Defining Equitable Geographic Districts in Road Networks via Stable Matching

David Eppstein
University of California, Irvine
eppstein@uci.edu

Michael T. Goodrich
University of California, Irvine
goodrich@uci.edu

Doruk Korkmaz
University of California, Irvine
dkorkmaz@uci.edu

Nil Mamano
University of California, Irvine
nmamano@uci.edu

ABSTRACT

We introduce a novel method for defining geographic districts in road networks using stable matching. In this approach, each geographic district is defined in terms of a *center*, which identifies a location of interest, such as a post office or polling place, and all other network vertices must be labeled with the center to which they are associated. We focus on defining geographic districts that are *equitable*, in that every district has the same number of vertices and the assignment is stable in terms of geographic distance. That is, there is no unassigned vertex-center pair such that both would prefer each other over their current assignments. We solve this problem using a version of the classic stable matching problem, called *symmetric stable matching*, in which the preferences of the elements in both sets obey a certain symmetry. We show that, for a planar graph or road network with n nodes and k centers, the problem can be solved in $O(n\sqrt{n}\log n)$ time, which improves upon the $O(nk)$ runtime of using the classic Gale–Shapley stable matching algorithm when k is large. Finally, we provide experimental results on road networks for these algorithms and a heuristic algorithm that performs better than the Gale–Shapley algorithm for any range of values of k.

CCS CONCEPTS

• **Information systems** → **Network data models**; • **Theory of computation** → *Shortest paths*;

KEYWORDS

road networks, stable matching, geographic districting

1 INTRODUCTION

Location analysis is a classical branch of optimization in geographic information systems. It includes problems such as *political districting*, in which a territory must be divided into regions under certain requirements for fairness (avoiding unfair gerrymandered districts), geographic compactness, and equal representivity with respect to the broader population (e.g., see [15–17].)

In this work, we consider an assignment problem in which we are given a set of facility locations and must distribute the rest of the territory to those facilities. We model the geographic space of interest as a weighted, undirected graph representing a road network, the population to be assigned to facilities as the set of all vertices of the graph, and the facility locations as a subset of k chosen *center* nodes of the graph. Each center has a *quota* indicating how many nodes it should match. The desired output is an assignment of every node to a center, such that the set of assigned nodes for each center equals its quota. The use of quotas in this way allows each of the facilities to have different operational capacities in terms of how much of the population they can serve.

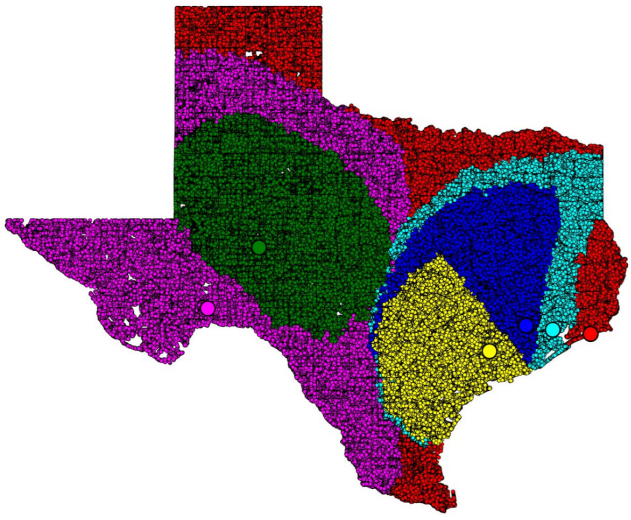

Figure 1: The solution to the *stable graph matching* problem for the 2010 road networks of Texas (2037K nodes and 2550K edges) from the DIMACS database [3]. There are $k = 6$ random centers with equal quota n/k.

We impose the conditions that each node has a preference for centers ordered by shortest-path distance from the node, and each center has a preference for nodes ordered by their distances from the center. Our goal is to match each center to its quota number of nodes and for the matching to be *stable*, meaning that no node and

SIGSPATIAL'17, November 7–10, 2017, Los Angeles Area, CA, USA

ACM ISBN 978-1-4503-5490-5/17/11…$15.00
https://doi.org/10.1145/3139958.3140015

center that are not assigned to each other prefer each other to their specified matches.

Equitable geographic districts should be "compact" to avoid the types of highly non-compact districts that have been the subject of recent legal cases involving gerrymandering. Using shortest-path distance as preferences for the stable matching results in generally compact districts, but it does not, however, imply that the districts are necessarily convex or even connected. Indeed, depending on the placement of centers and how quotas are defined, it may be necessary for some districts to be disconnected (see Figure 1, and the full version of the paper for additional examples [4]). Formally, we define the stable graph matching problem as follows:

Definition 1.1 (Stable graph matching problem). Given an undirected, weighted graph and a subset of k nodes denoted centers, find an assignment from each node to a centers such that *(i)* the same number of nodes is assigned to each center, up to round-off errors, and *(ii)* the matching is stable with respect to shortest-path distances; that is, there is no node u and center c such that u is not assigned to c, u is closer to c than to its assigned center, and c is closer to u than to one (any) of the nodes assigned to c.

1.1 New Results

In the standard stable matching problem, preferences are arbitrary. Each individual may choose as his or her preferences any permutation of the opposite-set individuals, independently of all other choices. Preferences resulting from shortest-path distances in an undirected graph are not arbitrary, however. Instead, they obey a certain symmetry property coming from the undirected nature of the graph and shortest paths within the graph. To capitalize on this idea, we define an abstract problem intermediate between stable graph matching and stable matching, which we call the *symmetric stable matching problem*.

Moreover, we develop a novel *nearest-neighbor chain algorithm* for any symmetric stable matching problem, using ideas borrowed from a very different application of nearest-neighbor chains, in hierarchical clustering problems [2, 12]. In the case of stable graph matching, this algorithm runs in time $O(n\sqrt{n}\log n)$, which improves upon the $O(nk)$ time of the classic Gale–Shapley stable matching algorithm when k is large.

Finally, we provide a heuristic circle-growing improvement to the Gale–Shapley algorithm for the case of stable graph matching.

1.2 Prior Related Work

Our notion of equitability introduces an interesting new (and more realistic) twist to Knuth's classic post office problem [13]. In the classic post office problem, one is given a collection of sites called "post offices" and one is interested in assigning nodes to their nearest post office with no consideration for quotas characterizing the capacity of each post office to handle mail. Thus, the classic post office problem is equivalent to our geographic districting problem with unbounded quotas. Knuth's discussion of the classic post office problem has given rise to a long line of research on spatial partitioning, including the important Voronoi diagrams (e.g., see [1]), which have also been extended to the graph setting [7].

The *stable matching problem* or *stable marriage problem* was introduced by Gale and Shapley [8]. This problem was originally described in terms of matching n men and n women based on each person having an ordered preference list for the members of the opposite sex in this group. In that context, stability means that no man–woman pair prefer each other to their assigned choices. Stability, defined in this way, is a necessary condition (and more important than, e.g., total utility) in order to prevent extramarital affairs. When generalized to the one-to-many case, this problem is also called the *college admission problem* [18], because it models a setting where n students are stably matched to $k < n$ colleges, each with a certain quota of admissions.

The Gale–Shapley algorithm [8]. finds a stable matching in time $O(nk)$ in the one-to-many (college admission) case, and this is optimal for arbitrary preferences. Existing research about stable matching studies several variations (e.g., [11, 14]), but the assumption that preferences are arbitrary has rarely been challenged. A first step in this direction was taken by Hoffman et al. [10], who considered the mathematical properties of a stable matching in a geometric setting, where "colleges" are points in $\mathbb{R}^2$ and "students" are all the points in $\mathbb{R}^2$, and both use distances as preferences. Eppstein *et al.* [5] extended their approach to images, where "students" and "colleges" are pixels, but their work does not extend to general graphs and road networks.

2 SYMMETRIC STABLE MATCHING

We present the symmetric stable matching problem in the one-to-many context of schools and students, and therefore all the results in this section also apply to the one-to-one case of men and women.

In order to formulate the symmetric stable matching problem, consider this alternative but equivalent definition of the stable matching problem. Each agent (school or student) gives a unique score to each agent from the other set, and ranks them in increasing order of these scores. Therefore, a set of scores such as $(a \leftarrow 7, b \leftarrow 2, c \leftarrow 10)$ corresponds to the list of preferences (b, a, c).

We call the preferences *symmetric* if the score of x for y equals the score of y for x. Moreover, in this case, we call these scores *distances*.

Definition 2.1. A stable matching problem is *symmetric* if the preferences are symmetric.

2.1 Mutual closest pair algorithm

Before introducing our nearest-neighbor chain algorithm for symmetric stable matching, we describe a simplified version of it, the mutual closest pair algorithm, which is based on the following definition and lemma:

Definition 2.2. In a stable matching problem, a *mutual closest pair* is a school and a student who have each other as first choice.

Lemma 2.3. *If preferences are symmetric, a mutual closest pair always exists.*

Due to space constraints, we defer all the proofs to the full version of the paper [4]. Note that although the pair realizing the global minimum distance are always a mutual closest pair, the reverse is not true: there can be other mutual closest pairs whose distance is not a global minimum. Moreover, Lemma 2.3 requires that the distances on which we are basing preferences be symmetric. If they

are not symmetric, as may be the case for shortest path distances in a directed graph, then there might not be any mutual closest pairs.

Algorithm 1. Mutual closest pair algorithm

Input: n students and m schools with symmetric preferences, and school quotas adding up to n.
Output: a stable matching between the students and schools.

(1) Initialize the matching empty.
(2) Repeat while there is an unmatched student:
  (a) Find a mutual closest pair s, x.
  (b) Match s and x, remove the student from the pool of unmatched students, reduce the quota of the school by one and remove it from the pool of unmatched schools if its the quota reached zero.

Due to Lemma 2.3, the algorithm will never fail to find a closest mutual pair. In the full paper we prove that the any symmetric stable matching problem has a unique solution, which will be found by any instance of Algorithm 1.

Any strategy that finds mutual closest pairs can be used in Algorithm 1. In the next section, we present one strategy for quickly finding mutual closest pairs by making use of a dynamic nearest-neighbor data structure. This data structure should be able to maintain a set of agents of the same type (students or schools) and answer queries asking for the closest one to a query agent of the opposite set. Moreover, it should support deletions, that is, allow to remove elements from the set.

2.2 Nearest-neighbor chain algorithm

The following algorithm, which we call the Nearest-neighbor chain algorithm, is based on the theory of hierarchical clustering [2, 12], and was first used in the context of stable matching (for grid-based geometric data only) in [5].

Algorithm 2. Nearest-neighbor chain algorithm

Input: n students and m schools with symmetric preferences, and school quotas adding up to n.
Output: a stable matching between the students and schools.

(1) Initialize the matching empty.
(2) Initialize a dynamic nearest-neighbor structure containing the students, and one containing the schools.
(3) Initialize an empty stack S.
(4) Repeat while there is an unmatched student:
  (a) If S is empty, add any unmatched student (or school) to it.
  (b) Let p be the agent at the top of the stack, and use the nearest-neighbor structures to find its nearest-neighbor q of the opposite set.
  (c) If q is not already in S, add it.
  (d) Otherwise, q must be the second-from-top element in S (as justified below), and p and q are a mutual closest pair. In this case, match p and q, and update the data structures accordingly: remove the student from the nearest-neighbor structure of students, reduce the quota of the school by one and remove it from the nearest-neighbor structure of schools if its quota reached zero, and remove p and q from the stack. Note that if the school was below the student in the stack and it still had positive quota, it would be added to the stack again in the next iteration, as it would still be the nearest-neighbor of the previous student in the stack. Hence, in this case, we can keep the school in the stack.

Note that the distance between consecutive elements in S only decreases. That's why, in Step (4d), q must be the second-from-top; if q was anywhere else, p would be closer to its predecessor in S than to q. Here we are using the fact that the preferences of each element are distinct. In the graph setting, we may use a tie-breaking rule to ensure that distances are unique.

Each step that adds a new element to S can be charged against a later pop operation and its associated match. Therefore, the number of repetitions is $O(n)$. This algorithm gives us the following theorem.

Theorem 2.4. *The symmetric stable matching problem can be solved in $O(n)$ query and update operations of a dynamic nearest-neighbor data structure.*

By making use of the dynamic nearest-neighbor data structure from [6] for planar graphs and road networks, the stable-graph-matching can be solved in $O(n\sqrt{n}\log n)$.

2.3 Circle-growing algorithm

As we mentioned in the introduction, the Gale–Shapley algorithm requires $O(kn)$ time to find a stable matching between n nodes and k centers. However, in the graph setting first we need to compute the preferences, that is, the shortest-path distances between every center and node. This step can be solved in $O(kn)$ time in planar graphs using the linear-time single-source shortest-path algorithm from Henzinger *et al.* from every center [9], or more generally in $O(kn\log n)$ time for sparse graphs using Dijkstra's algorithm.

However, with the following alternative algorithm, it is not necessary to compute the distances between all centers and nodes, and we can do without a separate Gale–Shapley phase of the algorithm altogether. Instead, we can use an algorithm similar to the *circle-growing method*, described by Hoffman *et al.* [10]. It can be visualized as a process in which we grow circles from each center, all at the same speed, and match each node to the first circle that grows across it.

We start k instances of Dijkstra's algorithm at the same time, one from each center. We explore, at each step, the next closest node to any of the centers, advancing one of the instances of Dijkstra's algorithm by a single step. We match each node to the center whose instance of Dijkstra's algorithm reaches it first. Note that when an instance of Dijkstra's algorithm, starting from center c, reaches a node x that has not already been matched, then c and x must be the global closest pair (omitting already matched pairs). We halt each instance of Dijkstra's algorithm as soon as its center reaches its quota. This stopping condition prevents wasted work in which an instance of Dijkstra's algorithm explores nodes farther than its farthest matched node.

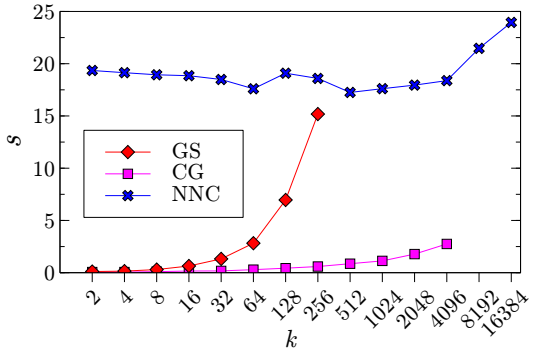

Figure 2: Runtime of the Gale-Shapley (*GS*), circle-growing (*CG*), and nearest-neighbor chain (*NNC*) algorithms in the Delaware ($n = 48812, m = 60027$) road network [3] for a range of number of centers k (in a logarithmic scale). Missing data points indicate that the computer ran out of memory.

3 EXPERIMENTS

In this section we present an empirical comparison of the algorithms. Figure 2 illustrates the main findings (see more on the full version of the paper [4]). The algorithms were implemented in Java 8 and executed on an Intel Core CPU i7-3537U 2.00GHz with 4GB of RAM, under Windows 10.

Figure 2 shows a clear picture of the respective algorithms' strengths and weaknesses:

The Gale–Shapley algorithm, with a runtime of $O(kn \log n)$, scales linearly with k. Moreover, because of the memory requirement of $\Theta(nk)$, we could not run it with large numbers of centers.

Our circle-growing algorithm was the fastest of our implemented algorithms in practice, over the range of values of k for which we could run it. For instance, on the Texas road network, which has over 2 million nodes, the algorithm finishes in 3 seconds when given 6 random centers. Additionally, the runtime of circle-growing did not appear to be strongly affected by the value of k. The reason for this is that, even though the algorithm runs k instances of Dijkstra's algorithm, the expected number of nodes that each instance explores decreases as k increases. However, this phenomenon may only be valid in expectation with randomly located centers.

The performance of the nearest-neighbor chain algorithm depends on the underlying nearest-neighbor data structure. However, in any case, the runtime and memory requirements are independent of k. That's why it is the only algorithm with a mostly flat curve in the plots, and is the only algorithm that was able to complete a solution for the entire range of values of k on all inputs that were small enough for it to run at all.

4 CONCLUSIONS

We have defined the symmetric stable matching problem, a subfamily of stable matching problems which arise naturally when preferences are determined by distances. We studied its basic properties and provided the *mutual closest pair algorithm*, which has the potential to be faster than the Gale–Shapley algorithm. Future researchers should consider the algorithms in this paper if they identify that a matching problem has symmetric preferences. As a special case of symmetric stable matching, we defined the stable graph matching problem. For this problem, we compared (a) the Gale–Shapley algorithm, (b) the mutual closest pair algorithm, and (c) the *circle-growing* algorithm, a heuristic improvement over the Gale–Shapley algorithm.

This work leaves open several questions for future research, that we list in the full version of the paper [4].

ACKNOWLEDGMENTS

This article reports on work supported by the DARPA under agreement no. AFRL FA8750-15-2-0092. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. This work was also supported in part from NSF grants 1228639, 1526631, 1217322, 1618301, and 1616248.

REFERENCES

[1] Franz Aurenhammer. 1991. Voronoi diagrams—A survey of a fundamental geometric data structure. *Comput. Surveys* 23, 3 (1991), 345–405. https://doi.org/10.1145/116873.116880

[2] Jean-Paul Benzécri. 1982. Construction d'une classification ascendante hiérarchique par la recherche en chaîne des voisins réciproques. *Les Cahiers de l'Analyse des Données* 7, 2 (1982), 209–218. http://www.numdam.org/item?id=CAD_1982__7_2_209_0

[3] Camil Demetrescu, Andrew V. Goldberg, and David S. Johnson. 2006. 9th DIMACS Implementation Challenge: Shortest Paths. (2006). http://www.dis.uniroma1.it/~challenge9/

[4] David Eppstein, Michael T. Goodrich, Doruk Korkmaz, and Nil Mamano. 2017. Defining equitable geographic districts in road networks via stable matching. Electronic preprint arXiv:1706.09593. (2017).

[5] David Eppstein, Michael T. Goodrich, and Nil Mamano. 2017. Algorithms for stable matching and clustering in a grid. In *Proc. 18th International Workshop on Combinatorial Image Analysis (IWCIA 2017), Plovdiv, Bulgaria, 2017 (Lecture Notes in Computer Science)*, Vol. 10256. Springer, Berlin, 117–131. https://doi.org/10.1007/978-3-319-59108-7_10

[6] David Eppstein, Michael T. Goodrich, and Nil Mamano. 2017. Reactive nearest-neighbor data structures for graphs. (2017). Unpublished.

[7] Martin Erwig. 2000. The graph Voronoi diagram with applications. *Networks* 36, 3 (2000), 156–163. https://doi.org/10.1002/1097-0037(200010)36:3<156::AID-NET2>3.0.CO;2-L

[8] David Gale and Lloyd S. Shapley. 1962. College admissions and the stability of marriage. *The American Mathematical Monthly* 69, 1 (1962), 9–15. https://doi.org/10.2307/2312726

[9] Monika R. Henzinger, Philip Klein, Satish Rao, and Sairam Subramanian. 1997. Faster shortest-path algorithms for planar graphs. *J. Comput. System Sci.* 55, 1 (1997), 3–23. https://doi.org/10.1006/jcss.1997.1493

[10] Christopher Hoffman, Alexander E. Holroyd, and Yuval Peres. 2006. A stable marriage of Poisson and Lebesgue. *Annals of Probability* 34, 4 (2006), 1241–1272. https://doi.org/10.1214/009117906000000098

[11] Robert W. Irving. 1994. Stable marriage and indifference. *Discrete Applied Mathematics* 48, 3 (1994), 261–272. https://doi.org/10.1016/0166-218X(92)00179-P

[12] J. Juan. 1982. Programme de classification hiérarchique par l'algorithme de la recherche en chaîne des voisins réciproques. *Les Cahiers de l'Analyse des Données* 7, 2 (1982), 219–225. http://www.numdam.org/item?id=CAD_1982__7_2_219_0

[13] Donald E. Knuth. 1998. *The Art of Computer Programming, Vol. 3: Sorting and Searching* (2nd ed.). Addison-Wesley, Reading, MA.

[14] Fuhito Kojima, Parag A. Pathak, and Alvin E. Roth. 2010. *Matching with couples: Stability and incentives in large markets.* Working Paper 16028. National Bureau of Economic Research. https://doi.org/10.3386/w16028

[15] Richard G. Niemi and John Deegan. 1978. A Theory of Political Districting. *American Political Science Review* 72, 4 (1978), 1304ÕÇô–1323. https://doi.org/10.2307/1954541

[16] Charles S. ReVelle and H. A. Eiselt. 2005. Location analysis: A synthesis and survey. *European Journal of Operational Research* 165, 1 (2005), 1–19. https://doi.org/10.1016/j.ejor.2003.11.032

[17] Federica Ricca, Andrea Scozzari, and Bruno Simeone. 2008. Weighted Voronoi region algorithms for political districting. *Mathematical and Computer Modelling* 48, 9-10 (2008), 1468–1477. https://doi.org/10.1016/j.mcm.2008.05.041

[18] Alvin E. Roth and Marilda Sotomayor. 1989. The college admissions problem revisited. *Econometrica* 57, 3 (1989), 559–570. https://doi.org/10.2307/1911052