Congestion games; empirical inefficiency of a decentralized approach

Tanner Sundwall, CS 412 Fall 2022, Brigham Young University

I. Introduction

In analyzing a system involving actions at the agent level, it is important to identify who is making the decisions. In congestion games, agents typically make allocations on their own behalf, according to their own utility functions. An example? Motorists deciding which paths to take to their destination in the face of traffic.

But what if this decision-making was centralized? The below analysis attempts to estimate how much more efficient such a system would become under a centralized approach. The objective of this research is to estimate the empirical inefficiencies over several perturbations of a simple congestion game.

II. Congestion Games

Congestion games are a subclass of game theory first proposed in 1973. Like most game theory applications, agents are faced with a set of potential actions, and are trying to maximize their payoff.

Congestion games are typically structured around a directed graph, where each agent is trying to travel from a fixed source node to a fixed destination node.

Congestion games tend to hold the following properties:

- Each edge has an associated loss function. When more agents use a given edge, it becomes more costly to use. Loss functions are considered independent of each other. In our analysis, we assume loss functions to be linear, and characterized solely with a ”loss scalar”.
- The ”notion of best” for such games tends to be the lowest cost; therefore we will be working with minimization problems.

A. Literature

Much of the research surrounding congestion games has revolved around a few axioms:

- A Nash equilibrium can be found for any congestion game [2]
- This equilibrium can be converged upon easily [2]
- Utility functions are linear combinations of cost functions [2]
- The price of anarchy (PoA) of a system indicates the penalty of selfish behavior. It is defined as the worst-case game with respect to the best-case (centralized) game. [4]

Mathematically, the ”socially optimal allocation” (or the centralized approach as we defined it) is ”obtained by maximizing the sum of the utility functions, subject to the constraint that the sum of the amounts given to each user is less than or equal to the supply” [1]. In our case, it is the *minimum* sum of utility functions.

We assume an infinite supply of resource; that is, edge load is not explicitly bound. But the loss functions we implement give way to constraint and scarcity.

B. Domains and Use Cases

Congestion games have applications across many domains. Most notably, congestion games are helpful when understanding user behavior in various networks. Such networks range from load balancing/request management, social/dating networks, internet service distribution, and physical road networks.

Other, much simpler, examples could also fit this definition; say, an individual deciding whether or not to visit a bar [2].

Theoretically, any system or process where an agent’s payoff depends on the load of a resource could be formulated as a congestion game.

While this analysis isn’t specific to any of these domains, the language and interpretations are within scope of a road network; where motorists attempt to select the best route in the face of potential traffic.

III. Price of Anarchy

The Price of Anarchy measures the cost of ”selfish routing”, and is conceptually identical to what we are trying to measure in our analysis.

Mathematically, the Price of Anarchy is defined as:

$$PoA = \min_{G} \frac{f(a_{eq})}{\max\limits_{a} f(a)}$$

Where G is the set of all possible congestion games in a given domain. $a_{eq} \in EQ(G)$; that is a_{eq} is some equilibrium in the set of all equilibria in G. In essence, it takes the ratio of the worst possible eq game $\in G$ and the opt game across A.

The Price of Anarchy differs depending on the game theory domain. What is the relevant metric for *congestion games*?

In [1], Sara Robinson summarized the findings of Tardos and Roughgarden regarding PoA minimum bounds: ”The general cost of a Nash flow is at most 4/3 of the minimum-latency flow for graphs with linear latency functions".

This is an important benchmark for our analysis; we shouldn't expect to see any cases where the *opt* cost is less than 3/4 of the *eq* cost.

IV. Terminology

$A = \{a_1, ..., a_n\}$ is the set of agents in the problems space.

$\mathcal{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ is the set of action profiles. Profile $\mathcal{A}_j$ corresponds to agent a_j.

The set of edges in the directed graph is denoted by $E = \{e_1, ..., e_m\}$. The set of nodes is denoted by $N = \{n_1, ..., n_k\}$. The function $\mathcal{C}(e_i) = E_i$ is the cost of edge e_i.

The loss function of edge j is defined by $\lambda_j = (1 + L_j C_j)$, where L_j is the loss scalar for edge j and C_j is the current volume of agents using edge j.

A given directed graph has a finite number of valid paths. A valid path is defined as a non-cyclic path from the source node to the destination node. $P = \{p_1, ..., p_r\}$ is the set of valid paths. The number of valid paths, $|P|$, is $\approx b^d$, where b is the branching factor of the graph, and n is the depth. This estimation metric assumes that all leaf nodes at level d connect with the destination node. It also assumes branching factors and sparsity patterns remain consistent. A path going through nodes 1, 2, and 6 would be denoted as $p_t = x_1 x_2 x_6$. It's easy to see that this term will only be equal 1 if all terms have value 1. This will be an important property in our problem formulation.

The term *agents* is used to define the individuals involved in the system; in the case of a road network, these would be the drivers. *Edges* define the roads between *Nodes* (intersections) that agents can take.

Each agent has their own utility function, $\mathcal{U}$. This function is assumed discrete, where each edge has an assigned utility. The utility for edge j of agent i is denoted as $u_{ij} \in \mathbb{R}$. This creates a utility matrix $U = \mathbb{R}^{nxm}$.

It is important to note that our minimization problem dictates these utilities to be inversed. A lower "utility" suggests that an agent is more likely to take a path. A higher "utility" is therefore indicated by a lower value.

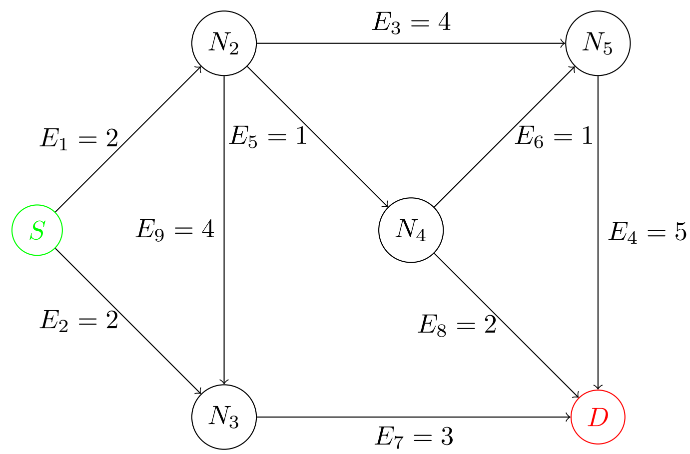
V. Limitations/Caveats

Some great simplifications were made in this analysis:

- All agents are assumed to have the same source and destination nodes (denoted as N_s and N_d).
- All agents are assumed to have the same weights; that is, each edge is burdened by any given agent equally.
- The loss function for edge j, λ_j, is assumed linear.
- Utility functions are assumed linear

VI. Scope

We will be analyzing a road network with 6 nodes, 9 edges, and between 4-10 agents. The utility values u_{ij} are assigned randomly (between 0.5 and 1.5) at each iteration. Since we are minimizing the objective function, a lower utility indicates a higher incentive to use an edge, as previously clarified. Below is a visualization of the directed graph for our domain:



In our domain, the loss scalars corresponding to the edges are fixed as follows:

$$L_j = \{1, 2, 3, 4, 1, 2, 3, 4, 2\}$$

It's important to note that the graph structure, edge values, and loss scalars are all arbitrary and do not reflect any true network. It was constructed solely for the purpose of aiding our analysis.

The problem size, sparsity, etc. were dictated by the computational limitations of the analysis. As discussed later, this problem has a large complexity.

VII. Decentralized Game

The decentralized congestion game is fairly simple. It is defined by the set of agents A, loss functions λ, utilities $\mathcal{U}$, and the set of action vectors $\mathcal{A}$. An action set for agent i must satisfy a complete path. An action set for agent i may look like:

$\mathcal{A}_i = \{1, 0, 1, 1, 0, 0, 0, 0, 0\}$

Where each entry indicates whether the agent uses that edge, and $\mathcal{A}_i \in P$.

Each agent attempts to minimize their own independent objective function. For agent 1:

$$\underset{x}{\text{minimize}} \sum_{j=1}^{m} (1 + L_j C_j) E_j u_{1j} x_{1j}$$
$$\text{subject to} \sum_{t=1}^{r} p_t = 1, \ t = 1, \ldots, r,$$

x_{ij} is a binary value indicating whether agent i used edge j. Since each edge is evaluated for each agent, there will be mn decision variables across all objective functions.

In our domain, each agent must take one of the following routes $\in P$, as dictated by the directed graph:

$$p_1 = x_1 x_3 x_4$$
$$p_2 = x_1 x_5 x_6 x_4$$
$$p_3 = x_2 x_7$$
$$p_4 = x_1 x_5 x_8$$
$$p_5 = x_1 x_9 x_7$$

The path constraints indicate that one and only one of these valid paths should be taken by each agent. A path only takes on value 1 when all decision variables in that path = 1.

This program assumes a fixed set of actions from the other agents (C_j term is fixed), making it a fairly simple problem. While we won't actually be solving this program, it sets up the understanding and motivation for finding equilibrium among all agents.

A. Equilibrium

Per [2], a congestion game can always reach Nash equilibrium. A key assumption is that agents make decisions one at a time, not concurrently. This is a necessary condition, since the C_j term in the program is assumed static.

To find equilibrium:

- 1) start with an arbitrary assignment of valid paths
- 2) evaluate some a_i (ordered arbitrarily)
- 3) if it's not the best option given a_{-i}, find the best option
- 4) loop through agents until no more changes should be made

This process, although contingent on the initial assignments, produces a set of equilibrium actions. Step 3 essentially solves instances of the generalized program formulated above.

The value of the action sets (total cost) can then be derived. If $\mathcal{L}_{i_{eq}}$ denotes the objective value at equilibrium for agent i, the total cost of the network is denoted by:

$$\mathcal{L}_{eq} = \sum_{i=1}^{n} \mathcal{L}_{i_{eq}}$$

VIII. Centralized Approach

In a centralized approach, the C_j terms in the decentralized objective functions become functions of other agent choices:

$$C_j = \sum_{i=1}^{n-1} x_{-ij}$$

Where $-i$ denotes all but the ith agent. The key distinction between the decentralized congestion game and our centralized analysis is the reduction to a single optimization problem in the centralized case.

Decision variables are identical to those in the congestion game objective functions, but in the structural form of the following single program, whose objective value is denoted as $\mathcal{L}_{opt}$:

$$\begin{aligned} \underset{x}{\text{minimize}} & \sum_{i=1}^{n}\sum_{j=1}^{m}(1 + L_j \sum_{i=1}^{n-1} x_{-ij}) u_{ij} E_j x_{ij} \\ \text{subject to} & \sum_{t=1}^{r} p_{ti} = 1,\ t = 1, \ldots, r, \\ & \vdots \\ & \sum_{t=1}^{r} p_{tn} = 1,\ t = 1, \ldots, r, \end{aligned}$$

where p_{ti} is a valid path $\in P$ for agent i. Constraints are stacked, each agent needing a single valid path. In comparison to the decentralized programs, our new objective function has the additional congestion sum outlined above, and n sums of utility.

A. A greedy solution and the motivation behind utility

The solution to this program is obvious for single-agent problems; take the route with the shortest total costs, loss functions aside. But for more congested systems, it's much more difficult. Hypothetically, the routes could be ranked by initial cost, and agents could be routed until the next route becomes more viable than the current. A greedy solution may work if the utility functions of each agent are the same; but we assumed a matrix $\mathcal{U}$ where entries are not homogeneous. In this case, coefficients can be added to edge to decisions that compete with a greedy solution, and the actual assignments of agent to path actually matter.

An example: agent a_5 may need to stop at N_4 for groceries; instead of simply adding this as a constraint, let's say that incoming edges to N_4 (just E_5 in this case) will receive extra weight. Therefore, the term μ_{54} will be very small, since the added benefit of taking this path will manifest in a lower value (this is a minimization problem).

B. Program Complexity

As graph size increases (or sparsity decreases), the size of the program increases. For a fixed number of agents, the program will be of size m, since each agent will have m decision variables. If agent size can vary, the objective function has nm variables, and nm^2 terms. The added complexity for the number or terms comes from the x_{-ij} terms that determine the congestion of an edge. Note that these aren't distinct variables, but variables that already appear elsewhere in the objective function. The complexity in the constraints is pretty straightforward; the number of constraints = $|A|$.

IX. Computational Obstacles of Centralized Approach

Our objective function and corresponding constraints take a fairly complex form; the decision variables can only take binary values, and therefore our program is a zero-one integer program. The objective function and constraints are non-linear (due to the products of decision variables), adding to an already NP-hard problem.

The *cvxpy* Python library does not support non-linear objective functions for Mixed Integer Nonlinear Programming Problems (MINLPs). Therefore, in the empirical analysis, the program formulated above was not solved. A more brute-force solution was adopted; check every possible solution to the problem, and take the best one.

This essentially evaluates every valid permutation. The complexity is problematic, however; there are $|P|^{|N|}$ potential solutions. This complexity limited the problem sizes in our analyses (10 agents or less).

X. Methods

Our empirical analysis attempted to observe the disparity of cost between the eq and opt solutions that correspond to the decentralized and centralized approaches.

Also included are the costs for a random assignment of actions ($rand$) and a naive assignment of actions ($naive$). The random assignment uses the same stochastic process that was used to assign the initial arbitrary action profiles when starting the equilibrium process. The naive assignment assumes every agent will take the shortest path regardless of loss functions or congestion.

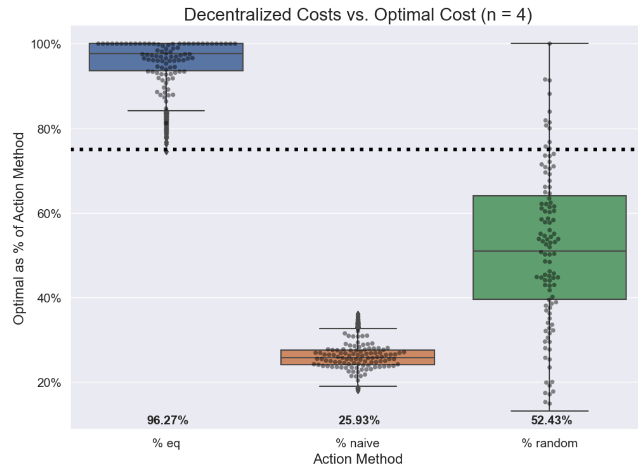
A python script simulated 10000+ problem instances across 6 problem sizes (4-10 agents). On each iteration, utilities were randomized, but the graph structure/sparsity remained the same, as well as the edge weights and loss scalars.

$naive$ is expected to be consistently the most inefficient option, whose performance is dependent on the loss scalar along the initial best path. $random$ shouldn't perform too well either, but should have higher variance, leading to an efficient results every so often.

The relationship between eq and opt is obviously at the center of attention, and should have the least disparity on average.

XI. Results

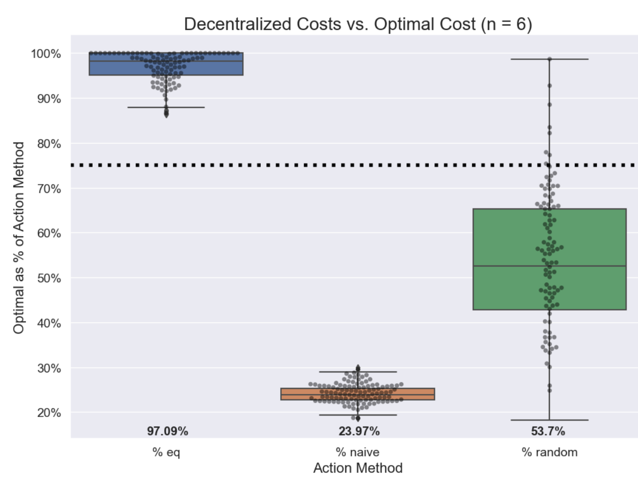
Included are three box-and-whisker plots across three problem sizes ($n = 4, 6, 9$). The y-axis indicates the opt cost as a percent of the cost of the other methods. Bars indicate the mean value, while dots indicate specific observations (simulations). The dotted black line indicates the theoretical PoA lower bound we touched on earlier.



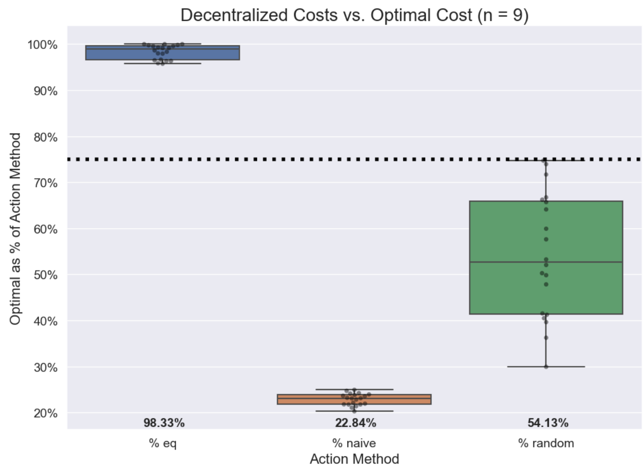
The % eq box-and-whisker plot indicates the cost inefficiency of an equilibrium congestion game. For our $n = 4$ observations, the mean % opt over eq is 96.27%. This indicates that on average, the centralized method produces a solution with 3.73% lower cost than a decentralized method. The large tail of the distribution for the eq observations indicate lots of "worst case" simulations. Conditions for lower bound inefficiency seem rare according to this distribution, and very few near our theoretical PoA.

35% of simulations saw an optimal cost of 100%; meaning that the opt solution and eq solution were equivalent.

Should we expect the amount of simulations with zero penalty to decrease as n increases? We will explore this a bit later.

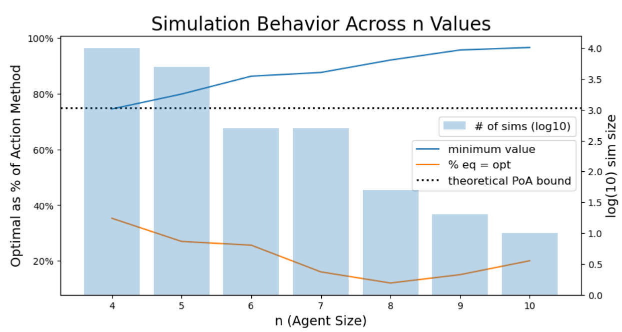


Our $n = 6$ plot shows similar results for random and naive strategies, but the tail end of the eq distribution has shrunk. Not a single simulation crossed below 85%, and the average efficiency increased to 96.33%.



Let's take a look at one last problem size, $n = 9$. While much fewer simulations were ran at this size, the eq distribution has tightened even further. Not a single observation lies below 95%, and the average nears closer to the opt value.

How does this trend manifest across all problem sizes? The blue line in the below graph shows how the minimum efficiency simulation increased as the number of agents in the simulation increased. This is a very important finding. Given our utility behavior and fixed graph structure, do the inefficiencies of a decentralized congestion system become trivial as n increases?



This chart also includes the proportion of simulations in which no inefficiencies were observed in the decentralized game. Given our previous observation of the increasing minimum values over n, it's surprising to see this proportion seemingly decrease as n increases. We will be careful to call this an inverse relationship, but we can surely say it deviates from the trend of the blue line.

A. *Properties of worst-case behavior*

What properties do lower performing simulations hold? And does this help explain differences across problem sizes, considering the rise in minimum values explained above?

Variation in results could stem from two areas: properties of the utility matrix, and properties of the initial conditions (these are the only stochastic processes in our simulations). The latter was immediately eliminated as a potential source of variation. The utility matrix was modified so that it was uniform and deterministic across simulations; and $\%eq$ values stayed constant. We can therefore conclude that our algorithm to find a decentralized equilibrium finds a unique solution across all initial states of a given problem instance.

So what properties of the utilities matrix cause some eq solutions to be so inefficient? This remains unclear. Through basic examinations of scatterplots, many potential properties could be ruled out. That is, these properties didn't appear to have a strong relationship with $\%eq$.

Such properties that were ruled out include analysis of in-group variances (variance of utilities across agents on a given edge), overall utility variance, overall utility mean, and overall utility max/min values.

To explain the improvement of worst-case games for larger problem sizes (and worst-case behavior in general), these properties must be identified.

XII. Conclusions

It is apparent that in our limited domain and scope, a centralized, universally optimal solution performs better than one with individual objective functions inherent to a congestion game. Our findings also follow suit with the theoretical price of anarchy we identified; worst-case games appeared to be bounded by the 3/4 inefficiency limit suggested.

What remains unclear, however, is the affect of agent size on network inefficiency. While worst-case games appear to decrease in magnitude as n increases, the proportion of optimally efficient games appears to potentially trend the opposite.

XIII. Future Work

Which assumptions and resources are limiting or conclusions?

The linear nature of the loss functions may suggest that this analysis would disperse drivers across different paths than would truly happen in practice. This is likely amplified by assuming large loss scalars (in our example domain, $L_j > 1, \forall j$. In practice, such a loss function would likely be of an exponential structure; adding more drivers doesn't increase travel times by too much until a fairly well-defined point. At this point, travel times increase exponentially. While our model was intentionally over-simplified, an implementable solution would require an identifiable, realistic loss function for each edge.

How does changing edge costs affect the distribution of paths among travelers? This is another intriguing dynamic that could be analyzed. If there is high variance between edge costs, different results may be achieved. A similar analysis could be performed, but with variance in loss scalars (or for non-linear loss functions, any function that deviates substantially from the rest).

Given the laborious computation times of our solving mechanism, how can our analysis be scaled? Finding a stable computational solution to our $\mathcal{L}_{opt}$ program would be a good start; assuming that the complexity of such an algorithm is substantially better than our brute-force solution.

How does the interaction between values in the utilities matrix affect eq performance? Although such characteristic weren't identified in our analysis, finding these properties is vital to understanding these networks from a pragmatic standpoint.

This brings us to our most vital question; *do congestion game inefficiencies truly disappear at large n values?* This question could be explored more extensively through a combination of the above improvements and extensions.

REFERENCES

[1] S. Robinson, "The Price of Anarchy," SIAM News, Volume 37, Number 5, June 2004

[2] K. Leyton-Brown, "Congestion Games," University of British Columbia lecture, unpublished

[3] Y. Mansour, N. Yosef, A. Koren, "Lecture 6: Congestion and potential games," Spring Semester, 2003-2004, unpublished

[4] D. Grimsman, "Resource Allocation Games," Brigham Young University, unpublished

[5] "SCIP Documentation," https://www.scipopt.org, unpublished

[6] J. D. Hunter, "Matplotlib: A 2D Graphics Environment," Computing in Science Engineering, vol. 9, no. 3, pp. 90-95, 2007

[7] "Draw a Graph Using LaTeX," https://www.baeldung.com, unpublished