Selfish Routing in Transportation Networks

Whitman Schorn
Department of Computer Science
Oberlin College
Oberlin, OH 44074-1019
wschorn@oberlin.edu

April 30, 2013

Abstract

This paper examines a game of traffic networks in which infinitesimally small individual players selfishly determine their routes, which may or may not lead to a collectively optimal total latency across the network. Outlined are several important examples for understanding selfish behavior, a standard notation as described by [6], and several approaches to improving the game's outcomes.

Acknowledgments: This paper would not have been possible without Professor Alexa Sharp, my unerringly inspiring and understanding advisor. Thanks also go to Professor Tom Wexler, for challenging me with many interesting problems and only solving some of them. And to my brother, Will.

1 Introduction

A single car rolling down a sunny country road is, technically, traffic. Any passage of a person or vehicle along routes of transportation is. So why doesn't that come to mind?

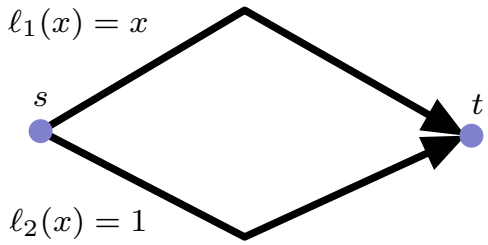
Why do commuters, city planners, and people with schedules in general cringe at the words 'there will be traffic'? In concrete terms, one doesn't care about the single car because the real worry is about thousands of other travelers changing lanes, cutting us off, and generally disrupting our drive. The sunny country road seems less important than the fastest way to work. Traffic, though it might seem innocuous, is in fact a word loaded with its worse-case scenarios: uncoordinated, selfish drivers make for poor decisions, bad outcomes and a general suffering for everyone involved. In our thoroughly optimized world, it begs the question: Why hasn't this been fixed?

This paper will attempt to answer that question, and moreover survey a theoretical framework for traffic that shows how the simple rules motivating drivers form a complex problem that defies an easy solution. To accomplish this, the paper first formalizes a model of transportation, and rigorously quantifies bad outcomes. Then, we will examine methods of coping with these outcomes, and ultimately expand our model to introduce a novel method of traffic control with its own strengths and limitations. Before any of that, though, let's examine our first example of selfish routing.

2 Motivating Example - Pigou's

Charming as the town may be, many Oberlin professors choose to commute in from Cleveland each morning. It isn't a very complicated matter, as only two roads exist between the locations: e_1 is a broad highway, which guarantees travelers can get from Cleveland to Oberlin in 1 hours without any worry of congestion.

Road e_2, on the other hand, is a small backroad, and the time it takes to travel the route changes based on how heavily it's used. One can more specifically say that some fraction x of the total routed traffic uses e_2, and that fraction (in hours) is how long the trip takes them. So if $\frac{1}{2}$ of the commuters use the road, it only takes $\frac{1}{2}$ an hour, and so on. Thus for any amount of traffic x, the latency experienced is determined by a linear function. Denote this with $\ell_1(x) = x$, and can also observe our highway, e_1, can be defined with a constant function $\ell_2(x) = 1$. It could be illustrated thusly:



The behavior of these drivers is simple: Each driver is only an infinitesimally small part of the total traffic, and they will switch their route to a shorter one if they have the opportunity to do so. So how do our drivers negotiate this? Naturally, any driver on e_1 will prefer to switch to e_2, since that route incurs less latency as long as some portion of the traffic isn't using it. Selfish behavior leads the commuters to an outcome where everyone experiences a full hour of waiting.

But what if drivers could coordinate, or had some other system governing their choices? Another distribution of traffic over the two roads might split traffic evenly - in this case, commuters on e_2 experience a pleasant $\frac{1}{2}$-hour drive, and in fact our average time for all drivers decreases to $\frac{3}{4}$ of an hour.

This formulation, *Pigou's example* [4], is small, but illustrates the key behavior that makes selfish routing challenging: independent, selfish behavior can result in socially undesirable outcomes. Having this core concept in mind, the elements of our problem can be more precisely quantified, as a first step to solving it.

3 Selfish routing definitions - Roughgarden Model

Consider a directed network $G = (V, E)$, with k pairs of nodes $\{s_1, t_1\}, \ldots, \{s_k, t_k\}$ denoting sources and destinations. Parallel edges are allowable. Let $\mathcal{P}_i$ denote the set of paths from s_i to t_i, and $\mathcal{P} = \cup_i \mathcal{P}_i$ denote the set of these sets. One can always assume $\mathcal{P}_i \neq \emptyset$ for each i. For each edge $e \in E$ there is a *latency function* ℓ_e, and an actual latency denoted by $\ell_e(\cdot)$. This latency is determined by the traffic on the edge, which is formalized as a *flow*: a function $f : \mathcal{P} \to \mathbb{R}^+$, where a flow f and an edge e,

$f_e = \sum_{P:e\in P} f_P$ define f_e as the total amount of flow on e. It is sometimes useful to refer to the pair $\{s_i, t_i\}$ and $s_i - t_i$ paths $\mathcal{P}_i$ as *commodity* i.

This encodes the basic structure of the roads, but what about our drivers? It is intuitive that the latency of a path $\mathcal{P}$ with respect to a flow f would be $\ell_P(f) = \sum_{e\in P} \ell_e(f_e)$. Furthermore each $\{s_i, t_i\}$ has a finite and positive traffic rate r_i, the amount of flow with source s_i and destination t_i. A flow is then called *feasible* if for all i, $\sum_{P\in\mathcal{P}_i} f_P = r_i$, that is if the correct amount of flow is sent over all of the paths.

Then there are two definitions of the cost $C(f)$ of a flow f, both of which are useful: it is the total latency incurred by the flow along all of the paths, and also the latency incurred by each edge. So $C(f) = \sum_{P\in P} \ell(f) f_P = \sum_{e\in E} \ell_e(f_e) f_e$.

A triple (G, r, ℓ) is then referred to as an *instance*, and a feasible flow for that instance that minimizes latency is said to be optimal.

Optimality is nice, but what about the outcomes observed in Pigou's example? In plain terms, a player will only change paths if, in comparison to their current one, some other path offers a lower latency. If the game reaches a point where no players have a reason to change their strategies, the players have reached an outcome. In game theoretic terms, a flow f for instance (G, r, ℓ) is at *Nash Equilibrium* (or is a *Nash flow*) if for all $i \in \{1, \ldots, k\}$, $P_1, P_2 \in \mathcal{P}_i$ with $f_{P1} > 0$, and $\delta \in (0, f_{P1}]$, we have $\ell_{P1}(f) \leq \ell_{P2}(f')$, where

$$f'_P = \begin{cases} f_P - \delta & \text{if } P = P_1 \\ f_P + \delta & \text{if } P = P_2 \\ f_P & \text{if } P \notin \{P_1, P_2\} \end{cases}$$

In other words, any path P_1, with non-negative flow, is full of drivers who are considering the latency of their fellow travelers on P_2, and f'_P represents the change in flow that would result from their taking a different route.

As δ tends towards 0, having continuity and monotonicity in our edge latency functions allows us to make a useful characterization of Nash flows, known as a *Wardrop's Principle* [8].

Proposition 1. *A flow f for instance (G, r, ℓ) is at Nash equilibrium if and only if for every $i \in \{1, \ldots, k\}$ and $\mathcal{P}_\infty$, $\mathcal{P} \in \mathcal{P}$ with $f_{P_1} > 0$, $\ell_{p_1}(f) \geq \ell_{P_2}(f)$.*

Put simply, a flow is at Nash Equilibrium exactly when all of the flow travels on minimum-cost paths. More specifically, all $s_i - t_i$ paths for a commodity either have 0 flow, or the same amount of latency as all other non-zero $s_i - t_i$ paths, thus all commodity i players are experiencing the same latency. Intuitively, one can see that when these conditions are not true some traffic has motivation to change paths. Furthermore, this leads us to a tidy expression:

Proposition 2. *Since all $s_i - t_i$ paths with flow experience the same latency ($L_i(f)$, say), the cost of a Nash flow f is $C(f) = \sum_{i=1}^{k} L_i(f) r_i$.*

Existing research [6] shows that Nash flows have two useful properties - one is guaranteed to exist, and it is unique, meaning for any two Nash flows f, f' for the same instance, $l_e(f_e) = l_e(f'_e) \forall e$ (interested readers can turn to the Appendix for proofs). Basically, this says our commuters will question whether $f_e = f'_e$, and if so, stick with their current route.

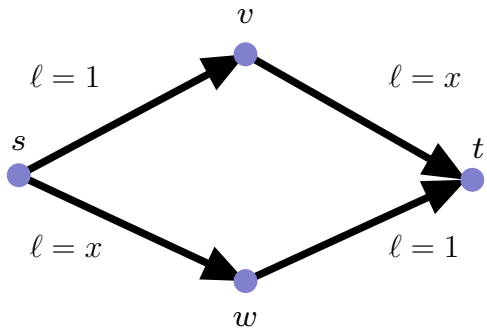
Armed with our definitions, let's take another look at Pigou's: The network is two nodes with two parallel edges, and a traffic routes itself such that the latency of each $s - t$ path is 1, just as Wardrop predicts.

4 Braess's Paradox

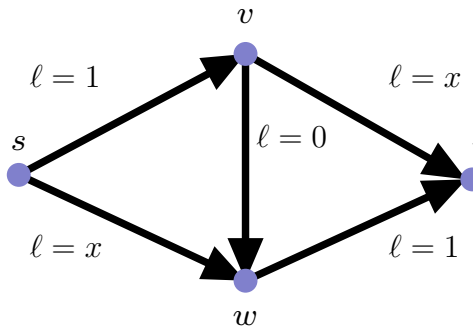
Our understanding of selfish routing, however is far from complete. Along with Pigou's example, another simple network exhibits behavior even more frustrating - and surprising - than its older counterpart.

Begin with a network as pictured in (a) below. Between s and t there exist two routes, each made up of a broad, long road and a short, narrow one. The equal paths receive equal traffic, thus giving every commuter a total $\frac{3}{2}$ latency.

(a)



(b)



Now suppose, in an effort to improve this theoretical commute, network planners added a road, from one path to the other, that was so short and incongestable that its latency function was $\ell(x) = 0$. (see (b)) Surely this could only improve the situation, or at least do no harm?

As it turns out, this addition to the network is far from innocuous: any one driver can save half an hour by following the route that uses the new road ($s \to v \to w \to t$). In the inevitable case that all drivers switch to this path, each driver gains a half-hour of latency, rather than losing that amount. Moreover, with players thus congesting the roads, no other path is superior - in fact, any other flow along the network is unstable and would eventually converge onto this one. Adding our zero-latency road then *worsened* the latency for *all* traffic!

Braess's paradox thus shows an even more dire picture of the consequences of selfish

routing: All drivers would strictly benefit from a coordination (in simply ignoring the new road), and more astoundingly the underlying network structure defied our attempts to improve latency. In this crazy mixed-up world, what kind of absolutes can be used to discern when good outcomes will go bad? In order to understand, and hopefully curb the effects of selfish routing, this problem requires the introduction of another concept central to game theory:

5 The Price of Anarchy

The Price of Anarchy, though gravely named, is actually a simple question: What is the worst-case ratio between the total latency of a Nash equilibrium and the latency of the optimal flow? This question can mean the difference between a negligible loss of efficiency and a city planner's worst nightmare. For example, in Braess' paradox, the original network had latency $\frac{3}{2}$ in both cases, so our Price of Anarchy is 1, but our 'improved' network had a nash flow / optimal flow ratio of $\frac{4}{3}$.

Rather than discussing explicit cases or ratios, it is in general to easier prove, and think of *bounds* on the price of anarchy. By showing some bound (are flows are, at worst, THIS inefficient), and then showing an example instance that is at the edge of that bound, the bound has been made tight - another proof wouldn't provide us with any more insight.

As it turns out, these tight bounds on the price of anarchy exist for most instances regardless of graph topology. Instead, the price of anarchy depends on the set of cost functions, and moreover one can assign an 'anarchy value' to functions in that set, thus reducing a vast number of possible situations to a more manageable classification of sets of cost functions. The following are suggested to the curious reader: [5], [3]. Specifically, transportation applications most commonly use quartic polynomials. [7]

6 Coping With Selfishness

6.1 Stackleberg Routing

Having a rough understanding of the cause and relative congestion of outcomes begs the question: what can be done to improve our situation? One approach, well-known in game theory, is to let one leader move first, and deliberately, after which other players follow. These are *Stackleberg Games*, and they can be adapted to fit our routing game quite neatly: Assume that the leader is in charge of some fraction of the total flow, and that this leader will route their traffic along some path in order to achieve a better flow. The followers (other players) then reach an equilibrium relative to the leader's strategy. Each time the leader updates their strategy, a new game takes place and a new equilibrium is reached.

We denote a Stackleberg instance by (G, r, ℓ, β) where (G, r, ℓ) as defined earlier, and β is the fraction of our rate r that the leader controls (so $\beta \in (0, 1)$.

So what can Stackleberg routing accomplish for our network game? Consider Pigou's example, in which the price of anarchy is $\frac{4}{3}$. Suppose that our leader controls half of the traffic, and picks the strategy of routing all traffic over the edge with cost function $\ell(x) = 1$. The remaining traffic then selfishly goes along the $\ell(x) = x$ edge, and the equilibrium thus induced is optimal.

Not bad! But Stackleberg routing isn't without its limitations. For example, if Pigou's is modified so that the cost function $c(x) = x$ becomes $c(x) = 2x$. In this case, the optimal flow puts $\frac{1}{4}$ of traffic on the modified edge. No matter what the leader does, however, the flow induced will be the same non-optimal Nash equilibrium.

Taking these capabilities and limitations into account,

A simple, but effective method for constructing a Stackleberg game strategy is know as the Largest Cost First (LCF) strategy.

1 - Compute an optimal from f^* for (G, r, ℓ).

2 - Order the edges of G from 1 to m such that $\ell_1(f_1) \leq \cdots \leq \ell_m(f_m)$

3 - Find $j \leq m$ such that j is minimal with $\sum_i = j + 1^m f_i^* \leq \beta r$.

4 - Send flow $f_i = f_i^*$ for $i > j$, $f_j = \beta r - \sum_i = j + 1^m f_i^*$, and $f_i = 0$ for $i < j$.

An edge is *saturated* by a strategy f if $f_i = f_i^*$. LCF saturates edges from the largest cost in f^* to the smallest, until there is no more controlled traffic to route.

LCF is provably quite effective in networks that consist only of parallel links, so much so that is reduces the price of anarchy to a constant. However it cannot be guaranteed to produce the optimal Stackleberg strategy, as in the following example:

Consider an instance $(G, 1, c, \frac{1}{6})$, where the network G made up of three parallel edges, with cost functions $\ell_1(x) = x, \ell_2(x) = 1 + x, \ell_3(x) = 1 + x$. LCF puts all controlled flow on one of the last two edges, while the rest of the traffic goes along the first edge. This leads to flow with a cost of $\frac{8}{9}$. A better strategy splits the controlled traffic evenly between e_2 and e_3 such that the induced flow costs only $\frac{7}{8}$.

6.2 Edges Taxes

Another tool in the arsenal of hopeful traffic managers is taxation. Ideally, this acts as a powerful factor to combat bad outcomes. To denote a system of taxation, simply assign a cost function ℓ_e a nonnegative real number t_e as the *tax* on e. Write $\ell_e + t_e$ to represent the adjusted cost function $(\ell_e + t_e)(x) = \ell_e(x) + t_e$. It is important to remember these taxes need not represent money in particular, just that given an instance (G, r, ell) one wants to find taxes t that minimize $d(G, r, \ell + t)$, where $d(G, r, \ell + t)$ is the common cost incurred by traffic in a nash flow on the instance using adjusted cost functions $\ell + t$.

So what benefits can taxes have for our networks? In Braess' paradox, for example, one might penalize players using the 0-latency edge such that no strategy would do so. Edge taxes also have the benefit of being more nuanced that simply barring use of an edge. However assigning these edge values proves to be a difficult problem, so much so that unless $P = NP$ there is no approximation algorithm that does markedly better than the trivial algorithm of setting all taxes to 0.

As it turns out, in instances with only linear cost functions, the benefits of edge taxes are no greater than only considering edge removal.

Beyond their mathematical difficulties, edge taxes and Stackleberg routing are further complicated by barriers to real-world implementation. Taxation specifically raises the question of whether the penalties a player incurs should count towards the total of the

objective function. The argument against not doing so is that whatever taxes are levied could be returned to players, possibly in some indirect form such as investing in the infrastructure they use. However there are many cases where this is infeasible, either logistically or because taxes represent time delays or other non monetary sacrifices. The worrisome scenario is one in which the taxes levied are grossly disproportionate to the original inefficiency players cope with.

The other issue of disproportionality is in Stackleberg routing, which demands an ever-increasing fractional control of traffic to improve overall network behavior. In a game with so many players that the individual is assumed to be infinitesimally small, assuming absolute control over the route of a large portion of that traffic implies a huge, and some might say contrived, amount of control.

Our final section, will examine a novel technique that attempts to incorporate elements of both of these solutions, but still retain a satisfactory level of self-determination for all players.

6.3 Red-ball networks

Red-ball networks differ from the standard model of traffic networks in two ways - some subset of edges are marked 'red-ball' edges, and some fraction of the traffic is designated *red ball* traffic (from herein called *marked* traffic and edges). The only constraint is that unmarked traffic may not use marked edges - all other viable strategies are acceptable for players.

Most formally, red-ball networks can be defined by the more abstract nonatomic congestion games.

Consider a NCG identical to the standard implementation described in [6]], except that our single player-type has been split into two groups - one of size $(r - r_b)$ (the unmarked traffic) and the other of size r_b (marked). Marked traffic retains an identical strategy set and rate of consumption for all elements, but unmarked traffic i has its strategy set restricted to $\mathcal{P}_i$ s.t. $e \notin E_b \forall e \in \mathcal{P}_i$.

One main point that makes red-ball networks interesting is that their application to

the real world doesn't require the network to actively control its traffic or levy extra taxes. It would not be infeasible to strategically implement red-ball style traffic controls simply by selecting some subset of license plate numbers to serve as 'marked' traffic, and then making a nominal investment in signage and public education (as opposed to costly infrastructure improvements). The networks take their name from the Red Ball Express, a system used by Allied vehicle convoys to move traffic across France while allowing for civilian traffic. [1] [2]

While there may be other problems that outweigh these benefits, red-ball networks still serve as another tool to gain insight on the complexity and robustness of the problem that are caused by selfish routing.

6.4 Conclusions

While conducting research for this paper, the author has encountered a wide spectrum of approaches to creating and conveying ideas about selfish routing. The subject has spanned across disciplines of economics, math, and computer science, while involving researchers across the globe. There is ongoing research into NCGs that may provide still more better solutions to the problems and paradoxes of a congested network. The next time the reader find themselves stuck in traffic, they may find it a little easier to cooperate with fellow drivers by knowing that just by engaging in little un-selfish behavior, they can make everyone's life a bit more optimal.

7 Appendix

7.1 Proofs - existence and uniqueness of Nash Flows

These proofs require further definitions:

(1) If x and y are two points in the Euclidean space $\mathcal{R}^n$, then a *convex combination* of x and y is a point on the line segment between the two of them of the form $\lambda x + (1-\lambda)y$ for some $\lambda \in [0, 1)$.

(2) A subset S of $\mathcal{R}^n$ is *convex* if it contains all its convex combinations (if x, y lie in S, so does $\lambda x + (1-\lambda)y$ for some $\lambda \in [0,1)$).

(3) A function $h : S \to \mathcal{R}$ defined on a convex subset S is itself *convex* if all line segments between two points on h's graph like above the graph: $h(\lambda x + (1-\lambda)y) \leq \lambda h(x) + (1-\lambda)h(y)$

(4) A function $\ell : \mathcal{R}^+ \to \mathcal{R}^+$ is *semiconvex* if the function $x\dot{\ell}(x)$ is convex.

Proposition 3. *Every instance (G, r, ℓ) admits a Nash flow.*

Proof. For an instance (G, r, ℓ), define the following nonlinear program (NLP):

$$\begin{aligned} \text{Minimize} \quad & \sum_{e \in E} h_e(f_e) & \\ \text{subject to} \quad & \sum_{P \in \mathcal{P}_i} f_P = r_i & \forall i \in \{1, \ldots, k\} \\ & f_e = \sum_{P \in \mathcal{P}: e \in P} f_P & \forall e \in E \\ & f_P \geq 0 & \forall P \in \mathcal{P}, \end{aligned}$$

Where $h_e(x) = \int_0^x \ell_e(t)dt$, is convex. The Nash flows of (G, r, ℓ) are precisely the optima of this program.

Note that our total cost, $C\dot{()}$ is a convex function by (3). The program (NLP) is a *convex program* when all cost functions are semiconvex. Furthermore, one need not distinguish a flow from the point in $(|\mathcal{P} + |E|)-$dimensional Euclidean space that it induces.

By definition, the derivative of h_e is ℓ_e, a continuous nondecreasing function. Therefore, each h_e is a convex function, and the program (NLP) is a convex program. By Proposition 1, a solution to (NLP) is optimal if and only if it is a Nash flow for (G, r, ℓ). (NLP) has a continuous objective function and a closed, bounded feasible region, and therefore admits an optimal solution.

□

This proof also illustrates the computational complexity of finding the Nash Equilibrium for an instance - all nonlinear programs can be solved to an arbitrary degree of decision in time polynomial in the size of the instance and bits of precision of the answer. But can one be sure there's only a single answer to find?

We next consider the uniqueness of Nash flows:

Proposition 4. *If f and f' are flows at Nash equilibrium for the instance (G, r, ℓ), $C(f) = C(f')$.*

Proof. Suppose f, f' are flows at Nash equilibrium for (G, r, ℓ). By the previous proof, we know this means they are also global optima for (NLP). Because (NLP)'s objective function is convex, whenever $f \neq f'$ the objective function must be linear between the two values; otherwise a convex combination of the two would be a feasible solution for (NLP) with a smaller objective value. Because (3) must hold with equality for all of the convex combinations. Since every function h_e in (NLP) is convex, this can only occur if every function h_e is linear between f_e and f'_e. It then follows that by the definition of h_e and continuity of the cost function ℓ_e, it follows that every ℓ_e is constant between f_e and f'_e - therefore they must be the same. □

References

[1] A. Axelrod. *The Real History of World War II: A New Look at the Past*. Real History Series. Sterling Publishing Company, Incorporated, 2008.

[2] David Colley. *The road to victory : the untold story of World War II's Red Ball Express*. Warner Books, New York, NY, 2001.

[3] José R Correa, Andreas S Schulz, and Nicolás E Stier-Moses. Selfish routing in capacitated networks. *Mathematics of Operations Research*, 29(4):961–976, 2004.

[4] A.C. Pigou. *The Economics of Welfare*. Number v. 1 in Cosimo classics economics. Cosimo, Incorporated, 1932.

[5] Tim Roughgarden. The price of anarchy is independent of the network topology. *Journal of Computer and System Sciences*, 67(2):341–364, 2003.

[6] Tim Roughgarden. *Selfish routing and the price of anarchy*. MIT Press, Cambridge, Mass, 2005.

[7] Y. Sheffi. *Urban Transportation Networks: Equilibrium Analysis With Mathematical Programming Methods*. Prentice-Hall, 1984.

[8] J.G. Wardrop. *Wardrop of Some Theoretical Aspects of Road Traffic Research-Road Paper*. Number no. 36 in Road Paper. Road Engineering Division, 1952.