Nash equilibrium sorting genetic algorithm for simultaneous competitive maximal covering location with multiple players

Abdullah Konak\textsuperscript{a}, Sadan Kulturel-Konak\textsuperscript{b} and Lawrence V. Snyder\textsuperscript{c}

\textsuperscript{a}Information Sciences and Technology, Penn State Berks, Reading, PA, USA; \textsuperscript{b}Management Information Systems, Penn State Berks, Reading, PA, USA; \textsuperscript{c}Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA

ABSTRACT

In this article, a new genetic algorithm (GA), called the Nash equilibrium sorting genetic algorithm (NESGA), is introduced to identify Nash equilibria for the competitive maximal covering location problem with two and three competitors, which is a combinatorial game theory problem where it is computationally intractable to enumerate all decision options of the competitors. Although GAs are widely used in combinatorial optimization, their applications to non-cooperative, simultaneous games have been limited owing to challenges in guiding the evolutionary search to Nash equilibria, which are not necessarily on the Pareto front of the search space and cannot be found by current multi-objective GAs. A novel fitness assignment strategy is proposed to help the NESGA to converge to multiple Nash equilibria. Computational experiments show that the NESGA can discover multiple Nash equilibria in a single run and outperform other game-theoretic GAs.

ARTICLE HISTORY

Received 13 February 2021
Accepted 14 June 2021

KEYWORDS

Facility location; competitive location; game theory; genetic algorithms; metaheuristics

1. Introduction

Facility location problems are among the classical operations research problems with the broadest range of application areas. One of the most widely studied facility location problems is the maximal covering location problem (MCLP) (Church and ReVelle 1974), where a decision maker seeks to cover as much customer demand as possible by opening multiple facilities, selected from a set of candidate points. Customer demands are concentrated at a set of discrete points, and a customer point is covered if there exists at least one facility within a critical distance from the customer point. The MCLP has extensively been studied in the literature, and several review articles (Drezner and Hamacher 2001; Farahani \textit{et al.} 2012; Melo, Nickel, and Saldanha-da-Gama 2009; Snyder 2006) summarize the existing work. The competitive version of the problem, where two or more competitors try to maximize their demand coverage in the same target market, is applicable when a firm determines facility locations in a competitive market. Regarding the nature of competition, the competitive MCLP can be modelled in two ways: simultaneous or sequential (Kress and Pesch 2012). In simultaneous models, the competitors make decisions without being aware of one another’s decisions. It is assumed that the competitors have the knowledge of all possible decisions that their opponents may make, and they maximize their expected payoff by choosing the decision that is the best response to their opponents’ possible decisions. Simultaneous games are useful for studying competitive behaviours as well as analysing strategies that would emerge as a response to market conditions, rules and principles.
Sequential models, which are more prevalent in the literature, are based on a Stackelberg game in which a competitor (i.e., leader) first locates its facilities, and then the second one (i.e., follower) locates its facilities optimally based on the leader’s decision. This NP-hard problem is usually formulated as an integer bi-level optimization problem. Sequential models are strategic decision-making tools at the organizational level compared to the macro-level use of simultaneous models. Several heuristics (Alekseeva and Kochetov 2013; Davydov et al. 2014; Davydov, Kochetov, and Carrizosa 2014; Seyhan, Snyder, and Zhang 2018) and exact approaches (Alekseeva, Kochetov, and Plyasunov 2015; Roboredo and Pessoa 2013) have been proposed to solve the sequential competitive MCLP.

In this article, the simultaneous version of the competitive MCLP is studied with two different demand capture models and two/three competitors, which has not been studied in the literature. A new approach, called the Nash equilibrium sorting genetic algorithm (NESGA), is proposed to find multiple Nash equilibria (NE) in the non-cooperative, simultaneous competitive MCLP and extend the problem to three competitors for the first time. In a multi-player, non-cooperative, simultaneous game, a decision profile is called NE if no player can improve its payoff by changing its decision option independently while the other players keep theirs unchanged. Simultaneous game theory problems are challenging for evolutionary algorithms because a problem may have no NE or multiple ones that are unlikely to be located on the Pareto-optimal front of the objective space. This aspect of non-cooperative simultaneous games makes it difficult for the existing single-/multi-objective evolutionary algorithms to converge to an NE. The current game-theoretic genetic algorithms (GAs) have not been applied to non-cooperative simultaneous games with three competitors. The NESGA utilizes a sorting-based fitness assignment strategy and multiple populations to discover multiple NE in a single run. The proposed sorting-based fitness assignment strategy can handle problems with multiple competitors.

The remaining sections of the article are organized as follows. Section 2 presents a review of the competitive MCLP and current game-theoretic evolutionary algorithms. Section 3 presents the mixed integer programming (MIP) formulations of the competitive MCLP with the closest-take-all and Huff gravity-based demand capture models. Section 4 presents the NESGA and its fitness evaluation method. In Section 5, it is shown that the NESGA can converge to multiple NE in a single run.

2. Background and previous work

Simultaneous games, where the players make their decisions simultaneously without knowledge of the other players’ decisions, are challenging to solve by evolutionary algorithms owing to the limitations imposed by finite population size (Adami, Schossau, and Hintze 2016). One of the challenges is identifying and maintaining NE during the search since a problem may have no NE, a single one or multiple ones. Sefrioui and Perlaux (2000) proposed an algorithm called NashGA for non-cooperative simultaneous games. In the NashGA, each player has its population representing a set of decisions of the player, and the best decision of a population at the current iteration is used to evaluate the decisions of the other populations in the next generation. Pace (2009) also proposed a multi-population GA to discover dominated strategies for the Traveller’s dilemma game, where the fitness of each solution in a population is evaluated against several randomly selected solutions from the other populations. Chen et al. (2002) used a negotiation policy to find the equilibrium in the payoff matrix of two players. Konak, Kulturel-Konak, and Snyder (2015) proposed the game-theoretic GA, where each individual of the leader’s population is evaluated against every individual of the follower’s population.

Several articles in the literature present a review of the competitive MCLP. Plastria (2001) introduced a taxonomy of the competitive MCLP in terms of the competition type, the market structure and the decision space for static problems. Kress and Pesch (2012) provided an extensive review of the sequential competitive MCLP. Alekseeva and Kochetov (2013) also presented a review of Stackelberg models and solution methods for the competitive MCLP. Eiselt, Marianov, and Drezner (2019) reviewed the latest trends in the competitive MCLP. The analysis of these review articles can be summarized in two main points: (1) the literature mainly focuses on sequential games, in particular,
Stackelberg-type games with two competitors; and (2) the existing approaches to the problem utilize metaheuristics or combinations of metaheuristics and mathematical programming methods since the problem is $\Sigma P_2$ complete (Noltemeier, Spoerhase, and Wirth 2007; Plastria and Vanhaverbeke 2008).

The competitive MCLP was first defined by ReVelle (1986) for determining the location of a single facility considering the existence of other competitor facilities (although similar ideas date back to Harold 1929). This formulation represents a static view of the problem. Serra and ReVelle (1994) extended the static problem defined by ReVelle (1986) as a Stackelberg game and described an iterative heuristic procedure such that the leader moves the location of a single facility to an unassigned location, and the follower solves the MCLP problem for the leader’s new solution.

Plastria and Vanhaverbeke (2008) considered MaxMin, MinRegret and Stackelberg-type objectives and presented integer programming formulations that could find exact Stackelberg equilibria when the follower had only a single facility to open. Later, Seyhan, Snyder, and Zhang (2018) extended this approach to multiple follower facilities, where the follower’s problem is assumed to be solved by a greedy algorithm (Kuehn and Hamburger 1963). Davydov et al. (2014) defined the competitive MCLP on the Euclidian plane, where the follower problem is solved with an exact approach and the leader decision options are chosen by a variable neighbourhood search heuristic. Küçükaydin, Aras, and Kuban Altınel (2011) introduced a bi-level mixed-integer nonlinear formulation to determine the leader and follower decisions considering the Huff gravity-based model. Barmak et al. (2016) and Nasiri et al. (2018) used a hybrid of heuristics and exact methods to solve a capacitated competitive MCLP.

In the context of the MCLP, a simultaneous game was first considered by Labbé and Louis Hakimi (1991), who studied the conditions for the existence of an NE for two competitors that aim to place a single facility on a network. Rhim, Ho, and Karmarkar (2003) analysed pure-strategy NE for an integrated facility location problem where competitors decide not only the locations to open facilities but also the capacity and pricing options of each facility. Godinho and Dias (2010) defined an iterative algorithm to solve a two-competitor simultaneous MCLP. Their algorithm starts from a feasible decision and sequentially computes the best response of each competitor to the last decision of the other until a pure NE is found, or a cycle of decisions is observed. Guo and Lai (2017) presented a simultaneous price and location game in which bricks-and-mortar stores compete with an online retailer. Cardinal and Hoefer (2010) extended the set covering problem on networks to non-cooperative games with cost-sharing among multiple players and identified NE in facility location games where players offer payments for connecting and opening existing facilities. Another related problem is to identify a saddle point in a continuous plane for the location of a new facility under the gravity-based Huff model (Drezner and Drezner 2004; Drezner 2014; Díaz-Báñez et al. 2011).

The competitive MCLP was also formulated as a multi-objective problem (Lančinskas and Žilinskas 2013; Redondo et al. 2015; Fernández and Tóth 2009; Konak, Kulturel-Konak, and Snyder 2017; Rohaninejad et al. 2017) or considering cooperation among competitors (Mallozzi 2011). In these approaches, the payoffs of two competitors are simultaneously optimized using a multi-objective GA to identify Pareto-optimal decision profiles. Note that the objective of the NESGA is to discover NE in non-cooperative simultaneous games where an NE does not necessarily reside on the Pareto-optimal front (such examples can be found in the decision profiles given in the Supplementary Material). A Pareto-optimal decision profile is also not necessarily an NE. In this sense, finding an NE for non-cooperative games is more challenging than discovering NE on the Pareto-optimal front, which provides a target to evaluate alternative decision profiles.

3. Competitive maximal covering location problem

There is a set of competitors, denoted by set $K$, targeting the demands of $n$ customer points distributed over a region. The candidate facility locations and customer points are represented by sets $I$ and $J$, respectively. Facilities are opened at the customer points in this article (i.e. $I = J$). Distance $d_{ij}$ represents the Euclidian distance between two points $i$ and $j$, and customers can only access facilities
located within a threshold distance of $D$. Each customer point $j$ has a deterministic demand of $w_j$. Each competitor $k$ aims to maximize its total demand capture $\pi_k$ by opening $\alpha_k$ number of facilities. Customers are indifferent among the competitors and locations. The only factor that determines the decision of a customer is the distance to the facilities.

Let $X_k = [x_{k1}, x_{k2}, \ldots, x_{kn}]$ represent a decision option of competitor $k$ such that $x_{ki} = 1$ if competitor $k$ locates a facility at point $i$, and $x_{ki} = 0$ otherwise, and $\Omega_k$ is the set of all decision options for competitor $k$. In the model, the competitors can co-locate their facilities at the same point. Let $z_{kij}$ be a binary variable such that $z_{kij} = 1$ if point $j$ is served by competitor $k$’s facility located at point $i$, and $z_{kij} = 0$ otherwise. The values of variable $z_{kij}$ depend on where the competitors locate their facilities as well as the assumptions about customer behaviour. Two demand capture models, namely 'closest-take-all' and 'Huff gravity-based' models, are considered.

The objective is to determine a decision for each competitor such that no competitor will benefit from changing its decision unilaterally while the other competitors keep their decisions unchanged. In other words, a set of NE decision profiles $[X_1^*, X_2^*, X_3^*]$ and $[X_1^*, X_2^*, X_3^*, X_4^*]$ is aimed to be identified for problems with $|K| = 2$ and $|K| = 3$, respectively, such that $X_k^*$ is the best decision option of competitor $k$ for the given best decision options of the other competitor(s). Let $s$ represent a decision profile (e.g. $s = [X_1, X_2, X_3]$) and $s_{-k}$ denote a partial decision profile excluding competitor $k$ (e.g. $s_{-1} = [X_2, X_3]$), and $\pi_k(X_k, s_{-k})$ be the payoff of decision option $X_k$ for a given partial decision profile $s_{-k}$.

**Definition 1:** A decision profile $s^*$ is called an NE if $\pi_k(X_k^*, s_{-k}^*) \geq \pi_k(X_k, s_{-k}) \quad \forall X_k \in \Omega_k, k \in K$.

### 3.1. Closest-take-all model

In the closest-take-all demand capture model, a customer always prefers the closest accessible facility, i.e. all demand of a customer point is captured by the closest accessible facility. If there exist multiple accessible facilities that are located at the same closest distance to a customer point, then the demand of such a point is shared equally by those facilities. Below, an MIP formulation is introduced for three-competitor games by extending an earlier model (Konak, Kulturel-Konak, and Snyder 2017) to verify whether a candidate NE returned by the NESGA is a true NE or not. Additional notation for the formulation is given as follows:

- $A(j)$ set of points that can cover customer point $j$, $A(j) = \{i \in I: d_{ij} \leq D\}$
- $C(i, j)$ set of points that are closer to point $j$ than point $i$, $C(i, j) = \{l \in A(j) : d_{lj} < d_{ij}\}$

**Problem MCLP (closest-take-all):**

$$\text{Max} \ \ \pi_1 = \sum_{j \in J} \sum_{i \in A(j)} \left( w_j z_{1ij} - \frac{1}{2} w_j z_{1ij} x_{2i} (1 - x_{3i}) - \frac{1}{2} w_j z_{1ij} x_{3i} (1 - x_{2i}) - \frac{2}{3} w_j z_{1ij} x_2 x_3 \right)$$

$$\text{Max} \ \ \pi_2 = \sum_{j \in J} \sum_{i \in A(j)} \left( w_j z_{2ij} - \frac{1}{2} w_j z_{2ij} x_{1i} (1 - x_{3i}) - \frac{1}{2} w_j z_{2ij} x_{3i} (1 - x_{1i}) - \frac{2}{3} w_j z_{2ij} x_1 x_3 \right)$$

$$\text{Max} \ \ \pi_3 = \sum_{j \in J} \sum_{i \in A(j)} \left( w_j z_{3ij} - \frac{1}{2} w_j z_{3ij} x_{1i} (1 - x_{2i}) - \frac{1}{2} w_j z_{3ij} x_{2i} (1 - x_{1i}) - \frac{2}{3} w_j z_{3ij} x_1 x_2 \right)$$

subject to:

$$\sum_{i \in A(j)} z_{kij} \leq 1 \quad k \in K, j \in J$$

$$z_{kij} \leq x_{ki} \quad k \in K, j \in J, i \in A(j)$$
\[ \sum_{i \in I} x_{ki} = \alpha_k \quad k \in K \]  

(4)

\[ z_{kij} \leq 1 - x_{lj} \quad k \in K, j \in J, i \in A(j), t \in K, l \in C(i,j) \]  

(5)

Although the objective functions of the above model are nonlinear owing to the interactions among the competitors, the model becomes linear when it is solved only for a competitor \( k \) with a given \( s_k \).Constraint (2) makes sure that point \( j \) can be covered by at most one facility of competitor \( k \). Constraint (3) requires that competitor \( k \) locates a facility at point \( i \) if this point covers any customers. Constraint (4) sets the number of facilities that competitor \( k \) opens. Constraint (5) states that competitor \( k \) cannot cover customer point \( j \) through point \( i \) if there exists a facility closer to point \( j \) than point \( i \).

### 3.2. Huff gravity-based model

In this model (Huff 1966), the probability that a customer prefers a specific facility is proportional to an attractiveness measure of the facility and inversely proportional to the distance. Let \( w_{kij} \) represent the amount of demand \( w_j \) captured by competitor \( k \)'s facility located at point \( i \). Assuming that all competitors and points have the same level of attractiveness, the ratio

\[ \frac{w_{kij}}{w_{rij}} = \frac{d_{ij}^\lambda}{d_{ij}^\lambda} \]  

holds between two facilities located at points \( i \) and \( l \) for \( i \neq j, l \neq j, d_{ij} \leq D \) and \( d_{lj} \leq D \), where \( \lambda \geq 1 \) is called the distance decay parameter. Note that the above ratio is also valid for the facilities of the same competitor (i.e. \( k = r \)).

The Huff gravity-based model implicitly assumes that \( w_j = \sum_{k \in K} \sum_{i \in I} w_{kij} z_{kij} \) if point \( j \) is covered by at least one facility. Since the customers have no preferences, the facilities located at point \( j \) share \( w_j \) equally. Below, an MIP formulation is introduced to verify whether a candidate NE is a true NE or not for the Huff gravity-based model.

**Problem MCLP (Huff gravity-based model):**

\[ \text{Max } \pi_k = \sum_{j \in J} \sum_{i \in A(j)} w_{kij} \]  

subject to:

\[ \sum_{i \in I} x_{ki} = \alpha_k \quad k \in K \]  

(8)

\[ z_{kij} \geq x_{ki} - \sum_{r \in K} x_{rj} \quad k \in K, j \in J, i \in A(j), i \neq j \]  

(9)

\[ z_{kij} \leq x_{ki} \quad k \in K, j \in J, i \in A(j) \]  

(10)

\[ z_{kij} = x_{kj} \quad k \in K, j \in J \]  

(11)

\[ z_{kij} \leq 1 - x_{lj} \quad r \in K, k \in K, j \in J, i \in A(j), i \neq j \]  

(12)

\[ w_{kij} \leq w_j z_{kij} \quad k \in K, j \in J, i \in A(j) \]  

(13)

\[ \sum_{k \in K} \sum_{i \in A(j)} w_{kij} \leq w_j \quad j \in J \]  

(14)
\[
\sum_{k \in K} \sum_{i \in A(j)} w_{kij} \geq w_jx_{rl} \quad r \in K, j \in J, l \in A(j)
\]  
(15)

\[
d_{ij}^k w_{kij} - d_{ij}^l w_{rlj} \leq M(2 - z_{kij} - z_{rlj}) \quad k, r \in K, j \in J, \{i, l\} \in A(j), i \neq j, l \neq j
\]  
(16)

\[
d_{ij}^k w_{kij} - d_{ij}^l w_{rlj} \geq M(z_{kij} + z_{rlj} - 2) \quad k, r \in K, j \in J, \{i, l\} \in A(j), i \neq j, l \neq j
\]  
(17)

\[
w_{kij} = w_jx_{klj} \left(1 - \frac{1}{2}x_{rlj}(1 - x_{ij}) - \frac{1}{2}(1 - x_{rlj})x_{ij} - \frac{2}{3}x_{rlj}x_{ij}\right) \quad k, r, t \in K, k \neq r \neq t, j \in J
\]  
(18)

\[z_{kij}, x_{ij} \in \{0, 1\}, w_{kij} \geq 0\]

Constraint (8) sets the number of facilities that competitor \(k\) opens. Constraint (9) states that competitor \(k\) must serve node \(j\) through its facility located at node \(i\) if no facility is located at node \(j\). Constraint (10) makes sure that competitor \(k\) can serve node \(j\) through node \(i\) only if the competitor has a facility located at node \(i\). Constraint (11) forces competitor \(k\) to serve node \(j\) if the competitor has a facility located at node \(j\) and \(i\) if no facility is located there. Constraints (13) and (14) are, respectively, upper bounds on the amount of demand \(w_j\) captured by competitor \(k\)’s facility located at point \(i\) and the total demand captured from node \(j\). Constraint (15) ensures that the total demand of node \(j\) is captured if node \(j\) is served by any facilities. Constraints (16) and (17) model the demand capture–distance relations given in Equation (6). Constraint (18) calculates the demand capture from node \(j\) through a facility located at node \(j\). The model becomes linear when it is solved for one competitor at a time to verify a candidate NE.

4. Description of the NESGA

In the NESGA, each competitor \(k \in K\) has its own population \(P_k = [X_k^1, \ldots, X_{\mu_k}]\) of \(\mu\) decision options, where \(X_{kq} = [x_{k1q}, \ldots, x_{k\mu_q}]\) denotes the \(q\)th decision option of competitor \(k\). The NESGA aims to converge the whole \(P_k\) toward a subdecision space that includes NE for competitor \(k\). To describe the NESGA, the game theory notation defined before by replacing \(\Omega_k\) with \(P_k\) is used, as follows:

\[X_k\] a decision option in \(P_k\)
\[X_{kq}\] the \(q\)th decision option in \(P_k\)
\[X_{k}^s\] a decision option that is a part of a candidate NE in \(P_k\)
\[S\] the set of all decision profiles for all competitors at an iteration (e.g. \(S = P_1 \times P_2 \times P_3\))
\[S_{\mu_k}\] the set of all partial decision profiles that exclude \(P_k\) at the current iteration
\[S_{\mu_k}^s\] the set of all partial candidate NE decision profiles that exclude \(P_k\)
\[s\] a decision profile in \(S\) (e.g. \(s = [X_1, X_2, X_3]\))
\[s_{\mu_k}\] a partial decision profile in \(S_{\mu_k}\)
\[s_{\mu_k}^s\] a partial decision profile in \(S_{\mu_k}^s\)
\[S_{\mu_k}^s(X_k^*)\] the subset of \(S_{\mu_k}^s\) such that \(X_k^*\) is a part of a candidate NE.
\[\pi_k(X_{kq}, s_{\mu_k})\] payoff of decision option \(X_{kq}\) with respect to a partial decision profile \(s_{\mu_k}\).

The following definition is used in the calculation of fitness values.

**Definition 2:** A decision profile \(s^* = [X_1^*, X_2^*, X_3^*]\) is called a candidate NE during an iteration of the NESGA if \(\pi_k(X_k^*, s_{\mu_k}^*) \geq \pi_k(X_{kq}, s_{\mu_k}^*)\) \(\forall X_{kq} \in P_k\) holds for each competitor \(k\). The set of all candidate NE decision profiles at the current iteration is denoted as \(N\).

In a generation, the NESGA compares each decision option in a population to every decision option in the other populations to construct a partial payoff matrix. Because multiple copies of a
decision option are irrelevant in the payoff matrix, the NESGA’s populations include only unique individuals. The overall procedure of the NESGA is given below, where $g_{max}$ is the maximum number of generations.

\[
\text{NESGA}() \\
\text{Initialize } P_k \text{ with } \mu \text{ random unique decision options} \\
\text{for } g = 1, \ldots, g_{max} \{ \\
\text{for } k \in K \{ \\
\text{Create } \mu \text{ unique offspring using procedure CrossoverMutation}(k) \\
\text{Add } \mu \text{ offspring to } P_k \\
\} \\
\text{Construct the payoff matrix} \\
\text{for } k \in K \{ \\
\text{Assign fitness values to } P_k \\
\text{Sort } P_k \text{ as described in the fitness assignment strategy} \\
\text{Remove the worst } \mu \text{ offspring from } P_k \\
\} \\
\}
\]

\subsection{4.1. Fitness assignment}

The fitness assignment of the NESGA is designed to maintain candidate NE in the current populations. Two concerns need to be addressed in the assignment of fitness values:

1. Assigning fitness values when the current populations do not include any candidate NE so that the populations retain decision options that are likely to produce promising offspring in future iterations.
2. If there are multiple candidate NE at the current iteration, how to retain them for future generations and assign fitness values to other decision options.

To address these concerns, a hierarchical approach is proposed to rank decision options in each population. The first component of the fitness function depends on whether a decision option is part of a candidate NE. If a decision option $X_{kq}$ is a part of the candidate NE in $P_k$, then its maximum payoff across the decision profiles in which it is a part of NE is used as its first fitness value:

\[
f^1_k(X_{kq}) = \begin{cases} \\
\max_{s^*_k \in S^*_k(X^*_k) \cap s_k^{-} } \pi_k(X_{kq}, s^*_{-k}) & X_{kq} = X^*_k, s^* \in N \\
0 & \text{otherwise} \end{cases} \tag{19}
\]

The second component of the fitness function depends on the existence of a candidate NE in the population, as follows:

\[
f^2_k(X_{kq}) = \begin{cases} \\
\max_{s_{-k} \in S_{-k} } \pi_k(X_{kq}, s_{-k}) & N = \emptyset \\
\max_{s^*_k \in S^*_k} \pi_k(X_{kq}, s^*_{-k}) & N \neq \emptyset \end{cases} \tag{20}
\]
If there is no current candidate NE in the populations \((i.e. N = \emptyset)\), then the maximum possible payoff of a decision option is used as its second fitness value. In other words, the maximax rule, which represents a 'risk-taking' behaviour, is used.

If the populations have candidate NE \((i.e. N \neq \emptyset)\), then the fitness values of the other decision options are evaluated considering only the set of the candidate NE. In this case, the maximax rule is also used but considering the payoff of the decision options only with respect to the candidate NE, not the entire payoff matrix. This approach intensifies the search around the decision options that are part of a candidate NE in each population and tests their fitness. After evaluating both fitness components, the populations are sorted in the descending order of Equation (19), breaking ties by Equation (20). Thereby, the NESGA maintains candidate NE over the generations and returns set \(N\) at the termination of the search.

### 4.2. Solution encoding and crossover/mutation

The NESGA can be used with any encoding schema and evolutionary operators. In this article, basic GA operators were utilized in the NESGA so that its performance, especially its fitness assignment strategy, could be studied without the effect of advanced encoding schemas, solution improvement methods, constraint handling techniques and parameter optimization. To compare the performance of the NESGA to the NashGA under the same conditions, both algorithms used a random key encoding (Bean 1994) and the same crossover/mutation operators. Let \(R_{kq} = \{r_{k1q}, \ldots, r_{knq}\}\) represent the chromosome of the \(q\)th decision option in \(P_k\) such that \(r_{kij} \in [0,1]\) is the random key corresponding to decision variable \(x_{ki}\). Given \(R_{kj}\), the decoding algorithm sets \(x_{ki} = 1\) for the largest \(\alpha_k\) random keys and \(x_{ki} = 0\) for all others to construct a decision option \(X_{kq}\). Thereby, the decoding algorithm ensures that a decision option \(X_{kq}\) always has \(\alpha_k\) facilities opened.

The NESGA used a uniform crossover to produce an offspring from two randomly selected parents such that the offspring randomly inherits each gene from either of the parents with a probability of 0.5. The mutation involves assigning a random number to a gene with a probability of \(1/n\). Identical solutions are not allowed in the population. Hence, if a generated offspring already exists in the population, it is discarded, and a new one is generated. The procedure of the crossover operator is given as follows:

```plaintext
CrossoverMutation(k) {
    Randomly uniformly select two decision options, \(a\) and \(b\), from \(P_k\)
    Do {
        for \(i = 1, \ldots, n\) {
            if (\text{rand}() < 0.5) then \(r_{ki} = r_{kia}\) else \(r_{ki} = r_{kib}\)
            if (\text{rand}() < 1/n) then \(r_{ki} = \text{rand}()\)
        }
        Decode \(R_k\) into \(X_k\)
    } While \((X_k \in P_k)\)
    Return \(X_k\)
}
```

### 5. Computational experiments

In the computational experiments, it is investigated whether the NESGA can converge to true NE, and its performance is compared to the NashGA (Sefrioui and Perlaux 2000). To demonstrate that the NESGA can discover multiple NE in a single run, competitive MCLP instances with known NE
are needed. Therefore, new test problem instances with \( n = \{30, 60, 120\} \) were created, and their true NE were identified through enumeration for small-size instances. The data of the test problems and their attributes are given in the Supplementary Material. The problem instances have different values of parameters \( \alpha_k, D \) and \( \lambda \), as given in the following tables.

If enumeration was not feasible, the MIPs given above were used to verify whether a candidate NE \( s^* \) with the payoff value \( \pi_k(X_k^*, s_{-k}^*) \) was a true NE or not, as follows. First, the optimal payoff value \( \pi_k^* \) of \( s^* \) for each competitor \( k \) was found using the MIP model after fixing the decision variables of the other competitors according to \( s_{-k}^* \). Then, the following relative error (RE) metric was calculated.

\[
RE(s^*) = \sum_{k \in K} \frac{\pi_k^* - \pi_k(X_k^*, s_{-k}^*)}{\pi_k(X_k^*, s_{-k}^*)}
\]  

(21)

By the definition of the NE, if \( RE(s^*) = 0 \), then the candidate \( s^* \) is a true NE. The mean relative error (MRE) for all candidate NE returned in a run can be calculated as follows:

\[
MRE = \frac{1}{|N|} \sum_{s^* \in N} RE(s^*)
\]  

(22)

The NESGA and NashGA were coded in MATLAB® R2018a. The enumeration and optimization algorithms to test equilibrium were coded in AMPL, and the resulting MIPs were solved by CPLEX 12.0 on the Pennsylvania State University’s Institute for Computational and Data Sciences’ Roar supercomputer. All MATLAB codes were run on a personal computer with a 4-core Intel i5 @ 3.5 GHz processor.

5.1. Computationalexperiments with the closest-take-all model

First, small problems with different \( \alpha_k \) values were used, as given in Table 1, to investigate whether the NESGA and the NashGA could discover NE, especially in problem instances with multiple NE. An exhaustive enumeration was used to determine NE for \( n = 30 \) and \( |K| = 3 \). The MIP model was also used to find the set of NE decision profiles for other problem instances with \( |K| = 2 \) where exhaustive enumeration was impossible, as follows. First, \( \Omega_1 \) was enumerated, and for each \( X_1 \in \Omega_1 \), the optimal decision \( X_2^* \) of competitor 2 was found using the MIP model. Then, the optimal response \( X_1^* \) to \( X_2^* \) was found, and if \( X_1^* = X_1 \), the decision profile \((X_1^*, X_2^*)\) is an NE.

Table 1 summarizes the performance of the NESGA and NashGA in terms of discovering NE in 30 random replications. The parameters of the NESGA and NashGA used in the experiments are also given in Table 1. Note that the NashGA was run for a larger \( g_{max} \) because the NESGA and NashGA require \((2\mu)^{|K|}\) and \(2\mu|K|\) objective function evaluations in a replication, respectively. These instances have different numbers of NE, ranging from one to eight (see Tables S1 and S2 in the Supplementary Material for details). Table 1 provides a summary of results as follows: the number of NE (nNE) discovered using enumeration, the number of unique candidate NE (nUCNE) found by the algorithms in 30 replications, the total number of true NE (nTNE) returned in 30 replications, the number of replications in which at least one true NE was discovered (nRTNE) and average central processing unit (CPU) time in seconds. A careful analysis of the results in Table 1 can reveal the pros and cons of the NESGA and NashGA. The results showed that the NESGA could discover multiple NE in a single run. The NESGA was able to discover all NE in the majority of instances with multiple NE. The NashGA also performed well for the instances with \( |K| = 2 \), especially if the instances had a limited number of NE, since the NashGA concentrates the search on a single solution. For the instances with \( |K| = 3 \), NE was found only for small values of \( \alpha_k \) with an enumeration. However, such instances provided strong evidence that the NESGA could also discover multiple NE in a single run for \( |K| = 3 \). In instance \((120,1,1,1,0.25)\), for example, the NESGA discovered all six symmetrical NE (see Table S2
Table 1. Comparison of the Nash equilibrium sorting genetic algorithm (NESGA) and Nash genetic algorithm (NashGA) on problem instances with known Nash equilibria (NE).

| Problem instance $(n, x_1, x_2, x_3, D)$ | NESGA $(|K| = 2; \mu = 40, g_{\text{max}} = 100; |K| = 3; \mu = 10, g_{\text{max}} = 200)$ | NashGA $(|K| = 2; \mu = 40, g_{\text{max}} = 4000; |K| = 3; \mu = 10, g_{\text{max}} = 28,000)$ |
|------------------------------------------|-----------------------------------------------|-----------------------------------------------|
|                                          | $n$NE | $n$UCNE | $n$TNE | $n$RTNE | Avg. CPU (s) | $n$UCNE | $n$TNE | Avg. CPU (s) | Avg. CPU (s) |
| (30, 3, 4, 0, 0.11)                      | 1     | 38      | 30     | 30     | 32     | 16     | 3      | 87      |
| (30, 3, 5, 0, 0.11)                      | 1     | 64      | 10     | 10     | 32     | 17     | 3      | 86      |
| (30, 3, 6, 0, 0.11)                      | 4     | 111     | 85     | 29     | 32     | 25     | 1      | 83      |
| (30, 3, 7, 0, 0.11)                      | 4     | 116     | 78     | 26     | 32     | 18     | 0      | 83      |
| (60, 2, 5, 0, 0.11)                      | 8     | 106     | 74     | 26     | 53     | 8      | 25     | 142     |
| (60, 2, 6, 0, 0.11)                      | 8     | 110     | 69     | 23     | 53     | 11     | 20     | 145     |
| (60, 2, 7, 0, 0.11)                      | 3     | 117     | 27     | 18     | 53     | 18     | 10     | 144     |
| (60, 2, 8, 0, 0.11)                      | 3     | 84      | 26     | 19     | 54     | 11     | 2      | 282     |
| (60, 2, 9, 0, 0.11)                      | 3     | 80      | 19     | 14     | 54     | 10     | 21     | 146     |
| (120, 2, 5, 0, 0.11)                     | 6     | 178     | 25     | 10     | 122    | 10     | 22     | 248     |
| (120, 2, 6, 0, 0.11)                     | 3     | 161     | 7      | 5      | 119    | 16     | 15     | 257     |
| (120, 2, 7, 0, 0.11)                     | 3     | 146     | 0      | 0      | 117    | 29     | 1      | 280     |
| (60, 1, 1, 2, 0.25)                      | 4     | 171     | 81     | 30     | 319    | 30     | 0      | 1227    |

Note: $n$NE = number of NE discovered using enumeration; $n$UCNE = number of unique candidate NE; $n$TNE = total number of true NE returned in 30 replications; $n$RTNE = number of replications in which at least one true NE was discovered; Avg. CPU = average central processing unit time.

Table 2. Comparison of the Nash equilibrium sorting genetic algorithm (NESGA) and Nash genetic algorithm (NashGA) on the closest-take-all model problem instances with unknown Nash equilibria (NE).

| Problem instance $(n, x_1, x_2, x_3, D)$ | NESGA $(|K| = 2; \mu = 40, g_{\text{max}} = 200; |K| = 3; \mu = 10, g_{\text{max}} = 200)$ | NashGA $(|K| = 2; \mu = 40, g_{\text{max}} = 8000; |K| = 3; \mu = 10, g_{\text{max}} = 28,000)$ |
|------------------------------------------|-----------------------------------------------|-----------------------------------------------|
|                                          | $n$VNE | $n$UCNE | $n$TNE | $n$RTNE | Avg.MRE | Avg.CPU (s) | $n$UCNE | $n$TNE | Avg.MRE | Avg.CPU (s) |
| (60, 4, 6, 0, 0.11)                      | 7     | 165     | 93     | 26     | 0.037   | 108       | 20     | 18     | 0.029 | 247       |
| (60, 4, 7, 0, 0.11)                      | 15    | 210     | 145    | 27     | 0.027   | 108       | 13     | 30     | 0.00  | 252       |
| (60, 4, 9, 0, 0.11)                      | 10    | 263     | 60     | 16     | 0.033   | 148       | 16     | 12     | 0.026 | 305       |
| (60, 3, 4, 6, 0.07)                      | 373   | 515     | 604    | 21     | 0.024   | 839       | 30     | 0      | 2.784 | 1693      |
| (120, 3, 4, 0, 0.11)                     | 5     | 240     | 36     | 16     | 0.034   | 246       | 7      | 26     | 0.000 | 462       |
| (120, 3, 5, 0, 0.11)                     | 1     | 306     | 3      | 3      | 0.045   | 241       | 22     | 6      | 0.065 | 479       |
| (120, 3, 6, 0, 0.11)                     | 0     | 366     | 0      | 0      | 0.218   | 238       | 25     | 0      | 0.339 | 489       |
| (120, 3, 7, 0, 0.11)                     | 1     | 294     | 1      | 1      | 0.057   | 234       | 28     | 0      | 0.062 | 501       |

Note: $n$VNE = number of candidate NE that were verified as true NE; $n$UCNE = number of unique candidate NE; $n$TNE = total number of true NE returned in 30 replications; $n$RTNE = number of replications in which at least one true NE was discovered; Avg. CPU = average central processing unit time.

in the Supplementary Material) and returned 5.26 true NE per replication (158/30) on average, while the NashGA was not able to find any of these six symmetrical NE.

Table 2 presents the results for instances with a larger number of facilities, for which the MIP model was used to verify whether a candidate NE returned by the algorithms was a true NE or not, as well as to calculate the MRE metric. In Table 2, the $n$VNE column indicates the number of candidate NE that were verified as a true NE, and the other columns have the same meaning as given in Table 1. In addition, the average MRE over 30 replications is provided (the ‘Avg. MRE’ column). Both algorithms were able to find NE in multiple replications for $|K| = 2$. However, the NashGA failed to discover an NE for the instances with $|K| = 3$. Similarly to the results in Table 1, the NESGA returned multiple NE in a single run. Overall, the NESGA identified a slightly larger number of NE in a larger number of replications compared to the NashGA.

In summary, the NESGA was able to discover multiple NE in a single run while the NashGA concentrates the search on a single candidate NE solution. Therefore, the NESGA outperformed the NashGA in instances with $|K| = 3$ or when the problem instance had multiple NE.
Table 3. Comparison of the Nash equilibrium sorting genetic algorithm (NESGA) and Nash genetic algorithm (NashGA) using the Huff gravity-based model with known Nash equilibria (NE) ($\lambda = 1$ for all instances).

| Problem instance $(n, \alpha_1, \alpha_2, \alpha_3, D)$ | NESGA $(n = 30; \mu = 20, g_{\text{max}} = 100; n > 30; \mu = 40, g_{\text{max}} = 100)$ | NashGA $(n = 30; \mu = 20, g_{\text{max}} = 2000; n > 30; \mu = 40, g_{\text{max}} = 4000)$ |
|-----------------------------------------------|---------------------------------|---------------------------------|
| $(30,2,4,0,0.17)$ | 4 16 88 30 14 | 4 29 28 |
| $(30,2,4,0,0.3)$ | 2 3 50 30 18 | 2 30 29 |
| $(30,2,4,0,0.5)$ | 1 2 24 24 30 | 1 30 42 |
| $(30,3,4,0,0.3)$ | 2 5 39 30 17 | 8 22 27 |
| $(30,3,4,0,0.5)$ | 1 3 28 28 30 | 1 30 40 |
| $(30,3,4,0,\infty)$ | 1 2 29 29 53 | 1 30 66 |
| $(60,2,5,0,0.18)$ | 9 45 150 27 123 | 11 26 200 |
| $(60,2,6,0,0.18)$ | 4 35 48 23 123 | 5 29 200 |
| $(60,2,7,0,0.18)$ | 1 30 9 9 122 | 2 29 199 |
| $(120,2,6,0,0.18)$ | 16 152 11 6 365 | 15 20 483 |

Note: $n_{\text{NE}} = \text{number of NE discovered using enumeration}; n_{\text{UCNE}} = \text{number of unique candidate NE}; n_{\text{TNE}} = \text{total number of true NE returned in 30 replications}; n_{\text{RTNE}} = \text{number of replications in which at least one true NE was discovered}; \text{avg. CPU} = \text{average central processing unit time}.$

5.2. Computational experiments with the Huff gravity-based model

The computational experiments in this section focused on evaluating the performance of the NESGA using the Huff gravity-based model. Table 3 summarizes the results of the problem instances with $|K| = 2$ for which NE could be determined through enumeration (see Table S3 in the Supplementary Material for the details of the results). The fitness assignment mechanism of the NESGA encourages the population to retain multiple candidate NE, whereas the NashGA intensifies the search around one NE. This advantage of the NashGA became apparent for problem instances with a single NE. For example, in instance $(30,2,4,0,0.5)$ with a single NE, the NashGA returned the NE in all replications, whereas the NESGA did so in 24 replications. On the other hand, the NESGA became advantageous when a problem instance had multiple NE. For example, in instance $(30,2,4,0,0.17)$ with four NE, the NashGA could not find one of them while the NESGA returned this NE in 21 replications [see $(30,2,4,0,0.17)$-III in Table S2 in the Supplementary Material]. This pattern was observed for the instances of $n = 60$. In instance $(60,2,5,0,0.18)$, the NESGA was able to find all nine NE in multiple replications but the NashGA missed some of them. On the other hand, the NashGA performed well for instance $(60,2,7,0,0.18)$, where there is a single NE. In summary, the NESGA was shown to be a more robust algorithm in terms of discovering the whole spectrum of NE.

Table 4 presents the results of two small-size problems with $|K| = 3$. The instance with $D = 0.3$ has 11 NE, which were identified by exhaustive enumeration. The NESGA found all of these 11 NE in multiple replications. In this instance, the NESGA returned a total of 286 unique candidate NE in 30 replications, and 257 of them were true NE. As seen in Table 4, the NESGA discovered each of the NE in multiple random replications, ranging from 17 to 28. The instance with $D = 0.5$ has three NE, and the NESGA was also able to find these NE in multiple replications. For this instance, the NESGA returned a total of 84 unique candidate NE in 30 replications, and 57 of them were true NE, whereas the NashGA could not find any of the listed NE.

Table 5 summarizes the results of larger problem instances for which at least one true NE was identified. The NESGA was able to identify a true NE in all problem instances, while the NashGA failed to do so in the problems with $|K| = 3$.

5.3. Convergence of the NESGA experiments with the Huff gravity-based model

Since NE may be inferior points in the solution space, demonstrating the convergence of the NESGA requires a different approach than a typical heuristic algorithm does. The MRE metric was used to demonstrate the convergence of the NESGA using instances $(60,4,3,2,0.30)$ and
Table 4. Performance of the Nash equilibrium sorting genetic algorithm (NESGA) in simultaneous problems with three competitors ($\lambda = 1, \mu = 10, g_{\text{max}} = 100$ for all).

| Problem instance $(n, \alpha_1, \alpha_2, \alpha_3, D)$ | NE | $\pi_1$ | $\pi_2$ | $\pi_3$ | Decision profile | No. of replications in which NE was found |
|---------------------------------|----|---------|---------|---------|-----------------|----------------------------------|
| (30,3,2,1,0.3)                  | I  | 6.630929 | 4.769115 | 2.669956 | (2,17,22)–(11,22)–(15) | 25                               |
|                                | II | 6.630929 | 4.868876 | 2.570195 | (2,17,22)–(15,22)–(11) | 25                               |
|                                | III| 6.630929 | 5.240151 | 2.19892  | (2,17,22)–(11,15)–(22) | 23                               |
|                                | IV | 6.738115 | 4.661929 | 2.463008 | (11,17,22)–(2,22)–(15) | 27                               |
|                                | V  | 6.738115 | 4.868876 | 2.570195 | (11,17,22)–(2,22)–(15) | 27                               |
|                                | VI | 6.837876 | 4.661929 | 2.19892  | (15,17,22)–(2,22)–(11) | 24                               |
|                                | VII| 6.837876 | 4.769115 | 2.463008 | (15,17,22)–(2,22)–(11) | 24                               |
|                                | VIII| 7.209151| 4.661929 | 2.19892  | (2,22,25)–(2,22)–(15) | 25                               |
|                                | IX | 7.439071 | 4.16792  | 2.463008 | (11,15,22)–(17,22)–(2) | 17                               |
|                                | X  | 7.439071 | 4.432008 | 2.19892  | (11,15,22)–(2,17)–(22) | 18                               |
|                                | XI | 7.439071 | 4.661929 | 1.969    | (11,15,22)–(2,22)–(17) | 23                               |

Table 5. Comparison of the Nash equilibrium sorting genetic algorithm (NESGA) and Nash genetic algorithm (NashGA) on Huff gravity-based model instances with unknown Nash equilibria (NE) ($\lambda = 1$ for all instances).

| Problem instance $(n, \alpha_1, \alpha_2, \alpha_3, D)$ | NESGA ($|K| = 2; \mu = 40, g_{\text{max}} = 100$; $|K| = 3; \mu = 10, g_{\text{max}} = 100$) | NashGA ($|K| = 2; \mu = 40, g_{\text{max}} = 4000$; $|K| = 3; \mu = 10, g_{\text{max}} = 14,000$) |
|---------------------------------|---------------------------------|---------------------------------|
|                                 | nVNE  | nUCNE | nTNE | nRTNE | MRE | Avg. CPU (s) | nVNE  | nUCNE | nTNE | nRTNE | MRE | Avg. CPU (s) |
| (60,4,6,0,0.18)                 | 2     | 100   | 12   | 9     | 0.014 | 123 | 9     | 2     | 11   | 0.010 | 182 |                     |
| (60,4,6,0,0.30)                 | 1     | 12    | 8    | 8     | 0.012 | 204 | 1     | 1     | 30   | 0.00  | 262 |                     |
| (60,4,3,2,0.30)                 | 2     | 184   | 2    | 1     | 0.046 | 271 | 30    | 0     | 0    | 1.096 | 362 |                     |
| (120,4,3,2,0.18)                | 1     | 113   | 1    | 1     | 0.127 | 492 | 30    | 0     | 0    | 1.556 | 664 |                     |
| (120,5,4,3,0.15)                | 1     | 715   | 1    | 1     | 0.114 | 484 | 30    | 0     | 0    | 0.165 | 676 |                     |
| (120,6,4,3,0.11)                | 1     | 739   | 1    | 1     | 0.124 | 391 | 30    | 0     | 0    | 1.65  | 578 |                     |
| (120,6,5,4,0.11)                | 2     | 745   | 2    | 1     | 0.141 | 398 | 30    | 0     | 0    | 1.762 | 563 |                     |

Note: NE = Nash equilibrium.

6. Conclusion

In this article, a new GA called the NESGA was proposed to discover NE in the case of the competitive MCLP with two and three competitors, for the first time. Identifying NE is important to study competitor behaviours that would emerge as a response to market competition, conditions, rules and...
principles in the context of the competitive MCLP. The NESGA is a multi-population GA with a hierarchical fitness assignment method that ranks population members with respect to all candidate NE in the population. The computational experiments showed that the NESGA could discover multiple NE in a single run, and the proposed fitness evaluation method was effective in problems with three competitors. The NESGA performed robustly over a wide range of problem instances, including ones with three competitors for which the NashGA failed to converge. In addition, a novel way was proposed to demonstrate the convergence of evolutionary algorithms in the case of simultaneous game theory problems. Certainly, the NESGA can be applied to any type of combinatorial game theory problem. Studying the performance of the proposed fitness evaluation method in other combinatorial game theory problems would be an interesting avenue for further research.

**Note**

1. Throughout the article, the same form of the abbreviation NE is used for both singular ‘Nash equilibrium’ and plural ‘Nash equilibria’.

**Disclosure statement**

No potential conflict of interest was reported by the authors.
Data availability statement

Data are included in the Supplementary Material.

References

Adami, Christoph, Jory Schossau, and Arend Hintze. 2016. “Evolutionary Game Theory Using Agent-Based Methods.” *Physics of Life Reviews* 19: 1–26.

Alekseeva, Ekaterina, and Yury Kochetov. 2013. “Matheuristics and Exact Methods for the Discrete (r|p)-Centroid Problem.” In *Metaheuristics for Bi-level Optimization*, edited by El-Ghazali Talbi, 189–219. Berlin: Springer.

Alekseeva, Ekaterina, Yury Kochetov, and Alexandr Plyasunov. 2015. “An Exact Method for the Discrete (r| p)-Centroid Problem.” *Journal of Global Optimization* 63 (3): 445–460.

Barmak, Azam Sadat, Majid Vafeai Jahan, Vahid Mahmoodian, and Zohre Sadranzhad. 2016. “A Bi-level Genetic Model for Competitive Capacitated Facility Location.” *American Journal of Intelligent Systems* 6 (3): 66–73.

Bean, James C. 1994. “Genetic Algorithms and Random Keys for Sequencing and Optimization.” *ORSA Journal on Computing* 6 (2): 154–160.

Cardinal, Jean, and Martin Hoefer. 2010. “Non-cooperative Facility Location and Covering Games.” *Theoretical Computer Science* 411 (16): 1855–1876.

Chen, Jen-Hsiang, Kuo-Ming Chao, Nick Godwin, Colin Reeves, and Peter Smith. 2002. “An Automated Negotiation Mechanism Based on Co-evolution and Game Theory.” In *Proceedings of the 2002 ACM Symposium on Applied Computing*, 63–67. Madrid, Spain: ACM.

Church, R., and C. ReVelle. 1974. “The Maximal Covering Location Problem.” *Papers of the Regional Science Association* 32: 101–118.

Drezner, Tammy. 2014. “A Review of Competitive Facility Location in the Plane.” *Logistics Research* 7 (114): 1–12.

Drezner, Zvi, and Horst W. Hamacher. 2001. *Facility Location: Applications and Theory*. New York: Springer Science & Business Media.

Eiselt, H. A., Vladimir Marianov, and Tammy Drezner. 2019. “Competitive Location Models.” In *Location Science*, edited by Gilbert Laporte, Stefan Nickel, and Francisco Saldanha da Gama, 391–429. Cham: Springer International Publishing.

Farahani, Reza Zanjirani, Nasrin Asgari, Nooshin Heidari, Mahtab Hosseinkia, and Mark Goh. 2012. “Covering Problems in Facility Location: A Review.” *Computers & Operations Research* 62 (1): 368–407.

Godinho, Pedro, and Joana Dias. 2010. “A Two-Player Competitive Discrete Location Model with Simultaneous Decisions.” *European Journal of Operational Research* 207 (3): 1419–1432.

Guo, Wen-Chung, and Fu-Chuan Lai. 2017. “Prices, Locations and Welfare When an Online Retailer Competes with Heterogeneous Brick-and-Mortar Retailers.”* The Journal of Industrial Economics* 65 (2): 439–468.

Harold, Hotelling. 1929. “Stability in Competition.” *Economic Journal* 39 (153): 41–57.

Huff, David L. 1966. “A Programmed Solution for Approximating an Optimum Retail Location.” *Land Economics* 42 (3): 293–303.
Labbé, Martine, and S. Louis Hakimi. 1991. “Market and Locational Equilibrium for Two Competitors.” *Operations Research* 39 (5): 749–756.

Lančinskas, Algirdas, and Julius Žilinskas. 2013. “Solution of Multi-objective Competitive Facility Location Problems Using Parallel NSGA-II on Large Scale Computing Systems.” In *Applied Parallel and Scientific Computing (PARA 2012) Lecture Notes in Computer Science*, edited by Pekka Manninen, and Per Öster, 422–433. Berlin: Springer.

Mallozzi, Lina. 2011. “Cooperative Games in Facility Location Situations with Regional Fixed Costs.” *Optimization Letters* 5 (1): 173–181.

Melo, M. Teresa, Stefan Nickel, and Francisco Saldanha-da-Gama. 2009. “Facility Location and Supply Chain Management—A Review.” *European Journal of Operational Research* 196 (2): 401–412.

Nasiri, Mohammad Mahdi, Vahid Mahmoodian, Ali Rahbari, and Shabnam Farahmand. 2018. “A Modified Genetic Algorithm for the Capacitated Competitive Facility Location Problem with the Partial Demand Satisfaction.” *Computers & Industrial Engineering* 124: 435–448.

Noltemeier, H., J. Spoerhase, and H. C. Wirth. 2007. “Multiple Voting Location and Single Voting Location on Trees.” *European Journal of Operational Research* 181 (2): 654–667.

Pace, M. 2009. “How a Genetic Algorithm Learns to Play Traveler’s Dilemma by Choosing Dominated Strategies to Achieve Greater Payoffs.” Paper presented at the 2009 IEEE Symposium on Computational Intelligence and Game, September 7–10.

Plastria, Frank. 2001. “Static Competitive Facility Location: An Overview of Optimisation Approaches.” *European Journal of Operational Research* 129 (3): 461–470.

Plastria, Frank, and Lieselot Vanhaverbeke. 2008. “Discrete Models for Competitive Location with Foresight.” *Computers & Operations Research* 35 (3): 683–700.

Redondo, Juana L., José Fernández, José Domingo Álvarez Hervás, Aránzazu Gila Arrondo, and Pilar M. Ortigosa. 2015. “Approximating the Pareto-Front of a Planar Bi-objective Competitive Facility Location and Design Problem.” *Computers & Operations Research* 62: 337–349.

ReVelle, Charles. 1986. “The Maximum Capture or Sphere of Influence Location Problem: Hotelling Revisited on a Network.” *Journal of Regional Science* 26 (2): 343–358.

Rhim, Hosun, Teck H. Ho, and Uday S. Karmarkar. 2003. “Competitive Location, Production, and Market Selection.” *European Journal of Operational Research* 149 (1): 211–228.

Robredo, Marcos Costa, and Artur Alves Pessoa. 2013. “A Branch-and-Cut Algorithm for the Discrete (r | p)-Centroid Problem.” *European Journal of Operational Research* 224 (1): 101–109.

Rohaninejad, Mohammad, Hamidreza Navidi, Behdin Vahedi Nouri, and Reza Kamranrad. 2017. “A new Approach to Cooperative Competition in Facility Location Problems: Mathematical Formulations and an Approximation Algorithm.” *Computers & Operations Research* 83: 45–53.

Sefrioui, M., and J. Perlaux. 2000. “Nash Genetic Algorithms: Examples and Applications.” Paper presented at the Proceedings of the 2000 Congress on Evolutionary Computation, La Jolla, CA, USA.

Serra, Daniel, and Charles ReVelle. 1994. “Market Capture by two Competitors: the Preemptive Location Problem.” *Journal of Regional Science* 34 (4): 549–561.

Seyhan, Tolga H., Lawrence V. Snyder, and Ying Zhang. 2018. “A New Heuristic Formulation for a Competitive Maximal Covering Location Problem.” *Transportation Science* 52 (5): 1156–1173.

Snyder, Lawrence V. 2006. “Facility Location Under Uncertainty: a Review.” *IIE Transactions* 38 (7): 547–564.