On the Smoothed Price of Anarchy of the Traffic Assignment Problem

Luciana Buriol[1], Marcus Ritt[1], Félix Rodrigues[1], and Guido Schäfer[2]

1 **Universidade Federal do Rio Grande do Sul**
Informatics Institute, Theoretical Computer Science Department
`{buriol,mrpritt,fcrodrigues}@inf.ufrgs.br`

2 **CWI Amsterdam**
Algorithms, Combinatorics and Optimization Group
`g.schaefer@cwi.nl`

Abstract

We study the effect of perturbations on the Price of Anarchy for the Traffic Assignment Problem. Adopting the smoothed analysis approach, we randomly perturb the latency functions of the given network and estimate the expected Price of Anarchy on the perturbed instances. We provide both theoretical and experimental results that show that the Smoothed Price of Anarchy is of the same order of magnitude as the original one.

1998 ACM Subject Classification G.2.2 Network Problems

Keywords and phrases Traffic Assignment Problem, Smooth Analysis, Price of Anarchy

1 Introduction

The Traffic Assignment Problem [5, 12] models applications in which traffic participants (also called *users*) choose routes in a given road network so as to minimize their individual travel times. Each arc of the network has an associated latency function that expresses the flow-dependent delay that users experience if they travel along that arc. The goal of every user is to choose a path from his origin to his destination such that the total delay to travel along this route is minimized. Because the delay of each user also depends on the choices made by the others, this problem can also naturally be interpreted as a strategic game in which players (users) compete for resources (roads) and every player acts selfishly in the sense that he attempts to choose a route of minimum delay.

Wardrop's first principle of equilibrium [12] states that each user seeks non-cooperatively to minimize his own travel time. A route assignment satisfying this first principle is also called a *Wardrop equilibrium* or *user equilibrium* (see Section 2 for formal definitions). Said differently, in a Wardrop equilibrium no user has an incentive to switch to another path because he travels along a shortest latency path from his origin to his destination. *Wardrop's second principle of equilibrium* [12] states that all users cooperatively choose their routes in order to minimize the average travel time of all users. A route assignment satisfying this second principle is called a *system optimum.* That is, a system optimum corresponds to the best possible route assignment that one could enforce if a global authority were able to control all users, regardless of their interests. It is a well-known fact that selfish route choices may lead to suboptimal outcomes (see, e.g., [4]).

In recent years, the study of the inefficiency of equilibria has received a lot attention. The *Price of Anarchy (PoA)* [8] is an inefficiency measure that refers to the maximum ratio (over all possible input instances) of the cost of a worst possible equilibrium outcome and the cost of an optimal outcome. In a seminal work, Roughgarden [9] analyzed the Price of Anarchy of the Traffic Assignment Problem and revealed that it is independent of the network topology and only depends on the type of latency functions. For example, for polynomial latency functions of degree at most p the Price of Anarchy grows like $\Theta(\frac{p}{\ln p})$. Researchers have investigated several "mechanisms" to reduce the Price of Anarchy for the Traffic Assignment Problem. One such example is the use of road tolls (see, e.g., [2, 7]).

In this paper, we start the investigation of the *Smoothed Price of Anarchy* of the Traffic Assignment Problem. Our motivation originates from the observation that in practical applications delays are hardly ever exact but usually subject to (small) fluctuations. Such fluctuations might be caused by various reasons such as roadworks, accidents, weather conditions, varying driver behavior, etc. In our studies we adopt the *smoothed analysis* approach introduced by Spielman and Teng [11]. The idea is to perturb each input instance by adding some random noise to the latency functions and to study the Price of Anarchy on the perturbed instances. The hope is that the Smoothed Price of Anarchy of the Traffic Assignment improves quickly as the magnitude of random perturbation increases. Such a result would provide some evidence that the worst-case point of view adopted in the studies of the Price of Anarchy is overly pessimistic in the context of the Traffic Assignment Problem. In a way, it suggests that the (high) Price of Anarchy is due to artificial worst-case instances that hardly occur in practice.

We propose a simple smoothing model in which the latency function of every arc is perturbed by a factor $(1+\varepsilon)$, where ε is chosen uniformly at random out of the range $[0,\sigma]$ (see Section 2.3 for details). We provide both theoretical and experimental results that show that the Price of Anarchy is rather invariant under these random perturbations. For Pigou instances with polynomial latency functions we derive a closed-form expression for the Smoothed Price of Anarchy (see Section 3). Even for perturbations in the order of the maximum degree of the polynomial, the Smoothed Price of Anarchy does not differ significantly from the Price of Anarchy. We observe a similar effect in our experiments. We consider some real-world instances from the Transportation Network Test Problems [1] incorporating latency functions as suggested by the U.S. Bureau of Public Roads [3] (see Section 4). Our experiments suggest that random perturbations only have a moderate effect on the Price of Anarchy.

2 Preliminaries

2.1 Traffic Assignment Problem

The *Traffic Assignment Problem (TAP)* that we consider in this paper in defined as follows. We assume that the road network is given by a directed multigraph $G=(V,E)$, where V is the set of vertices and E is the set of arcs. The users are modeled by a set of commodities K with each commodity $i\in K$ having an associated vertex pair $(s_i,t_i)\in V\times V$. Users that have the same origin-destination pair (s_i,t_i) are said to belong to the same commodity i. For each commodity $i\in K$ we are given a demand d_i which specifies the total flow (corresponding to the users of commodity i) that has to be sent from s_i to t_i.

The set of paths from s_i to t_i is denoted as $\mathcal{P}_i$. Let $\mathcal{P}=\cup_{i\in K}\mathcal{P}_i$. A flow f specifies for each path $P\in\mathcal{P}$ a non-negative flow value that is sent along P, i.e., f is a function $f:\mathcal{P}\to\mathbb{R}^+$. The flow on arc $e\in E$ is defined as $f_e=\sum_{P:e\in P}f_P$, where $P\in\mathcal{P}$. A flow f is *feasible* if it satisfies the demand for every commodity, i.e., $\sum_{P\in\mathcal{P}_i}f_P=d_i$ for every $i\in K$.

For each arc $e\in E$ we are given a latency function $l_e:\mathbb{R}^+\to\mathbb{R}^+$ which maps the flow f_e of an edge e to the traversal time $l_e(f_e)$. The latency of a path $P\in\mathcal{P}$ is defined as the sum of the edge latencies in the path, i.e., $l_P = \sum_{e\in P} l_e(f_e)$. Subsequently, we use (G, d, l) to refer to an instance of the Traffic Assignment Problem.

We assume that all latency functions are nonnegative, differentiable and nondecreasing. For real-world instances, the most common type of latency functions originates form the U.S. Bureau of Public Roads [3], which can be expressed as

$$l_e(f_e) = t_e\left(1 + \alpha\left(\frac{f_e}{c_e}\right)^{\beta}\right). \tag{1}$$

Here t_e is the free-flow travel time of edge e, i.e., the time it takes to travel through road e if there is no congestion. The constant c_e stands for the capacity of edge e and α and β are tuning parameters, usually set to 0.15 and 4, respectively (all variables are greater than zero).

In order to evaluate the total travel time of the network, we define a cost function $c(f) = \sum_{e\in E} l_e(f_e)f_e$. The *system optimum* refers to a feasible flow that minimizes this cost function. Computing an optimal flow can be described by the following program:

$$\begin{aligned} \underset{f}{\text{minimize}} \quad & c(f) = \sum_{e\in E} l_e(f_e)f_e \\ \text{subject to} \quad & \sum_{P\in\mathcal{P}_i} f_P = d_i \qquad \forall i \in K \\ & \sum_{P\in\mathcal{P}:e\in P} f_P = f_e \qquad \forall e \in E \\ & f_P \geq 0 \qquad \forall P \in \mathcal{P}. \end{aligned} \tag{2}$$

A feasible flow f is a *Wardrop flow* (or *user equilibrium*) if the flow of every commodity i travels along a minimum latency path available. That is, for every commodity i all flow-carrying paths have the same latency and all other paths have no smaller latency. More formally, a flow f is a Wardrop flow if

$$\forall i \in K, \;\; \forall P_1, P_2 \in \mathcal{P}_i, \;\; f_{P_1} > 0: \quad l_{P_1}(f) \leq l_{P_2}(f). \tag{3}$$

An optimal flow corresponds to a Wardrop flow with respect to marginal cost functions. In order for this equivalence to hold we further need to assume that all latency functions are *standard* [9], i.e., $x \cdot l(x)$ is convex. The *marginal cost function* of edge e is defined as $l_e^*(x) = l_e(x) + x\frac{d}{dx}(l_e(x))$. Now, a feasible flow f^* is an optimal flow for (G, d, l) if and only if it is a Wardrop flow for the instance (G, d, l^*) (see [9] for details).

The problem of computing a Wardrop flow can be described by the following program:

$$\begin{aligned} \underset{f}{\text{minimize}} \quad & \sum_{e\in E} \int_0^{f_e} l_e(x)\, dx \\ \text{subject to} \quad & \sum_{P\in\mathcal{P}_i} f_P = d_i \qquad \forall i \in K \\ & \sum_{P\in\mathcal{P}:e\in P} f_P = f_e \qquad \forall e \in E \\ & f_P \geq 0 \qquad \forall P \in \mathcal{P}. \end{aligned} \tag{4}$$

We note that the cost $c(f)$ of a Wardrop flow f is unique (see [9]).

In this form, computing a Wardrop flow as well as an optimal flow can be done by using the Frank-Wolfe algorithm [6]. The algorithm starts by finding a feasible solution to the linear constraints of the problem. Then in each iteration it finds a descent direction and a distance to descend, thereby reducing the objective function value. The algorithm stops when no improvement can be made to the objective function value.

2.2 Price of Anarchy

The *Price of Anarchy (PoA)* is a measure of the inefficiency of equilibria that was introduced by Koutsoupias and Papadimitriou [8]. It measures how well players in a game perform when they are at a Nash equilibrium, compared to an optimum outcome that could be achieved if all players cooperated.

Suppose we are given a strategic game G with n players, a set of strategies S_i for each player i and a cost function $c_i : S \to \mathbb{R}$, where $S = S_1 \times \cdots \times S_n$. Further assume that we are given a social cost function $C : S \to \mathbb{R}$ that maps every strategy profile $s \in S$ to some non-negative cost of the game. Given an instance $I = (G, (S_i), (c_i))$, let $\mathrm{NE}(I)$ be the set of strategy profiles $s \in S$ that are a Nash equilibrium for I. The Price of Anarchy of I is defined as

$$\mathrm{PoA}(I) = \frac{\max_{s \in \mathrm{NE}(I)} C(s)}{\min_{s \in S} C(s)}$$

The Price of Anarchy of a class of games $\mathcal{G}$ is defined as $\mathrm{PoA}(\mathcal{G}) = \max_{I \in \mathcal{G}} \mathrm{PoA}(I)$.

In the context of the Traffic Assignment Problem (TAP), the above definition simplifies to the following: Let $I = (G, d, l)$ be an instance of TAP. The Price of Anarchy of I is $\mathrm{PoA}(I) = c(f)/c(f^*)$, where f and f^* are a Wardrop flow and an optimal flow of I, respectively. (Recall that the cost of a Wardrop flow is unique.) The Price of Anarchy of TAP is defined as $\mathrm{PoA} = \max_I \mathrm{PoA}(I)$, where the maximum is taken over all possible input instances.

The Price of Anarchy depends on which types of latency functions we allow our instances to have. Roughgarden and Tardos [10] proved that for linear latencies the Price of Anarchy is $\frac{4}{3}$. Furthermore, Roughgarden proved that the Price of Anarchy is independent of network topology [9]. Besides other results, these studies reveal that the Price of Anarchy for polynomial latency functions is admitted on very simple single-commodity instances consisting of two parallel arcs, also known as *Pigou instances.*

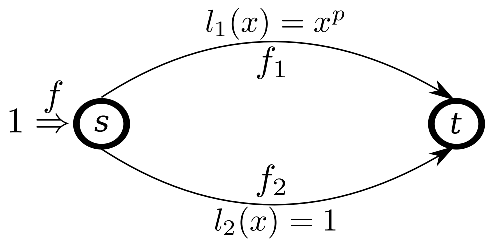

Figure 1 Pigou instance with polynomial latency functions.

Consider the instance $I = (G, d, l)$ depicted in Figure 1. There is one unit of flow that has to be sent from s to t. The respective latency functions of the upper edge e_1 and the lower edge e_2 are $l_1(x) = x^p$ and $l_2(x) = 1$. The Wardrop flow f sends the entire flow on e_1, i.e., $f_1 = 1$ and $f_2 = 0$, and has a cost $c(f) = 1$. In order to compute an optimal flow, we exploit the equivalence that an optimal flow is a Wardrop flow with respect to marginal cost functions $l_1^*(x) = (p+1)x^p$ and $l_2^*(x) = 1$. Equalizing these latency functions, we obtain that

$f_1^* = (p+1)^{-1/p}$ and $f_2^* = 1-(p+1)^{-1/p}$. The cost of this flow is

$$c(f^*) = (p+1)^{-1/p}\left((p+1)^{-1/p}\right)^p + 1 - (p+1)^{-1/p} = \frac{(p+1)(p+1)^{1/p} - p}{(p+1)(p+1)^{1/p}}.$$

The Price of Anarchy of this instance I is therefore

$$\mathrm{PoA}(I) = \frac{c(f)}{c(f^*)} = \frac{(p+1)\sqrt[p]{p+1}}{(p+1)\sqrt[p]{p+1} - p}. \tag{5}$$

This is actually the worst possible, and is $\Theta(\frac{p}{\ln p})$. Roughgarden [9] showed that the Price of Anarchy of multi-commodity instances with polynomial latency functions of degree at most p is at most $\mathrm{PoA}(I)$ as stated in (5).

2.3 Smoothed Price of Anarchy

Smoothed analysis was introduced by Spielman and Teng [11] in order to overcome the pessimistic viewpoint adopted in worst-case analyses. It is a relatively new approach that can be seen as a hybrid of worst-case and average-case analysis. It was originally introduced to study the smoothed complexity of algorithms. But the concept naturally extends to other performance criteria.

We extend the idea to the Price of Anarchy measure. The idea is to randomly perturb a given input instance and to analyze the expected Price of Anarchy on the perturbed instances. Suppose we are given a class of games $\mathcal{G}$. Given an instance $I = (G, (S_i), (c_i)) \in \mathcal{G}$, we randomly perturb I by adding some random noise to the input data. (Note that there might be several ways to perturb the input instance. How this should be done depends on the underlying application.) Let $\bar{I}$ be an instance that can be obtained from I by random perturbations and let σ be a parameter for the magnitude of perturbation. The *Smoothed Price of Anarchy (SPoA)* of I is then defined as

$$\mathrm{SPoA}(I,\sigma) = \frac{\mathbf{E}[\max_{s\in \mathrm{NE}(\bar{I})} \bar{C}(s)]}{\mathbf{E}[\min_{s\in S} \bar{C}(s)]},$$

where the expectation is taken over all instances $\bar{I}$ that are obtainable from I by random perturbations of magnitude σ. Here $\bar{C}$ refers to the cost of the perturbed instance. The Smoothed Price of Anarchy of $\mathcal{G}$ is then $\mathrm{SPoA}(\mathcal{G},\sigma) = \max_{I\in\mathcal{G}} \mathrm{SPoA}(I,\sigma)$.

Perturbation Model for TAP Instances: In our context, we perturb TAP instances by adding some random noise to the latency functions. Our perturbations thus reflect fluctuations in the travel times of the edges. More specifically, suppose we are given an instance $I = (G, d, l)$ of TAP. We then define *perturbed latency functions* $\bar{l}$ as follows:

$$\forall e \in E: \quad \bar{l}_e = (1+\varepsilon_e) l_e, \quad \varepsilon_e \xleftarrow{\text{i.u.r}} [0,\sigma]. \tag{6}$$

Note that ε_e is chosen independently uniformly at random out of the range $[0,\sigma]$ for every edge $e \in E$. Let f_I and f_I^* denote a Wardrop flow and an optimal flow, respectively, for a given instance I. The Smoothed Price of Anarchy of I is then defined as

$$\mathrm{SPoA}(I,\sigma) = \frac{\mathbf{E}[\bar{c}(f_{\bar{I}})]}{\mathbf{E}[\bar{c}(f_{\bar{I}}^*]},$$

where $\bar{c}$ refers to the total cost with respect to the perturbed latency functions, i.e., $\bar{c}(f) = \sum_{e \in E} \bar{l}_e(f_e) f_e$ for a given flow f. As before, the expectation is taken over all instances $\bar{I}$ that are obtainable from I by perturbations as defined in (6). The Smoothed Price of Anarchy of $\mathcal{G}$ is defined as $\mathrm{SPoA}(\mathcal{G}, \sigma) = \max_{I \in \mathcal{G}} \mathrm{SPoA}(I, \sigma)$.

Clearly, other smoothing models are conceivable as well. However, here we have chosen the one above because of its good trade-off between simplicity and relevance. Note that a consequence of our relative perturbation model is that the effect of random perturbations is more severe on edges that are sensitive to variations in traffic rate while it is less severe on edges which are rather insensitive to changes in traffic rate.

Note that for our real-world instances, whose latency functions are of the form indicated in (1), the above perturbation is equivalent to substituting the free-flow travel time t_e with $(1 + \varepsilon) t_e$.

3 Smoothed PoA of Pigou Instances

We consider Pigou instances with polynomial latency functions. We will derive exact bounds on the Smoothed Price of Anarchy under our random perturbations for these instances. These bounds also establish a lower bound on the Smoothed Price of Anarchy for general multi-commodity instances. We leave it as an important open problem to derive bounds on the Smoothed Price of Anarchy for multi-commodity instances and polynomial latency functions.

▶ **Theorem 1.** *The Smoothed Price of Anarchy of the Pigou instance with polynomial latency functions of degree p is*

$$SPoA(I, \sigma) = \frac{3 + \sigma}{3 + \frac{3\sigma}{2} + \frac{3p^3((1+p)(1+\sigma))^{-1/p}\left(-1+(1+\sigma)^{2+\frac{1}{p}}\right)\left(-1-\sigma+(1+\sigma)^{\frac{1}{p}}\right)}{(1+2p)(-1+p^2)\sigma^2}} \tag{7}$$

Figure 2(a) illustrates the POA bound for Pigou instances for $p = 2, 3, 4$ as a function of σ, while Figure 2(b) shows the POA bound as a function of p, with fixed $\sigma = 0, 0.1, 0.5$ and 1.

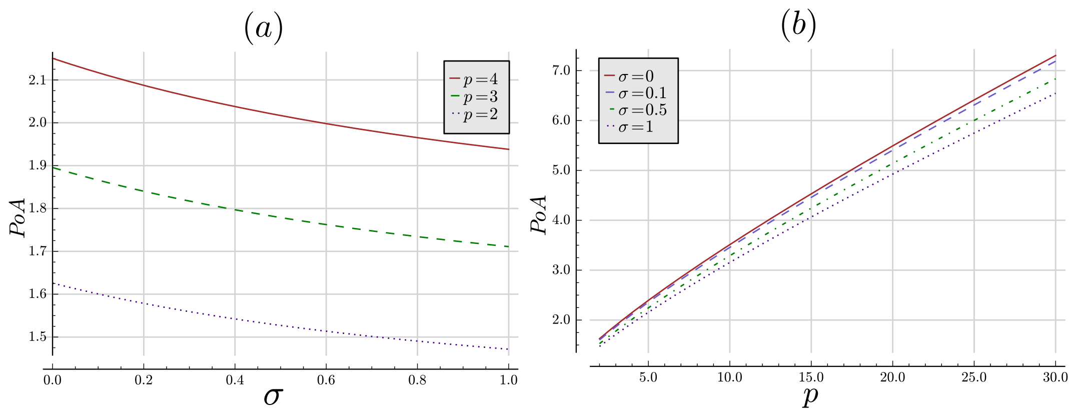

Figure 2 Smoothed Price of Anarchy of Pigou instances with polynomial latency functions.

Proof of Theorem 1. Let I be the original Pigou instance as introduced in Section 2.2. After perturbing the latency functions of I as described above, we obtain the instance depicted in Figure 3 with latency functions

$$l_1(x) = (1+\varepsilon_1)x^p \quad \text{and} \quad l_2(x) = 1+\varepsilon_2,$$

where $\varepsilon_1, \varepsilon_2 \in [0, \sigma]$ are random variables.

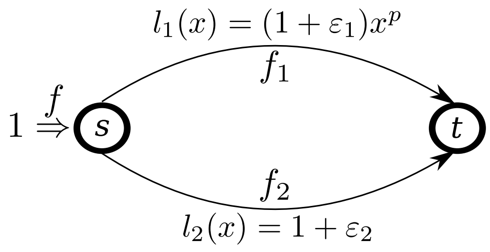
$l_1(x) = (1+\varepsilon_1)x^p$
f_1
$1 \overset{f}{\Rightarrow}$
s
t
f_2
$l_2(x) = 1+\varepsilon_2$

Figure 3 Pigou instance with polynomial latency functions and perturbations $\varepsilon_1, \varepsilon_2 \in [0, \sigma]$.

We first determine a Wardrop flow. With the addition of perturbation, edge e_1 is used as long as its latency is lower than the latency of e_2. Therefore

$$f_1 \leq \sqrt[p]{\frac{1+\varepsilon_2}{1+\varepsilon_1}}.$$

Since this can be greater than our maximum flow,

$$f_1 = \min\left(\sqrt[p]{\frac{1+\varepsilon_2}{1+\varepsilon_1}}, 1\right) \quad \text{and} \quad f_2 = 1 - \min\left(\sqrt[p]{\frac{1+\varepsilon_2}{1+\varepsilon_1}}, 1\right).$$

The flow f_1 is going to be 1 as long as $\varepsilon_2 \geq \varepsilon_1$. If this is the case, then $c(f) = 1+\varepsilon_1$. If $\varepsilon_2 \leq \varepsilon_1$, then

$$c(f) = \sqrt[p]{\frac{1+\varepsilon_2}{1+\varepsilon_1}}(1+\varepsilon_1)\left(\sqrt[p]{\frac{1+\varepsilon_2}{1+\varepsilon_1}}\right)^p + \left(1 - \sqrt[p]{\frac{1+\varepsilon_2}{1+\varepsilon_1}}\right)(1+\varepsilon_2) = 1+\varepsilon_2.$$

In order to determine $\mathbf{E}[c(f)]$ for $\varepsilon_1, \varepsilon_2$ chosen uniformly at random from $[0, \sigma]$, we need to solve the following double integral. (Note that the combined probability density function is $\frac{1}{\sigma^2}$).

$$\begin{aligned}\mathbf{E}[c(f)] &= \int_0^\sigma \int_0^{\varepsilon_2} \frac{1+\varepsilon_1}{\sigma^2} \; d\varepsilon_1 d\varepsilon_2 + \int_0^\sigma \int_{\varepsilon_2}^\sigma \frac{1+\varepsilon_2}{\sigma^2} \; d\varepsilon_1 d\varepsilon_2 \\ &= \frac{3+\sigma}{6} + \frac{3+\sigma}{6} = 1 + \frac{\sigma}{3}\end{aligned}$$

In order to compute the system optimum flow, we exploit the fact that an optimal flow is a Wardrop flow with respect to the marginal cost functions $l_1^*(x) = (p+1)(\varepsilon_1+1)x^p$ and $l_2^*(x) = 1+\varepsilon_2$. Then

$$f_1^* = \sqrt[p]{\frac{1+\varepsilon_2}{(p+1)(\varepsilon_1+1)}} \quad \text{and} \quad f_2^* = 1 - \sqrt[p]{\frac{1+\varepsilon_2}{(p+1)(\varepsilon_1+1)}}.$$

Note that σ must at most p for the optimum flow f_1^* to remain below the maximum flow. The cost of f^* is

$$\begin{aligned} c(f^*) &= \sqrt[p]{\frac{1+\varepsilon_2}{(p+1)(\varepsilon_1+1)}}(1+\varepsilon_1)\left(\sqrt[p]{\frac{1+\varepsilon_2}{(p+1)(\varepsilon_1+1)}}\right)^p + \left(1-\sqrt[p]{\frac{1+\varepsilon_2}{(p+1)(\varepsilon_1+1)}}\right)(1+\varepsilon_2) \\ &= 1+\varepsilon_2 - p(1+p)^{-1-\frac{1}{p}}(1+\varepsilon_1)^{-\frac{1}{p}}(1+\varepsilon_2)^{1+\frac{1}{p}}. \end{aligned}$$

Taking the expectation over the random choices $\varepsilon_1, \varepsilon_2 \in [0, \sigma]$, we obtain

$$\begin{aligned} \mathbf{E}[c(f^*)] &= \int_0^\sigma \int_0^\sigma \frac{1+\varepsilon_2 - p(1+p)^{-1-\frac{1}{p}}(1+\varepsilon_1)^{-\frac{1}{p}}(1+\varepsilon_2)^{1+\frac{1}{p}}}{\sigma^2}\, d\varepsilon_1 d\varepsilon_2 \\ &= 1+\frac{\sigma}{2}+\frac{p^3((1+p)(1+\sigma))^{-1/p}\left(-1+(1+\sigma)^{2+\frac{1}{p}}\right)\left(-1-\sigma+(1+\sigma)^{\frac{1}{p}}\right)}{(1+2p)\left(-1+p^2\right)\sigma^2} \end{aligned}$$

Thus

$$\mathrm{SPoA}(I,\sigma) = \frac{3+\sigma}{3+\frac{3\sigma}{2}+\frac{3p^3((1+p)(1+\sigma))^{-1/p}\left(-1+(1+\sigma)^{2+\frac{1}{p}}\right)\left(-1-\sigma+(1+\sigma)^{\frac{1}{p}}\right)}{(1+2p)(-1+p^2)\sigma^2}} \tag{8}$$

◀

Recall that the Pigou instance is the worst-case instance for the Price of Anarchy. Clearly, the Smoothed Price of Anarchy either stays the same or improves (i.e., decreases). As our bound shows, it improves but the decrease is rather low. Even for perturbations of the magnitude $\sigma = 1$, the decrease is about 10% only. Note that in this case we may double the latency functions. With increasing degree, this decrease becomes more significant. If we restrict σ to be less than or equal to $\frac{1}{p}$, which can be seen in Figure 4, then the Smoothed Price of Anarchy asymptotically remains $\Theta(\frac{p}{\ln p})$ as in the deterministic case.

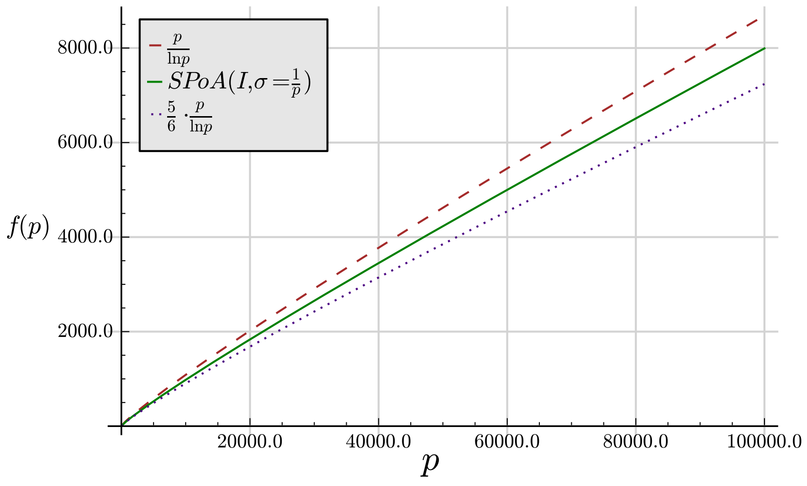

Figure 4 Smoothed Price of Anarchy of Pigou instances shown to remain $\Theta(\frac{p}{\ln p})$

4 Computational Results

4.1 Experimental Setup

In order to evaluate if real world instances behave in a different manner in relation to the worst case instances for the Price of Anarchy, we tested a few benchmark instances freely available for academic research from the Transportation Network Test Problems [1].

In these instances, the latencies follow the U.S. Bureau of Public Roads' definition, shown in (1), with $\alpha = 0.15$ and $\beta = 4$. We chose a few instances to compare and perturb, with both big and small instances evaluated. The instances details can be seen in Table 1.

Table 1 List of benchmark instances used in the experiments.

Instance Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|E\|\cdot\|K\|$
Sioux Falls	24	76	528	40,128
Friedrichshain	224	523	506	264,638
Chicago Sketch	933	2,950	83,113	245,183,350
Berlin Center	12,100	19,570	49,689	972,413,730

To find the user equilibrium and system optimum, we used the Frank-Wolfe algorithm [6]. The algorithm was implemented in C++ and compiled in 64 bit gcc version 4.4.5, in a Linux kernel version 2.6.35. The machine used for the tests has an Intel® Core™ i7 CPU with 4 cores, with 12 GB of RAM memory.

4.2 Benchmark Instances Results

We perturbed the instances with $\sigma \in \{10^{-9}, 10^{-8}, ..., 10^{-2}\}$. We also evaluated instances with a greater perturbation, with $\sigma \in \{0.1, 0.2, ..., 0.9\}$. The algorithm was stopped when it reached a relative gap less then of 0.00001, except the Sioux Falls instance which the minimum relative gap was set to 0.000001. For each perturbation magnitude σ_i, 10 runs were executed and the average value was considered. Then, the Price of Anarchy of the unperturbed instances are presented in Table 2. Also, for among all averages for the different values of σ, the mean, the standard variation, the minimum and maximum values are presented.

Table 2 Price of Anarchy and related measures found for each instance.

Instance Name	POA	Mean	Minimum	Maximum	Standard Deviation
Sioux Falls	1.039682	1.039689	1.039676	1.039707	8.049609×10^{-6}
Friedrichshain	1.086374	1.086422	1.086345	1.086599	4.996005×10^{-5}
Chicago Sketch	1.023569	1.023567	1.023561	1.023572	2.137639×10^{-6}
Berlin Center	1.006141	1.006142	1.006133	1.006155	3.831177×10^{-6}

In Table 3 we can see the average time that the perturbed instances executed. It is clear that for these instances the perturbations did not significantly alter their execution time. Note that the Sioux Falls instance takes longer than the Friedrichshain instance due to the smaller relative gap used on the Sioux Falls instance.

Both the smoothed and the original Price of Anarchy are close to one for all tested instances, which gives some empirical evidence that the worst case is not so likely to occur in real world instances. Furthermore, when we look at the Smoothed Price of Anarchy for all instances, as is shown in Figure 5, we notice that even for relatively large σ, the Smoothed Price of Anarchy remains almost constant and very close to the original Price of Anarchy.

Table 3 Execution time found for original instances and the average for the perturbed instances, in seconds.

Instance Name	UE time	SO time	Average UE time	Average SO time
Sioux Falls	1.58	1.6	1.5935	1.613314
Friedrichshain	0.43	0.92	0.431411	0.963976
Chicago Sketch	27.99	36.89	27.347117	36.568070
Berlin Center	233.91	360.07	220.975359	360.644545

The small trend that the Smoothed Price of Anarchy tends to follows on these instances seems to be related more with the particular instance than with a more general rule. This can be seen on the difference between the Friedrichshain instance and the Chicago instance, while in the Berlin instance it appears to remains constant.

The fact that the Smoothed Price of Anarchy does not drop significantly from the original Price of Anarchy, allied with these experimental results, shows that while perturbation does occur frequently in real world scenarios, it does not have a great influence on the actual distance from users equilibrium to the overall system optimum.

5 Conclusions

The Traffic Assignment Problem concerns the choice of routes in a road network given a set of users with an origin and a destination. It is of extreme importance for traffic planning and in real world cases perturbation occurs frequently, therefore it is useful to have a notion of how much can this perturbation affect its instances. It is of particular interest how the Price of Anarchy is affected in these situations, since the goal in road network planning is usually to approximate the user equilibrium to the system optimum.

We propose a perturbation model and a measure of perturbation of the Price of Anarchy based on the smoothed analysis for algorithms, the Smoothed Price of Anarchy. We give a lower bound for the Smoothed Price of Anarchy that is in the same order as the worst case Price of Anarchy for polynomial latencies.

Finally, we show experimentally that the effects of perturbation on the Price of Anarchy of real world instances, at least for the known instance benchmarks in the literature present in the Transportation Network Test Problems, are severely limited and show no general trend.

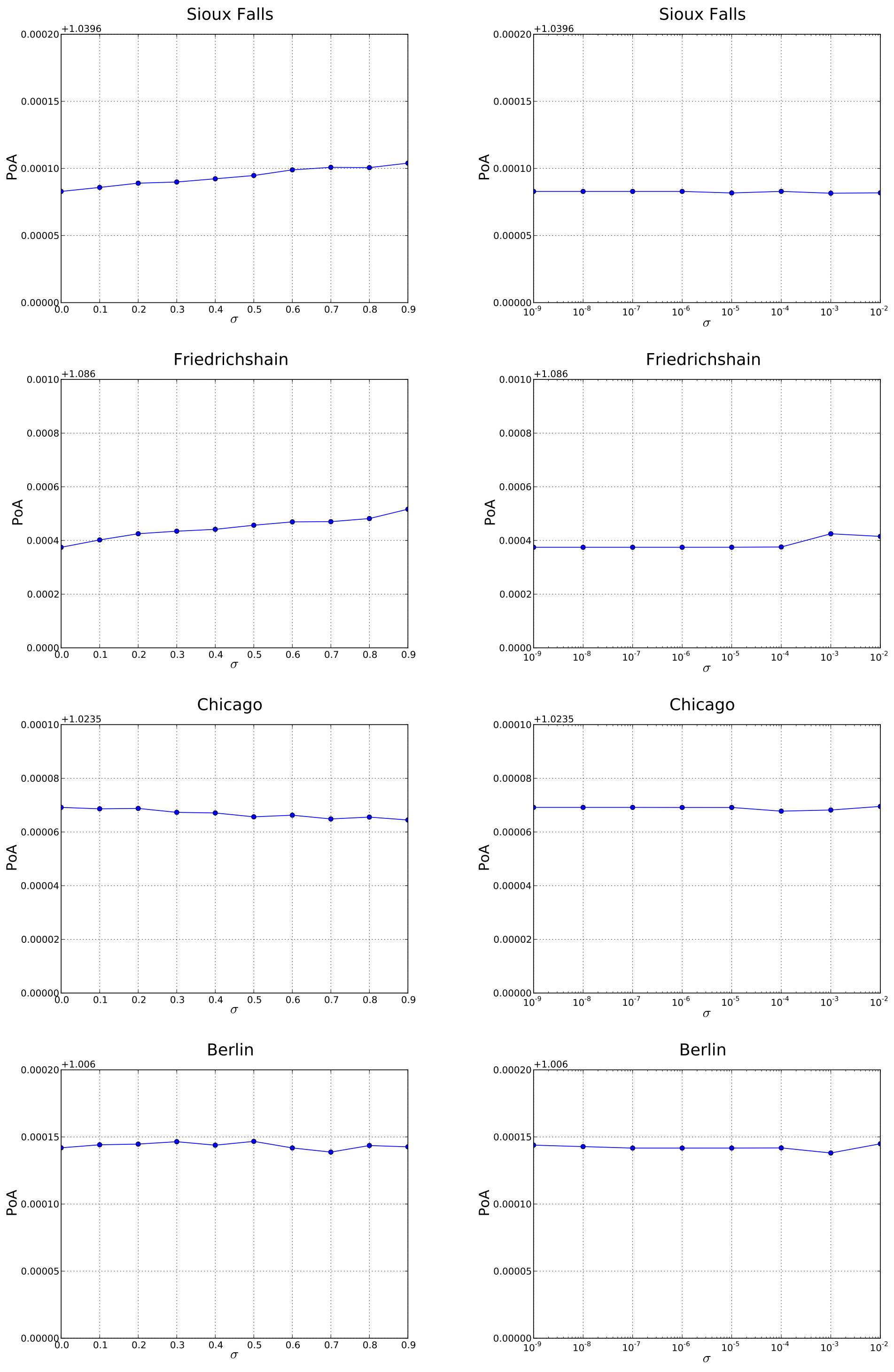

Figure 5 Experimental Smoothed Price of Anarchy for the Sioux Falls (first), Friedrichshain (second), Chicago sketch (third) and Berlin Center (forth). On the left $\sigma \in \{0.1, ..., 0.9\}$ while on the right side $\sigma \in \{10^{-9}, ..., 10^{-2}\}$.

References

1 Bar-Gera, H.: Transportation Network Test Problems. `http://www.bgu.ac.il/~bargera/tntp/` (Jun 2011)

2 Bergendorff, P., Hearn, D.W., Ramana, M.V.: Congestion Toll Pricing of Traffic Networks, pp. 51–71. Lecture Notes in Economics and Mathematical Systems, Springer-Verlag (1996)

3 Bureau of Public Roads: Traffic assignment manual. U.S. Department of Commerce, Urban Planning Division, Washington, DC. (1964)

4 Dubey, P.: Inefficiency of nash equilibria. Math. Oper. Res. 11, 1–8 (February 1986), `http://dx.doi.org/10.1287/moor.11.1.1`

5 Florian, M., Hearn, D.: Network Equilibrium Models and Algorithms, vol. 8 (1995)

6 Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Research Logistics Quarterly 3(1-2), 95–110 (1956), `http://dx.doi.org/10.1002/nav.3800030109`

7 Hearn, D.W., Ramana, M.V.: Solving congestion toll pricing models. Equilibrium and Advanced Transportation Modeling pp. 109–124 (1998)

8 Koutsoupias, E., Papadimitriou, C.: Worst-case equilibria. In: In Proceedings Of The 16Th Annual Symposium On Theoretical Aspects Of Computer Science. pp. 404–413 (1999)

9 Roughgarden, T.: The price of anarchy is independent of the network topology. J. Comput. Syst. Sci. 67, 341–364 (September 2003)

10 Roughgarden, T., Tardos, E.: How bad is selfish routing? Journal of the ACM (JACM) 49(2), 259 (2002), `http://portal.acm.org/citation.cfm?id=506147.506153`

11 Spielman, D.A., Teng, S.H.: Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. In: Journal of the ACM. pp. 296–305 (2001)

12 Wardrop, J.: Some theoretical aspects of road traffic research. Proceedings of the Institution of Civil Engineers, Part II 1(36), 352–362 (1952)