturbed composite Newton method. *SIAM Journal on Optimization*, 6(1), pp. 47–56, 1996.

[26] S.J. Wright. *Primal-Dual Interior-Point Methods.* SIAM, Philadelphia, USA, 1997.

[27] Y. Ye. *Interior Point Algorithms, Theory and Analysis.* John Wiley & Sons, Chichester, UK, 1997.

[28] Y. Zhang and D. Zhang. On the polynomiality of the Mehrotra-type predictor-corrector interior point algorithms. *Mathematical Programming*, 68, 303–317, 1995.

[29] Y. Zhang . Solving large-scale linear programs by inteior point methods under the Matlab enviroment. *Optimization Methods and Software,* 10, pp.1–31, 1999.

[30] X. Zhu, J. Peng, T. Terlaky, and G. Zhang. On implementing self-regular proximity based feasible IPMs. *Technical Report* 2003/2*, Advanced Optimization Lab. Department of Computing and Software, McMaster University,* `http://www.cas.mcmaster.ca/~oplab/publication.`

A Appendix

In this section we prove two technical lemmas that have been used frequently during the analysis.

Lemma A.1 *Let $(\Delta x^a, \Delta y^a, \Delta s^a)$ be the solution of (4). Then*

$$\Delta x_i^a \Delta s_i^a \leq \frac{x_i s_i}{4}, \ \forall i \in \mathcal{I}_+.$$

Proof: By equation (4) for $i \in \mathcal{I}_+$ we have

$$s_i \Delta x_i^a + x_i \Delta s_i^a = -x_i s_i.$$

If we divide this equation by $x_i s_i$ we get

$$\frac{\Delta x_i^a}{x_i} + \frac{\Delta s_i^a}{s_i} = -1.$$

Since $\Delta x_i^a \Delta s_i^a > 0$, this equality implies that both $\Delta x_i^a < 0$ and $\Delta s_i^a < 0$. Then, from

$$0 \leq \left(\frac{\Delta x_i^a}{x_i} - \frac{\Delta s_i^a}{s_i} \right)^2 = \left(\frac{\Delta x_i^a}{x_i} \right)^2 + \left(\frac{\Delta s_i^a}{s_i} \right)^2 - 2\frac{\Delta x_i^a \Delta s_i^a}{x_i s_i} = 1 - 4\frac{\Delta x_i^a \Delta s_i^a}{x_i s_i}$$

we get

$$\Delta x_i^a \Delta s_i^a \leq \frac{x_i s_i}{4}.$$

□

Lemma A.2 *Let $(\Delta x^a, \Delta y^a, \Delta s^a)$ be the solution of (4), then we have*

$$\sum_{i \in \mathcal{I}_+} \Delta x_i^a \Delta s_i^a = \sum_{i \in \mathcal{I}_-} |\Delta x_i^a \Delta s_i^a| \leq \frac{1}{4} \sum_{i \in \mathcal{I}_+} x_i s_i \leq \frac{x^T s}{4}.$$

Proof: Since $(\Delta x^a)^T \Delta s^a = 0$, the proof is a direct consequence of Lemma A.1. □