Mechanism Design for Networks of Smart Cars*

Joshua Grinberg, Amr Mohamed, Joseph Shayani

January 4, 2017

Abstract

We study how to improve travel efficiency in networks of smart cars by allowing drivers to communicate their priority levels. The goal is to settle conflicts between cars at intersections in a way that minimizes societal cost. The protocol we introduce implements a mechanism to elicit drivers' priority levels that incentivizes truthful reporting using a wait-period and then settles conflicts greedily. We demonstrate empirically that the mechanism is truthful and the protocol increases efficiency in some simple networks.

1 Introduction

Many car manufacturers are beginning to install a new technology in cars called VANET (Vehicle Ad hoc Network), which allows cars to communicate wirelessly with other nearby VANET-equipped cars (See [MHM15]). The benefits of this technology include safer driving and more efficient routing. Cars in communication can alert nearby vehicles that another car is expected to run a red light, and they can help coordinate traffic to reduce congestion. An estimated 1.2 million lives are lost in car accidents worldwide annually, and congestion in the United States alone is estimated to exceed $100 billion annually.[1]

Our report investigates how VANET can be used to improve transportation efficiency by coordinating traffic so that it favors cars that are in a rush or in an emergency (i.e. "high-priority" cars) over cars with time to spare (i.e. "low-priority" cars). Existing routing protocols, like stop signs or traffic lights, may not be efficient because they do not favor high-priority cars over low-priority cars. In our report, we define a Greedy protocol for settling "conflicts" between cars at intersections given information about cars' priorities, and we design a mechanism, Button for eliciting priority information from cars that incentives cars to reveal their priority levels truthfully. We demonstrate positive incentive-compatibility and efficiency results for Button a in a simplified setting through simulations.

*We are grateful for the feedback and advice we received from Professor Tim Roughgarden and the CS269I staff. We would like to thank Dr. Scott Kominers for his suggestion to consider the Button mechanism.

[1]See `http://inrix.com/economic-environment-cost-congestion/`.

The rest of the paper is organized as follows: Section 2 sets up the model of road network, describes our cars' kinetics, and explains the terminology of protocols resolving conflicts and mechanisms for assigning prioirties. Section 1 defines the GREEDY protocol and the BUTTON mechanism and gives positive empirical results. Section 4 discusses limitations of the protocol and mechanism implemented. Section 5 describes directions for future work.

2 Model of the Travel Problem

We start by describing the setup of the "travel problem."

2.1 Networks, roads, and intersections

Cars $1, \ldots, C$ drive on a network of nodes and directed edges. Each node in the network is either a "road" or an "intersection." Roads are nodes with degree 1 or 2, and intersections are nodes with degree greater than 2.

2.2 Trips

Each car makes N trips. Each trip is a fixed path in the network, i.e., a sequence of nodes in which every consecutive pair of nodes is connected by a properly oriented edge. We are not considering the problem of choosing optimal routes, so the route of each trip is fixed and cannot be changed mid-trip.

2.3 Moving through the network

At each time step $t = 0, 1, 2, \ldots$, each car advances one node in its trip path, if possible. A road may be occupied by an arbitrarily long queue of cars,[2] but an intersection may only have one car at a time. Say that a car is "eligible to advance" if the next node in its path is an intersection, and the car is at the front of the queue at its current road. (If the next node in a car's path is a road, it will always be able to advance, since we assume roads have unlimited capacity.)

2.4 Priorities and costs

At the start of each trip, every car is high-priority with probability $p > 0$, independent of all other cars and priorities in previous rounds, and low-priority with probability $1 - p$. Each time step that a car is on the road, it incurs a cost of 1 if low-priority and $H > 1$ if high-priority.

2.5 Protocols and mechanisms

When multiple cars are eligible to advance to the same intersection, there is a "conflict." A "protocol" is a systematic way to settle conflicts using information about where cars are located in the network and cars' priority levels.

[2] Hence in our model, traffic from one intersection never backs up another intersection.

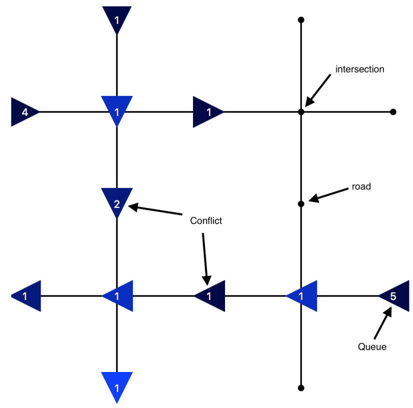

Figure 1: A basic network of cars

For example, we may informally define a stop sign protocol: Whenever there is a conflict, the stop sign protocol allows which car reached the front of its queue first to advance (and if multiple cars reached the front of the queue simultaneously, the car "on the right" advances first). The dependence of the stop sign protocol on priorities levels is "trivial," i.e., there is no dependence.

Information about each car's priority level is private. Therefore, if a protocol depends on priority levels non-trivially, the protocol must implement a "mechanism," i.e., a function that translates signals from cars into an assignment of priority levels.

Given a choice of protocol and mechanism (if the protocol depends non-trivially on priority levels), we can instantiate an arbitrary network with cars and consider the cost incurred by all cars driving through the network over the N trips. The goal of the paper is to explore how to choose a protocol and mechanism to minimize this total cost.

3 Applying Mechanism Design to the Travel Problem

3.1 A simple example

Consider a simple instance of the travel problem described in Section 2: 2 roads each with 1 car, 1 intersection. It is obvious that a cost-minimizing protocol must allow the high-priority car to advance before the low-priority car, whenever the two cars have different priority levels.

Even though the choice protocol is clear, it is unclear how to solicit information from cars, since all cars have an incentive to signal that they are high-priority. We need to design a mechanisms that incentivizes cars to signal their priorities truthfully.

3.2 Mechanisms, signals, and priorities

We define a mechanism M formally as a function that maps signal data (one signal per car per trip for all trips up to trip n) to an assignment of priority levels $\mathcal{P} \in \{l, h\}^C$, where C is the number of cars.

Algorithm 1 A mechanism maps car signals to car priorities

$\mathcal{S} \leftarrow [(, \ldots,)] \times N$
function GetPriorities(Mechanism M, Trip n)
  for each car $c = 1, \ldots, C$ **do**
    Elicit signal σ from c.
    $\mathcal{S}[n](c) \leftarrow \sigma$
  return Priorities output by mechanism $M(\mathcal{S})$

3.3 The Greedy protocol

Suppose that for some trip, we use some mechanism M to assign cars $1, \ldots, C$ priorities levels $\mathcal{P} \in \{l, h\}^C$. Now, we can resolve conflicts between competing eligible cars greedily, i.e., by selecting the front car from the road with the highest total priority (to be defined more precisely later in this section).

Before we define the greedy protocol (see Algorithm 2), we need some notation. Let I be the set of all intersections, and for intersection i, let $R(i)$ be the set of all road nodes r with a directed node from r to i. Let $l(r, t, \mathcal{P})$ denote the number of low-priority cars at road r at time t, and let $h(r, t, \mathcal{P})$ denote the number of high-priority cars, according to the priorities $\mathcal{P}$.

Algorithm 2 A greedy protocol for settling intersection conflicts

---

procedure Greedy(Mechanism M)
  for Trip $n = 1, \ldots, N$ **do**
    $t \leftarrow 0$
    $\mathcal{P} \leftarrow$ GetPriorities(M, n)
    while Some cars have not yet reached destination **do**
      for $i \in I$ **do**
        for $r \in R(i)$ **do**
          Calculate total cost of cars $c(r) \leftarrow l(r, t, \mathcal{P}) + H * h(r, t, \mathcal{P})$
        Let $\hat{R} \leftarrow \arg\max_{r \in R(i)} \{c(r)\}$ be the set of roads with highest total cost
        if $|\hat{R}| = 1$ **then**
          Pick unique road and let car at front of queue advance
        else
          Pick one road in $\hat{R}$ uniformly at random and let front car advance
      $t \leftarrow t + 1$

3.4 The Button Mechanism: Using wait period to elicit truthful signals

Our goal is to design a mechanism that will assign priorities that are close to cars' true priority. As a benchmark for our analysis, we define the ideal mechanism:

Algorithm 3 The ideal, mind-reading mechanism

```
function IDEAL(Signals S)
    return True priorities of each car, independent of signals S
```

We now work to define a mechanism that comes close to IDEAL. Let the signal space Σ consist of two signals, low and high, i.e., $\Sigma = \{l, h\}$. When we elicit signals, we simply ask a car to report its priority. In order to prevent cars from reporting h every trip, we design a mechanism BUTTON that uses wait-periods: Whenever car c signals high-priority, BUTTON will assign c low-priority for the next T trips, regardless of c's signal. See Algorithm 4 below.

Algorithm 4 A mechanism that uses a wait period

```
S ← [(, . . . , )] × N
function BUTTON(trip n)
    P ← (, . . . , )
    for each car c = 1, . . . , C do
        if S[n](c) = l or c was assigned h for any of the previous T trips then
            P(c) ← l
        else
            P(c) ← h
    return P
```

We call the above mechanism BUTTON because we imagine that it could be implemented in the following way: There is a button in the car that the driver can press to guarantee himself high-priority treatment, but he cannot press the button more than once in any window of T trips. We set $T = \lfloor 1/p \rfloor - 1$, so that if the driver presses the button as often as possible (and if $\lfloor 1/p \rfloor = 1/p$), he will press it p-fraction of the time.

Now that we have defined a mechanism, it is time to consider strategies for a car c in the travel problem.

Assumption 1. *We make the following assumptions to simplify our analysis:*

1. *For car c's nth trip; the routes, priorities, and previous signals of the other cars on trip n are independent of c's signals in the previous $n-1$ rounds, e.g., car c meets a new set of cars each time it is on the road.*
2. *Car c travels on the same road network and takes the same path in the network every trip.*

The random process that describe c's trips is Markovian, and by Assumption 1, the only information relevant to car c's decision about how to signal at trip n is c's priority and wait-time for trip n. Call the (priority, wait-time) pair a "state", denoted (π, ℓ).

A "policy" Π is a map from states to signals $\{(\pi, \ell) : \pi = l, h; \ell = 0, 1, \ldots, T\} \to \{l, h\}$. Note that the value $\Pi(\pi, \ell)$ for $\ell > 0$ is irrelevant because of the definition of BUTTON.

We have overlooked one detail: The number of the state n might also be relevant to c. E.g., if the game is about to end and c has nothing to lose, c might think differently about whether to signal h. To handle this formally, we should introduce time discounting. However, for this draft, we will ignore the issue and take N sufficiently high so that the effect of "end of game" reasoning is negligible.

We make one more key assumption for our initial analysis:

Assumption 2. *The trip for car c using the* GREEDY *protocol is always at least as short if c is assigned high-priority compared to if c is assigned low-priority (when all other cars' trips and priorities are fixed), and sometimes strictly shorter.*

Thus, in the scope of a single trip, a car always prefers to be assigned high-priority. Assumption 2 is certainly true when there is 1 intersection and 2 cars, as in Section 3.1.

What policy is optimal for c? Suppose the paths and priorities of other cars are drawn (independently each trip) from some distribution, which may or may not be known to c. Other cars follows some policy to give signals to BUTTON. Before the start of a trip n, we can consider the expected duration of c's trip, $E_c(\rho)$, over the resolution of other cars' data and the randomness in the GREEDY protocol, given that c is assigned priority ρ by BUTTON. By Assumption 2, $E_c(h) < E_c(l)$.

We define an "optimal" policy Π as a policy that minimizes expected cost per trip:

$$A(\Pi) := \frac{1}{N} \sum_{n=1}^{N} \mathbf{E}(\text{Cost of trip } n; \Pi)$$

Here the expectation is over the resolution of other cars' data, the randomness in the GREEDY protocol, and the randomness of car c's priorities $\{\pi_n\}_{n=1}^N$. Because other cars' data and the randomness of GREEDY are independent of $\{\pi_n\}_{n=1}^N$, we can write

$$A(\Pi) = \frac{1}{N} \sum_{n=1}^{N} \sum_{\pi, \rho \in \{l,h\}} \mathbf{P}(\pi_n = \pi, \rho_n = \rho; \Pi) E_c(\rho) K(\pi), \tag{1}$$

where $E_c(\rho)$ is as defined above, and where $K(l) = 1$ and $K(h) = H$ are the costs of low and high priority cars. Here we have averaged over the resolution of other cars' data and the randomness in GREEDY.

Claim 3. *If Π is an optimal policy, then $\Pi(h, 0) = h$.*

Proof Sketch. Observe, the inner sum of (1) consists of 4 terms. By Assumption 2 and the fact that $C(l) < C(h)$, we have the following 2 inequalities:

$$E_c(h)C(h) < E_c(l)C(h) \tag{2}$$

$$E_c(l)C(l) < E_c(l)C(h) \tag{3}$$

In order to reason about the change in $A(\Pi)$, we consider how changing the policy Π increases or decreases $\mathbf{P}(\pi_n = \pi, \rho_n = \rho; \Pi)$ for different values of π and ρ.

Proceed in the following order:

1. Show that optimal policy Π must set some $\Pi(\pi, 0) = h$ for some π. Otherwise $\mathbf{P}(\pi_n = \pi, \rho_n = h; \Pi) = 0$ for both $\pi = l, h$. We can definitely increase $\mathbf{P}(\pi_n = l, \rho_n = h; \tilde{\Pi})$ by setting Π to signal h when $\pi = l$, and then we increase $\mathbf{P}(\pi_n = l, \rho_n = l; \tilde{\Pi})$ and leave the other two probabilities unchanged. By inequality (3), we have decreased total cost.

2. It remains to rule out the policy that signals high only at state $(l, 0)$. Use inequality (2) with a similar argument as in the first step to show that switching to policy Π that signals at both $(l, 0)$ and $(h, 0)$ decreases total cost.

□

Corollary 4. *The following are the only two possible optimal policies:*

1. ***"Truthful:"*** $\Pi_1(h, 0) = h$ *and* $\Pi_1(\pi, \ell) = l$ *for all other* (π, ℓ)
2. ***"Aggressive:"*** $\Pi_2(\pi, 0) = h$ *and* $\Pi_2(\pi, \ell) = l$ *for if* $\ell > 0$*, regardless of* π*.*

Policy Π_1 is close to "truthful" in the sense that car c signals high-priority only when it is truly on a high-priority trip and as often as possible (when it is high-priority), subject to the constraint that it cannot signal high-priority during the wait-period. Policy Π_2 is "aggressive" in the sense that is signals high priority as soon as the wait-period is over, regardless of the car's true priority.

If all cars use Truthful, then the priorities assigned to cars by BUTTON should be close to cars' true priorities. Then we are hopeful that, if GREEDY performs well given true priorities, we can achieve low total cost (for all cars) using GREEDY with BUTTON. First, however, we need to prove that it is in the best interest of cars to use the Truthful policy.

3.5 Testing the dominance of the Button Mechanism

In this subsection, we show empirically that BUTTON is a dominant strategy. That is, we demonstrate that under some assumptions about the set up of the road network and the priority structure, the Truthful policy from Corollary 4 is optimal, regardless of whether other cars are playing Truthful or Aggressive.

We construct a network with a single intersection and $C = 10$ cars. We set the probability of being high priority $p = .3$. We follow a single car c for $N = 10000$ trips. First we assume all other cars use the Truthful policy, and then we calculate the average cost incurred by c when c uses the Truthful policy and the Aggressive policy. Next we assume all other cars use the Aggressive policy, and we again compare the average cost incurred by c when c uses the Truthful policy and the Aggressive policy. We repeat the experiment over various values of high cost H in the range $[2, 10]$.

The results (first plot in Figure 3.5) show that regardless of what policy other cars use, the single car c is better off using the Truthful policy. The figure also illustrates that the spread between Truthful and Aggressive increases as H increases. The intuition behind the increasing spread is that, as H increases, it is relatively less beneficial to be aggressive and signal high when c is low-priority.

The dominance of Truthful is not a special case in the choice of parameters described above. Rather, at least in the case of a single intersection, the results holds more generally. The second plot in Figure 3.5 shows that as we vary the probability of

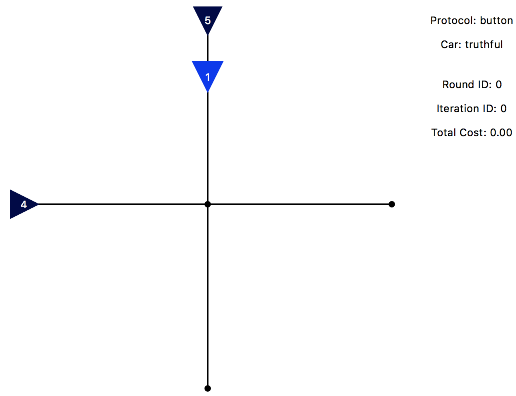

Figure 2: A single-intersection network with many cars.

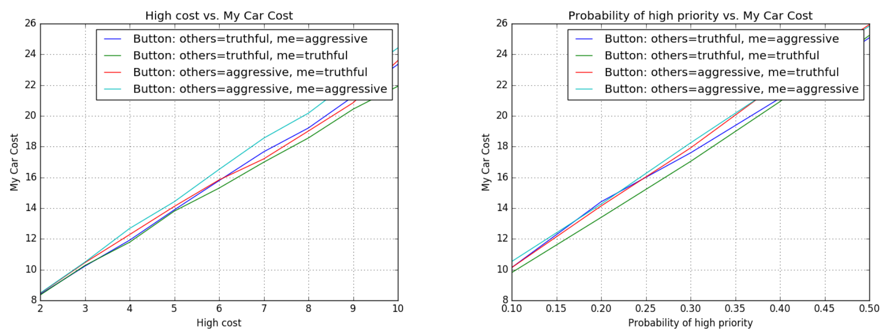

Figure 3: Truthful is dominant, and the dominance is robust to changes in p.

high priority p in $[.1, .5]$, Truthful remains dominant. Of course, after $p > .5$, the wait-period $T(p) = 0$, so the BUTTON is degenerate.

3.6 Performance in single-intersection networks

Now that we have shown that BUTTON is incentive compatible, we can inquire about the total societal cost of the GREEDY protocol using BUTTON under the assumption that all cars play Truthful.

GREEDY(BUTTON) performs almost as well as GREEDY(IDEAL)[3] in single intersection networks. As expected, it performs much better than RANDOM protocol, which chooses one car in each conflict randomly. In fact, GREEDY (BUTTON) is much closer to the "oracle" protocol GREEDY(IDEAL) than the "baseline" RANDOM. See Figure 4.

[3] Called Optimal Greedy in the figures.

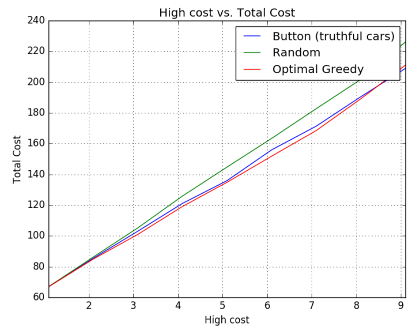

Figure 4: GREEDY runs almost as well as possible when implemented with BUTTON and cars use the Truthful policy.

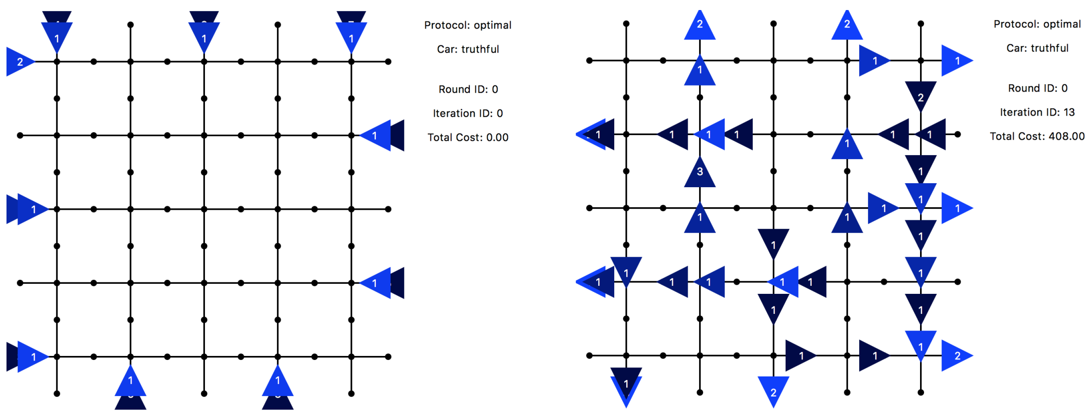

Figure 5: A network with many cars and intersections.

4 Limitations of the Greedy protocol and the Button Mechanism

4.1 The Greedy protocol performs poorly in large networks

The goal of the BUTTON is to elicit truthful reports about cars' priorities in order to allow GREEDY to run with accurate information. But it when the number of roads and cars in a network grows large, even with perfectly accurate information, GREEDY performs worse.

See Figure 7 for some intuition: GREEDY allows the high-priority cars going south to advance before the low-priority cars going east. The eastbound low-priority cars are forced to wait, even though it would be a Pareto improvement for some of them to advance first, since the southbound cars come into conflict with the westbound cars at the following intersection. RANDOM, on average, lets half of the low-priority eastbound cars advance.

GREEDY is good at resolving local conflicts and performs well in networks with a single intersection or big networks with few cars. In such examples, the "downstream"

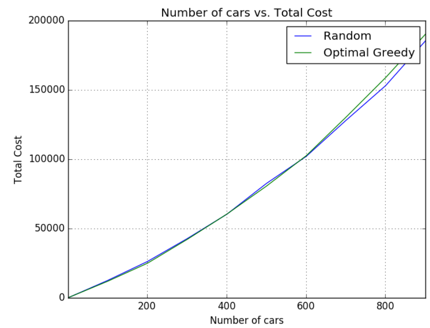

Figure 6: Random outperforms Greedy when the number of cars in the multi-intersection network grows large.

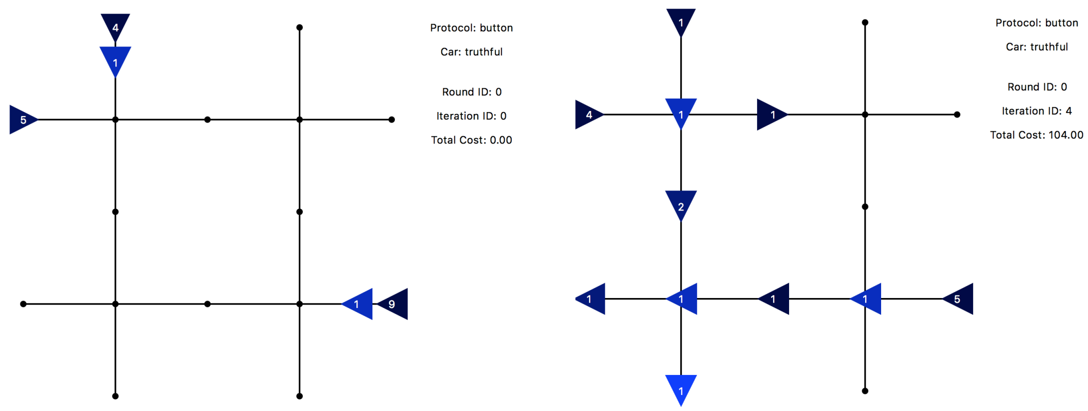

Figure 7: Downstream effects.

effects of local decisions are limited. However, in a big network with many intersections and many cars, there are many downstream effects, and Greedy may perform no better or worse than Random.

4.2 Driving around to game the system

In order to escape the wait-period, a driver may want to make aimless trips. Aimless driving will create more traffic and hurt overall efficiency.

As long as cars require drivers and the energy necessary for driving a car is expensive, the aimless driving attack will not be attractive for most drivers. However, if cars were self-driving and the cost of fueling cars were inexpensive, the aimless driving attack could be a concern.

The aimless drive attack motivates an alternative mechanism for limiting the amount of times that cars can signal high-priority: It may be possible to use money (either real money or a scrip currency) to allow cars at intersections to participate in auctions. In order to signal high-priority, a car would have to have enough currency. We could charge a driving tax that prevents an aimless low-priority car from accruing currency, so that aimless driving is no longer useful.

We considered trying to extend the theory of the VCG mechanism to implement

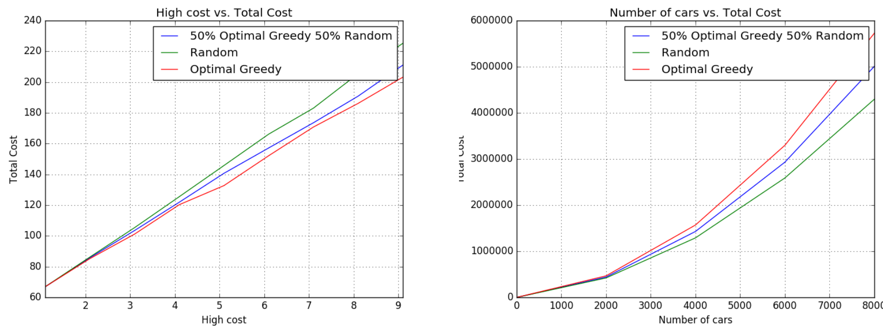

Figure 8: Mixing Greedy and Random.

truthful auctions. Implementing and analyzing VCG proved challenging for several reasons. Because of the dynamic nature of the problem (new cars arriving and leaving), calculating the externality of a car is far from straightforward. In addition, using scrip currency, the value of which is endogenous to the repeated game, breaks the assumption of quasi-linearity of money that is crucial to the truthfulness of the VCG mechanism in a one-shot setting.

5 Extensions and Future Work

5.1 Mixing Greedy and Random

Whereas in Section 3.6, we saw that Greedy outperformed Random in single intersection networks; in Section 4.1, we discovered that Random does better in networks with many intersections and cars. These discoveries motivate combining the two protocols to get a protocol that is more robust to a wider variety of networks, i.e., by resolving conflicts randomly sometimes and greedily other times. Figure 8 demonstrates that the combined protocol performs nearly as well as Greedy in the single intersection network and does better than Greedy in large networks as the number of cars increases.

5.2 Optimal number of Roads

We observe a curious relation between the number of roads and the total cost of a network: When all other parameters, including the number of cars, are fixed, and the number of roads increases, the total cost at first decreases sharply before leveling off and then increasing. See Figure 9.

Note that the decrease in total cost, no doubt due to less congestion, occurs despite an increase in the total distance of travel for each car on each trip. The question of finding the optimal number of roads seems to motivate a problems in urban planning, i.e., find the radius of a city that minimizes cost of travel, given a fixed target population.

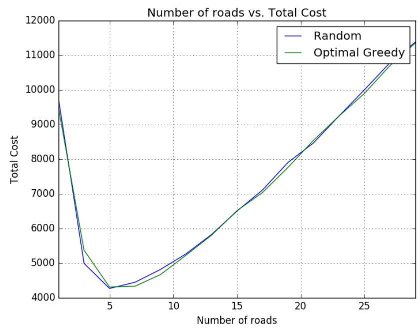

Figure 9: Varying the number of roads.

5.3 Next steps

Our next step is to address the failure of GREEDY in congested, multi-intersection networks. We need to select an effective polynomial-time protocol that approximates optimal resolution of conflicts in the network in place of GREEDY, and then we will investigate whether we can get positive results when we implement the selected protocol with BUTTON.

References

[MHM15] James J Mulcahy, Shihong Huang, and Imad Mahgoub. Autonomic computing and vanet. In *SoutheastCon 2015*, pages 1–7. IEEE, 2015.