Best Response Dynamics for Congestion Games

Daniel Bickley*
Miami University

ABSTRACT

This paper looks at the process of Best Response Dynamics and an algorithm derived from the process. The algorithm can be used to find an equilibrium of a congestion network. The equilibrium will have the property that each object experiences the least amount of congestion possible, given the paths of all of the other objects. The equilibrium of networks are extremely important, every traffic pattern or computer network has an equilibrium. The equilibrium can be used to find how efficient a given network is. The equilibrium can also be used to determine how to make the network more efficient. This paper introduces the field of math behind Best Response Dynamics, called Game Theory, with several important definitions. After establishing a baseline level of knowledge, the algorithm is introduced, stressing important design decisions. Finally, the algorithm will be analyzed to determine how close the output of the algorithm is to the optimal solution.

1 INTRODUCTION

In order to fully understand Best Response Dynamics, it is important to be familiar with a branch of math known as Algorithmic Game Theory. This field refers to the use of an algorithmic or mathematical method to determine how individuals (players) can receive the best possible outcomes (payoffs). Players can also seek to reduce costs, however the two are essentially synonymous. Game Theory is especially important when studying routing algorithms, as all routers seek to reduce congestion and costs of moving data.

The study of Game Theory is focused around situations known as "games". Games are situations that contain a series of players either working together or competing with each other for specific outcomes. Each player can choose their own actions, also called strategies, such as deciding where to send a specific data packet. The strategies that each player adopts can alter the payoff for a different player. Some strategies will provide better payoffs when an opponent chooses strategy A, but the same strategy might produce a smaller payoff when the same opponent chooses strategy B.

One of the key subjects in Game Theory that is vital to Best Response Dynamics is known as the Nash Equilibrium. The Nash Equilibrium of a game refers to a specific configuration when all players of a game do not have any incentive to change their own strategy. In layman's terms, when the strategy of every player will produce the largest payout given the strategies of all the other players, then the configuration is a Nash Equilibrium.

This paper will examine a specific type of game known as a "Congestion Game". Congestion games focus around a network and all of the data flowing through it. Each piece of data flowing through the network is one player within the game. Every player has their own destination with their own strategies about where to go. This fact makes it perfect to determine a Nash Equilibrium for each congestion game. In order to find the Nash Equilibrium, you need to go through the process of Best Response Dynamics.

*e-mail: bickledb@miamioh.edu

2 MOTIVATION

Networks and congestion plague everyone every day. The congestion can take the form of annoyingly long traffic jams or it can cause your connection to the Internet to become slow. Determining the equilibrium of networks can help reduce the traffic or make life easier in several ways. First, by calculating the equilibrium of a network, you can determine the price of taking an edge away from the network temporarily or permanently. Consider when a driver is going to their destination, and one key road is closed. What other road would be the fastest detour? Determining the equilibrium for the new network without the edge will show the best detour will be. Then, when a road has to be closed, the traffic delay will be as small as possible. Second, calculating the equilibrium of a network can show what edges really matter within it. If all of the paths in a network use one specific edge, then it could be a good idea to add a secondary connection in case of disaster. Or, it can be used to determine if there is an edge that is causing a bottleneck. If such an edge exists, then it can be changed to make the entire network faster. Calculating the equilibrium is a great way to plan for the future and prevent huge issues before they occur. Third, it is possible to calculate when closing an edge or a road will actually reduce the amount of traffic! There have been several real world examples where closing roads in cities actually reduced the overall amount of traffic! In 1990, a brief closure of 42nd street in New York City reduced overall traffic. [4] By calculating the equilibrium of networks with Best Response Dynamics, similar situations could be found, allowing swift and decisive action.

3 ALGORITHMIC GAME THEORY

Before Best Response Dynamics can be used and analyzed, it is important to have a fundamental knowledge of Algorithmic Game Theory. While several key phrases were defined earlier in the paper, more detail is needed to understand the process of Best Response Dynamics. This section focuses on Congestion Games, Nash Equilibrium, and the definition of Best Response Dynamics.

3.1 Congestion Games

The definition of a Congestion Game is extremely similar to a generic "game" but with a few extra bells and whistles. Mathematical definitions of Games and Congestion Games can be found below. There are many different versions of how congestion games are played, such as if the players can share edge costs or not. The rules for the congestion games used in this paper allow for sharing of edges. This means that if there are n players using an edge e, then the cost of e is divided by n and shared with each player. More rules force every player to start out in their own node and have to go to a specific end node.

Definition 1 *A Game consists of:*
A list of players A $(a_1, a_2, a_3 ... a_n)$*, A set of strategies S* $(s_1, s_2, s_3, ... s_m)$ *for each player in the list. Each strategy has an associated outcome O* $(o_{s_1}, o_{s_2}, o_{s_3} ... o_{s_n})$ *that will depend on what all of the other players do.*

Congestion games are simply games that have additional rules that are added onto the original game. In congestion games, the outcomes are replaced with the total costs of the paths chosen. The players also seek to reduce the amount of congestion they experience, instead of maximizing their own payoffs. This paper bases the definition of Congestion Networks on what is described by Tardos by combining the notation and definition of congestion graphs and concepts from Game Theory. [3]

Definition 2 *A congestion game consists of:*
List of Players A, denoted as $(a_1, a_2, ...a_n)$. *List of Nodes M, denoted as* $(m_1, m_2, ...m_k)$. *List of Edges, E, denoted as* $(e_1, e_2, ...e_p)$. $\forall a \in A, \exists t \in E$ *called the source node of a.* $\exists t \in M$ *called the terminal node.* $\forall t \in E, \exists c$ *such that c is the positive weight of edge e.*

One of the key parts of congestion games is that each player can go through a process called Best Response Dynamics to find the best strategy to follow. Best Response Dynamics works by calculating if a player would get to their destination with less congestion if they chose a different strategy. If there exists any such strategy, then the player will change their own strategy to follow it. Eventually, the process will cycle through each individual, until an equilibrium is achieved. This equilibrium has a special name, the Nash Equilibrium.

Definition 3 *A Nash Equilibrium is a situation where each player has selected a strategy such that no matter what strategy other players choose, each player would not benefit from switching strategies.*

It is important to note the fact that a Nash Equilibrium may not be the best possible outcome for a game. There can be other sets of strategies that can produce a higher payoff overall, but are more costly for individual players. Each player only cares about their own payoff, and minimizing their own congestion. If there is a situation where all the players would benefit as a whole, but a specific player needs to follow a path with more congestion, the player would not budge. This idea will be elaborated in the section "Analysis of the Algorithm".

3.2 Best-Response Dynamics

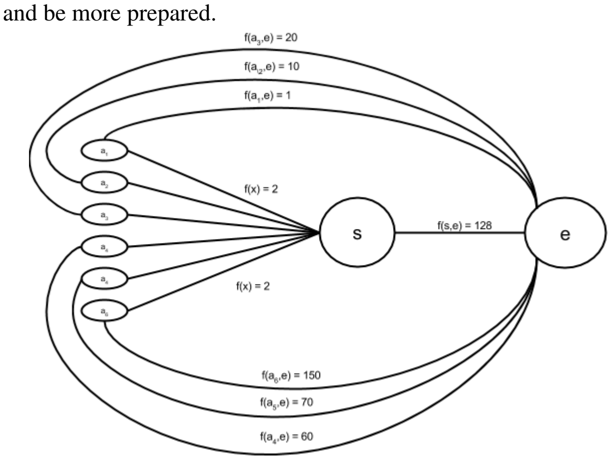
Best Response Dynamics is an iterative process where a set of players each make decisions about how to maximize their own payout. To facilitate discussion, we will use the definition of Best Response Dynamics as defined by Tim Roughgarden [5].

Definition 4 *Best Response Dynamics is the following system:*
Let situation s not be a Nash equilibrium.
Pick arbitrary player i and a better strategy $s_i^{'}$, *then i will change from* s_i *to* $s_i^{'}$

In regular terms, Best Response Dynamics is a process where every player in the game has an opportunity to decide if they would benefit from changing strategies. If the player would benefit, then they will immediately change strategies. Then, every other player has an opportunity to change their own strategy based on the new set of strategies. A real world example of Best Response Dynamics is when a driver uses a back road instead of a highway to avoid rush hour traffic. The driver evaluates their potential payoff when they realize rush hour traffic is occurring. Then they search for quicker ways to their destination, and go on a back road. Other drivers may follow them, until the traffic in the back road is greater than the rush hour traffic. When implemented in algorithmic form, Best Response Dynamics is an iterative process. Whenever one player in the congestion game changes paths, every other player has the opportunity to react. The process will continue until no single player would benefit from changing their strategy, which occurs when a Nash Equilibrium is achieved. One of the main issues posed by this algorithm is the question of whether or not Best Response Dynamics will ever stop. To put a long story short, Best Response Dynamics will eventually reach a Nash Equilibrium, and Tardos' proof is outlined later in this paper.

Consider this example, which is based on a simpler example used by Tardos [3] to explain Best-Response Dynamics, but with more real life embellishment:

Suppose everyone in a college level Algorithms class $(a_1, a_2, a_3, a_4, a_5, a_6)$ is running late to a very strenuous and long exam. Parking near the building is extremely limited, so every student needs to get to the parking lot as fast as possible. Each student has an individual car that they can use to drive to their exam, or they can ride a bus together. This situation is presented in the figure below. In order to ride the bus, the students must go to the bus stop individually, each suffering a cost of 2 units, before suffering the cost of riding the bus to the exam, in this case 128 units. The cost of the bus is high because of the long wait time at the bus. Then, each individual has separate costs of driving alone, denoted by $f(a_n, e)$. In this situation, the costs of each road include both time to drive, money to park, and potential studying time. Driving separately will cause each individual to experience the whole amount of cost, and sharing a road (such as using the bus to ride to the exam) allows the individuals to share the cost as they can study for the exam together and be more prepared.



Each of the students knows the following:

1. They all start at separate dorm rooms, denoted as $(a_1, a_2, a_3 ... a_6)$.
2. If they choose to ride the bus, then they must go to the parking lot at their dorm, denoted as P.
3. Each student would like to get to the exam as soon as possible, do as well as possible, and reduce their parking costs as much as possible.
4. Because the students are young and hip, they each constantly update social media about what they are doing, so every student instantly knows about what every other student is doing.
5. Each of the students have different cars they can drive to the exam. Therefore, some individuals will be able to get to the exam before others in varying amounts of comfort.

Now, consider what is going on through each student's mind.

In this situation, a_1 will consider two of his choices, they can either take their own fast car and get to the exam very quickly, pay for parking, with some time left to study or else they could ride the bus, get to the exam slightly later, not pay, and not study as much. Judging his options, a_1 would logically choose to drive himself to the exam, as that produces the least amount of cost.

Then, a_2 will consider the two choices of driving or riding the bus. Their car is not as fast as a_1's car, but it still will get a_2 to the exam for less cost than what the bus would inflict. So, logically, a_2 will choose to drive their own car to the exam.

The same thought process will dominate for $a_3, a_4, and a_5$, for similar reasons. However, a_5 finds that driving to the exam is only slightly better than taking the bus. Finally, poor a_6 gets to decide what they want to do. Their car is old, broken down, and completely unreliable. It is doubtful that it could even make it to the exam. So, a_6 decides to use the bus. Here is where the problem gets interesting.

Now, a_5 sees that his friend a_6 is using the bus. Realizing that going on the bus would now be better because the cost can be shared between a_5 and a_6, a_5 hops on the bus. Now, the cost of the bus is split among two people. Now, a_4 realizes that more of his friends are on the bus, so he goes to join as well, spreading the cost out further.

This situation repeats itself all the way up to a_1. Despite the fact that a_1 could hop on the bus and share the cost of the bus even more, the overall cost of riding the bus would still be greater than driving alone. Therefore a_1 chooses to drive to the exam.

A quick look at how every student fared shows that no student would be better off if they switched their methods of getting to the exam. Everyone on the bus, a_2, a_3, a_4, a_5, a_5, are all having a grand old time studying together and not paying for parking. At the same time, a_1 is perfectly happy to drive to the exam alone, pay for parking, but show off his car along the way. Therefore, the process of Best Response Dynamics found an equilibrium where no student would benefit from changing strategies. In fact, if there were any better strategies, the students would take it.

This is a very basic example of how Best Response Dynamics works. Most of the time, the process will take more iterations to solve, and calculating the best paths will be more difficult. However, it can be easily done if the algorithm that is defined in the next section is used.

4 ALGORITHM

Now that a baseline knowledge of Congestion Games, Nash Equilibrium, and how Best Response Dynamics works has been reached, the next step is to define the problem the algorithm will solve and then describe how the algorithm works in detail.

4.1 Problem

We will formally define the problem as the following using notation and definitions adapted from Tardos [3]:

- Input: Directed Graph $G = (V, E), \forall i \in E, c_i \geq 0, \exists j \in E$ such that E_j is designated as a source node S, a list of all players $A(a_1, a_2, ..a_n)$, and $\forall k \in A, \exists l \in E$, designated as the terminal node for a_k.
- Output: List of paths P $(p_1, p_2, ...p_n)$
- Goal: $\forall k \in P, c_k^p$ is at a minimum given the cost of all of the other paths $\in P$.

Essentially, the algorithm takes in a directed graph G with non-negative edge values, a list of all of the players within the graph, G contains a node specifically for every player in the graph called the Terminal Node, and a node that all of the players are trying to reach, called Source.

The output of the algorithm is defined as a list of paths, with one path for each player of the game. This path will be the sequence of edges each player will follow starting from the terminal node of each player to the source node, with their corresponding weights.

The algorithm's goal is to minimize the cost of traveling around the network given the total costs of all of the other players. This means that the list of paths in the output form a Nash Equilibrium of the given graph G.

4.2 Algorithm Definition

The following algorithm is taken from the Wikipeidia page for Braess's Paradox [1] , with slight changes to notation to increase understanding and make the analysis easier.

Let directed graph G with players $(a_1, a_2, ...a_n)$ and strategies P form a congestion game.

```
  while P is not at equilibrium
    compute the potential energy e of P
    for each driver d in P:
      for each alternate path p available to d
        compute the potential energy of the pattern when d takes
        path p
        if n ¡ e:
          modify P so that d takes path p
continue the first while
```

The algorithm starts with a congestion game, as defined in the previous sections. Then, while the players have not achieved equilibrium, they search through the graph to find a better path to their destination. If the player finds a better route, they choose it, and the process continues. The algorithm looks very simple, except for one key issue, how do we calculate the potential energy of each path?

For the sub-procedure of calculating the potential energy, we will use an algorithm and notation as defined by Tardos. [3]

The analysis will use a notion of Potential Energy, denoted as ϕ, that is based on the cost of an edge being shared with every player that uses it. Additionally, the cost of an edge k will be denoted as c_k, and the cost of a path k will be denoted as c_k^p. To start off this discussion, let the cost of an edge f to be defined as the original cost of f divided by the number of players.

$$c'_f = \frac{c_f}{n_f} \tag{1}$$

Consider a situation where player a_{n+1} wants to move from edge e_k to an empty edge e'_k. To player a_{n+1}, the new edge e'_k reduces the amount of congestion a_{n+1} experiences. When a_{n+1} leaves e_k to switch, the overall cost of the graph goes down by $\frac{e_k}{n+1}$. This makes sense, as the cost of edge k was split among $n+1$ players, and one of them has been removed, removing one $\frac{e_k}{n+1}$ from the total cost of the graph. However, player a_{n+1} has to go somewhere, so it moves to edge e'_k. This adds c'_k back into the overall cost of the network, as there is now one player on edge e'_k.

Remark 1 *For now, we will not consider the other players using e_k. Their overall costs will rise, but we only care about the current player we are looking at, a_{n+1}. The other players using e_k will be able to respond at a later time.*

The only way for a_{n+1} to switch edges is for the following equation to be true.

$$\frac{c'_k}{n'_k} < \frac{c_k}{n_k + 1} \tag{2}$$

Unfortunately, most players will need to go through multiple edges to get to their ending point, forming a path. Calculating the cost of a path is very similar to calculating the cost of an edge. The formula to calculate the path cost is as follows:

$$c_e^p = \sum_{e \in p} \frac{c_e}{n_e} \tag{3}$$

In order for a player a_{n+1} to switch between paths, a similar circumstance as Equation 2 is followed, however with path costs instead of edge costs.

The nice part of calculating costs this way is that we can easily define the potential of a path and the potential of the entire graph.

Let us consider an arbitrary path p_k, with n players, $(a_1, a_2, ...a_n)$. Currently, the overall cost of the path is $c_k^p = \frac{c_k^p}{n}$. Consider one player changing paths. Then, the new cost of the path is $c_k^p = \frac{c_k^p}{n-1}$. Then, another player changes paths, changing the cost of the path to $c_k^p = \frac{c_k^p}{n-2}$. We can define the potential of path p_k as $\sum_{m=1}^{n} \frac{p_k}{m}$. This is the potential amount of cost that can be removed when an arbitrary player changes from the path to a different path. We will expand the summation as follows:

$$\phi_{p_k} = \frac{p_k}{1} + \frac{p_k}{2} + ... \frac{p_k}{n} \tag{4}$$

Now, we will use a little bit of algebra and knowledge of infinite series to make the potential of a path more clear.

$$\phi_{p_k} = p_k\left(\frac{1}{1} + \frac{1}{2} + ... + \frac{1}{n-2} + \frac{1}{n-1} + \frac{1}{n}\right) \tag{5}$$

Notice the expression within parenthesis. This is an example of the Harmonic Series! To make the notation as clear as possible, we will define X_e to be the number of players on a given edge within a path. Then, let $H(x)$ be a shortcut for the Harmonic series.

$$\phi_{p_k} = \sum_{e \in k} c_e H(X_e) \tag{6}$$

Finally, we need to use the potential of a path to calculate the potential of the entire graph. This is very simple, the potential of the graph is the sum of all of the potentials of the paths.

$$\phi_G = \phi_{(p_1, p_2, ...p_k)} = \sum_{e \in p, \forall p} c_e H(X_e) \tag{7}$$

With that, we have all the necessary machinery to be able to implement the Best Response Dynamics algorithm to find the equilibrium of a graph. However, there are several important details that must be proved for this algorithm to be used.

5 Proofs

The first theorem and proof is taken from Tardos [3] with changes to the notation.

Theorem 1 *The potential ϕ_G cannot be lowered past a certain optimal value.*

This proof is fairly obvious. Suppose that there is an optimal value of ϕ_G. Because each edge weight must be greater than or equal to 0, then this optimal value forms a lower bound for the equilibrium of G. No matter what combination of paths player a_t follows, it could never have a smaller potential than the theoretical optimum. Therefore, there is a lower boundary for the potential.

Theorem 2 *Suppose that the current set of paths is $(p_1, p_2, ...p_k)$, and player a_t updates their path from p_j to p'_j. Then, the new potential $\phi(p_1, p_2, ..., p'_j...p_k)$ is strictly less than the old potential $\phi(p_1, p_2, ...p_j....p_k)$*

In order for a player a_t to change their path from p_j to p'_j, then $(c_j^p)' < \frac{c_j^p}{n}$. The new section of paths will be denoted as $p_{newStuff}$, and the selection of paths the player bypasses will be denoted as $p_{ignored}$ After this switch, a_t still goes through all of the same edges before the change, and then it follows the rest of the new path to get to the end node. We then know the following:

$$\sum_{\forall e \in p_{newStuff}} \frac{(c_e^p)'}{n+1} < \sum_{\forall t \in p_{ignored}} \frac{c_t^p}{n} \tag{8}$$

This means that in order for a_t to change to a different path, the new section between where a_t changes paths and the end node must be smaller than the old section between where a_t changes paths and the end node. Now, we need to determine how the potential is changed when a_t changes their path.

Based on equation 8, and our definition of potential, the increase of the potential of $p_{newStuff}$ increases when a_t decides to move down $p_{newstuff}$ is as follows:

$$\sum_{\forall f \in p_{newStuff}} c_f^p(H(p_f + 1) - H(p_f)) = \sum_{\forall f \in p_{newStuff}} \frac{c_f^p}{n+1} \tag{9}$$

This equation shows that the change in the potential of $p_{newStuff}$ changes by the same amount as the cost of adding a new player to every part of the path in $p_{newStuff}$. This should not be a very surprising revelation.

Similarly, the decrease of the potential of p_{old} can be shown to be:

$$\sum_{\forall f \in p_{old}} p_{f_c}(H(p_f) - H(p_f - 1) = \sum_{\forall t \in p_{ignored}} \frac{c_t^p}{n} \tag{10}$$

When you combine equations 8, 9, and 10, it becomes obvious that the decrease of the potential of p_{old} will always be larger than the increase of $p_{newStuff}$. Therefore, Theorem 2 is true.

Theorem 3 *Best Response Dynamics always leads to a set of paths that form a Nash equilibrium solution.*

Because of the proof of Theorem 2, this is almost trivial. Remember that the Nash equilibrium will be found when no player a_t could benefit from changing their own strategy given all of the other player strategies. Because of Theorem 1, the potential of ϕ_G can only be reduced to a certain point. Then, because of Theorem 2, when a player a_t changes their path, ϕ_G will always be reduced, at some point the Best Response Dynamics algorithm will terminate at some sort of equilibrium.

6 Analysis of the Algorithm

Currently, this algorithm does not provide a decent running time function to analyze. This is due to the fact that there are a huge number of possible paths. While the algorithm will always terminate in a finite number of steps, by Theorem 2 and Theorem 3, we do not know exactly how long it will take. Furthermore, analyzing the running time is not necessarily the best way to analyze the algorithm, because the running time does not necessarily depend on the input size. A given monstrously large input network could arrive at a equilibrium with a few iterations, and a relatively small input network could take a much longer time to come to an equilibrium. With that in mind, the best way to analyze the algorithm is to figure out the difference between the equilibrium Best Response Dynamics finds and a theoretical optimum. To find that difference, the upper bound of the price of stability needs to be determined. The price of stability will be defined according to Anshelevich [2].

Definition 5 *The Price of Stability is the ratio of the best possible Nash Equilibrium of a network and its cost to the optimum network cost.*

To bound the price of stability, there are several more tools we need to prove.

Let $C(p_1, p_2, ...p_k)$ be the total cost for every player when their chosen paths are $(p_1, p_2, ...p_k)$ Given this fact, we can produce a relationship between C and ϕ

Theorem 4 *For a set of paths,* $(p_1, p_2, ...p_k)$
$C(p_1, p_2, ...p_k) \leq \phi(p_1, p_2, ...p_k) \leq H(k) * C(p_1, p_2, ...p_k)$

To prove this, we need even more notation! Let E^+ denote the set of edges that are used be every $p \in P$. Now we can create the following identity:

$$C(p_1, p_2, ...p_k) = \sum_{e \in E^+} c_e \tag{11}$$

Now, we will add it another bit of notation, $\forall e, x_e \leq k$. Then, we can write the following equation.

$$C(p_1, p_2, ...p_k) = \sum_{e \in E^+} c_e \leq \sum_{e \in E^+} c_e H(x_e) = \phi(p_1, p_2, ...p_k) \tag{12}$$

This equation states that the total cost of all the edges used in every p is less than or equal to the potential of the graph G. This makes sense, considering the harmonic series will always be greater than 1, so when you multiply the cost of each edge by it, the end result will be larger.

The theorem will be proved with by the following equation:

$$\phi(p_1, p_2, ..., p_k) = \sum_{e \in E^+} c_e H(x_e) \leq \sum_{e \in E^+} c_e H(k) = H(k) C(p_1, p_2, ...p_n) \tag{13}$$

This equation shows that the potential will always be less than or equal to all of the costs of the edges used, multiplied by the Harmonic series. If you now look at Equations 11 to 13, it is trivial to see why Theorem 4 is true.

Thanks to Theorem 4, we can now determine what the Price of Stability is for an instance of the problem.

Theorem 5 *There is a Nash Equilibrium solution where the total cost to all agents is greater than the social optimum by at most a factor of H(k).*

To prove this, consider a social optimum that uses paths $p_1^*, p_2^*, ...p_n^*$). Now if we use Best Response Dynamics, a new Nash Equilibrium will be found that uses paths $p_1, p_2, ...p_n$. By Theorem 2, $\phi(p_1, p_2, ...p_n) \leq \phi(p_1^*, p_2^*, ...p_n^*)$ Now, we just have to compare this outcome with the proof of Theorem 4. From there, we can create the equation

$$C(p_1, ...p_k) \leq \phi(p_1, ...p_k) \leq \phi(p_1^*, ...p_k^*) \leq H(k) C(p_1^*, ...p_k^*) \tag{14}$$

As you can see, the price of stability has an upper bound of a factor of $H(k)$.

References

[1] Braess's paradox, October 2014.

[2] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '04, pages 295–304, Washington, DC, USA, 2004. IEEE Computer Society.

[3] J. Kleinberg and E. Tardos. *Algorithm Design*, chapter 12, pages 690–702. Pearson.

[4] G. Kolata. http://www.nytimes.com/1990/12/25/health/what-if-they-closed-42d-street-and-nobody-noticed.html. *New York Times*, 1990.

[5] T. Roughgarden. Algorithmic game theory lecture 16: Best-response dynamics. Class Notes, November 2013.