European Journal of Operational Research 285 (2020) 1153–1160

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Interfaces with Other Disciplines

Search and rescue in the face of uncertain threats

Thomas Lidbetter

Department of Management Science and Information Systems, Rutgers Business School, Newark, NJ 07102, USA

ARTICLE INFO

Article history:
Received 26 March 2019
Accepted 15 February 2020
Available online 21 February 2020

Keywords:
Game theory
Search games
Search and rescue
Trees

ABSTRACT

We consider a search problem in which one or more targets must be rescued by a search party, or *Searcher*. The targets may be survivors of some natural disaster, or prisoners held by an adversary. The targets are hidden among a finite set of locations, but when a location is searched, there is a known probability that the search will come to an end, perhaps because the Searcher becomes trapped herself, or is captured by the adversary. If this happens before all the targets have been recovered, then the rescue attempt is deemed a failure. The objective is to find the search that maximizes the probability of recovering all the targets. We present and solve a game theoretic model for this problem, by placing it in a more general framework that encompasses another game previously introduced by the author. We also consider an extension to the game in which the targets are hidden on the vertices of a graph. In the case that there is only one target, we give a solution of the game played on a tree.

1. Introduction

Many search and rescue operations may be dangerous for the search party, or *Searcher*. For example, searching in unstable buildings for earthquake survivors, searching for lost miners in a cave system, or performing a military rescue operation. One or more targets must be rescued, but there is a danger that the search is cut short due to one of these threats. For example, the Searcher is captured herself (in the case of a military operation), or to some incident caused by Nature results in the search being terminated (such as the Searcher becoming trapped herself).

We model such operations by introducing a new search model. We assume that a known number of targets are located among a finite set of possible locations, and at each location there is a given probability (which may depend on the location) that when that location is searched, the search will come to an end. The objective is to choose which order to search the locations to maximize the probability that the targets are all rescued.

We do not make any assumptions on the probability distribution with which the targets are located, but rather seek to find the randomized search that maximizes the probability of success in the worst case. Equivalently, we study a zero-sum game between the Searcher and a *Hider* who chooses where the targets are hidden. The latter way of framing the problem is particular appropriate for military applications. We call this game the *search and rescue game*, and it lies in the field of *search games*. For good general overviews of the literature on search games, see Alpern and Gal (2003), Gal (2011) or Hohzaki (2016).

We begin by defining the game precisely in Section 2, in which we also point out a relation of the game to a scheduling problem introduced by Agnetis, Detti, Pranzo, and Sodhi (2009). In Section 3, we give a solution to the game. It turns out that the solution can be found by a similar method to that of the game introduced by Lidbetter (2013b). The game studied in the latter paper also involves a known number of targets located among a finite number of locations, but there is no danger that the Searcher will be captured herself. Instead, there is a cost associated with searching each location, and the objective is to minimize the total cost of finding all the targets. Despite their similarities, the game of Lidbetter (2013b) and the game of this paper do not appear to be equivalent, so we unify them in a more general framework, simultaneously giving solutions to both.

In Section 4 we extend the game so that it is played on a graph. More precisely, we assume the hiding locations are vertices of the graph, and the Searcher must begin her search at a given vertex. We assume that the vertices of the graph must be searched according to an *expanding search*, a search paradigm introduced by Alpern and Lidbetter (2013). Roughly speaking, an expanding search of a graph is an ordering of the vertices such that each vertex is adjacent to some previously chosen vertex. In the context of the search and rescue game, this model of search is appropriate for situations in which locations must be searched contiguously, and after each one has been successfully searched, it can be marked as "secure", so that there is no danger of capture if the Searcher revisits it. The expanding search paradigm has been used more

E-mail address: tlidbetter@business.rutgers.edu

https://doi.org/10.1016/j.ejor.2020.02.029
0377-2217/

recently in Angelopoulos, Dürr, and Lidbetter (2019) and Alpern and Lidbetter (2019).

We give a solution to the search and rescue game played on a tree in the case that there is only one target, by giving a recursive method for calculating optimal strategies and the value of the game. A similar approach was taken by Alpern (2010), who found the solution to a different search game on a tree. In Alpern's game, the Searcher walks on a tree with the aim of minimizing the time taken to find a target, and the time taken to traverse an arc depends on the direction of travel.

This work takes a different approach to much of the literature on search games, which have the objective of minimizing some cost incurred in finding one or more target. In addition to the papers on search games already cited, we finish this section by briefly discussing some other recent work that takes a cost minimizing approach.

The classic model of network search games, as studied in Gal (1979, 2001), assumes that a Searcher, beginning at a fixed point of a network, wishes to find a immobile Hider located on the network in minimal time. This framework was extended in Dagan and Gal (2008) and Alpern, Baston, and Gal (2008) to allow an arbitrary starting point for the Searcher, and in Alpern (2019) to restrict both the Searcher's starting point and the Hider's hiding point to a fixed subset of the network. Alpern and Lidbetter (2015) considered a model of network search in which the Searcher has two speeds of travel: a slow speed at which she can detect the object when she passes it, and a fast speed at which she cannot. Lidbetter (2013a) evaluated the performance of the Searcher strategy known as the *Random Chinese Postman Tour* in the classic network search model of Gal (1979). Baston and Kikuta (2013, 2015) consider a more general search game on a network where the vertices may have *search costs* that the Searcher has to pay to search them.

There are some instances of search games being considered that do not take a cost minimizing approach. For example Lin and Singham (2016) consider a game in which a Searcher wishes to maximize the probability of finding a target before an unknown deadline; Gal and Casas (2014) introduced a predator-prey game in which a predator wishes to maximize the probability of capturing the prey.

This is the first paper, as far as we are aware, to consider a search game that models a scenario in which the Searcher may be captured herself.

2. Preliminaries

We now formally define the search and rescue game. A set of k targets must be rescued from a set of locations $S \equiv \{1, \dots, n\}$, where $1 \le k \le n-1$. The locations must be searched sequentially until all the targets have been found. If a location i is searched, there is a probability $p_i \in (0, 1)$ that the search is successful and all targets located there will be found. This probability is independent of where the location appears in the sequence. With probability $1 - p_i$, the Searcher will be captured herself, and no more locations can be searched. Note that we disallow $p_i = 0$ because the Hider could place a target in any such location i so that the value of the game is 0, and we disallow $p_i = 1$ because it is trivially the case that such a location i should be searched first.

Formally, a strategy for the Searcher is an ordering $\sigma : S \to S$, so that $\sigma(i)$ is the hiding location in the ith position in the ordering. We refer to an ordering σ as a *search*. For the Hider, note that strategies in which more than one target is hidden in the same location are dominated, so we take the Hider's strategy set to be all subsets $H \in S^{(k)} \equiv \{A \subseteq S : |A| = k\}$.

In order to define the payoff, first note that the probability the Searcher will not be captured while searching a subset A of hiding locations is

$$f(A) \equiv \prod_{i \in A} p_i.$$

For a given Searcher strategy σ, let

$$S_i^\sigma = \cup \{j \in S : \sigma^{-1}(j) \le i\}$$

be the first i locations searched. Then for a given Hider strategy H and Searcher strategy σ, the payoff $P(H, \sigma)$ of the game is $f(S_i^\sigma)$, where i is minimal such that $H \subseteq S_i^\sigma$. That is, $P(H, \sigma)$ is the probability that the Searcher will not be captured by the time she finds all the targets.

The Searcher's objective is to maximize the payoff, and the Hider's is to minimize it. This is a finite zero-sum game, so, by the minimax theorem of von Neumann (1928) for zero-sum games, it has optimal (max-min) mixed strategies and a value. A mixed strategy s for the Searcher is a probability distribution over all searches of S, and a mixed strategy h for the Hider is a probability distribution over $S^{(k)}$. For a mixed Hider strategy h and a mixed Searcher strategy s, we denote the expected payoff of the game by $P(h, s)$.

2.1. Relation to unreliable job sequencing

Consider the search and rescue game in the case that $k = 1$. In this case, a mixed strategy for the Hider is a probability distribution over the set S of locations. Such a strategy can be described by a vector of probabilities $\mathbf{x} \in \mathbb{R}^n$ with $\sum_i x_i = 1$, where x_i is the probability that the Hider is in location i. Then for a given Searcher strategy σ, the probability $P(\mathbf{x}, \sigma)$ the target is rescued when the Hider uses some mixed strategy $\mathbf{x}$ is given by

$$P(\mathbf{x}, \sigma) = x_{\sigma(1)} p_{\sigma(1)} + x_{\sigma(2)} p_{\sigma(1)} p_{\sigma(2)} + \cdots + x_{\sigma(n)} p_{\sigma(1)} \cdots p_{\sigma(n)}. \tag{1}$$

We call the problem of choosing σ to maximize $P(\mathbf{x}, \sigma)$ the *best response problem*, which is distinct from the problem of finding an optimal mixed strategy for the Searcher in the game. This problem has been considered by Agnetis et al. (2009) in the context of machine scheduling. Here, the problem is that n jobs must be processed by a machine, and a reward of x_i is obtained after successfully processing job i. The jobs are processed sequentially, and the probability that a job i is successfully processed is p_i. Otherwise, with probability $1 - p_i$, job i fails, and no more jobs can be processed. The expected reward for a given ordering of the jobs is equal to $P(\mathbf{x}, \sigma)$.

Although Agnetis et al. (2009) consider the more general problem where the jobs are sequenced by multiple machines, they show that if there is only one machine, the optimal policy is given by ordering the jobs in non-increasing order of the index $p_i x_i / (1 - p_i)$.

Agnetis et al. (2009) also point out a connection between their problem and a classic problem of Monma and Sidney (1979) in which n components have to be sequentially tested until either a component fails or all the components pass the tests. The cost of testing component i is c_i and the probability it passes the test is q_i. Agnetis et al. (2009) show that with $p_i = q_i$ and $x_i = c_i / q_i$, this problem is equivalent to choosing an ordering to *minimize* the expression on the right-hand side of (1) (as opposed to their problem, which is to maximize it). The solution is to order S in non-*decreasing* order of the index $x_i p_i / (1 - p_i)$, rather than non-increasing order.

3. A more general search game

We now place the search and rescue game in a more general context by defining a broader class of search games between a

Hider and a Searcher. As before, the game is played on a set S of locations, and this time $f : 2^S \to \mathbb{R}$ is an arbitrary set function. We view f as a reward function, and we assume that the values $f(A)$ are given by an oracle. The Hider's strategy set is all subsets $H \in S^{(k)}$, for some k, and the Searcher's strategy set is all searches (or orderings) of S.

As before, for a given Hider strategy H and Searcher strategy σ, the payoff $P = P_f$ of the game is given by $P(H, \sigma) = f(S_i^\sigma)$, where i is minimal such that $H \subseteq S_i^\sigma$. The Searcher is the maximizer and the Hider the minimizer. We will denote this game by Γ_f.

We give a sufficient condition on f for which Γ_f has a simple closed-form solution. For $i \in S$ and $A \subseteq S$, let $f_A(i) = f(A \cup i) - f(A)$. (For brevity, we write $A \cup i$ for $A \cup \{i\}$ and $f(i)$ for $f(\{i\})$.)

Definition 1. Let $f : 2^S \to \mathbb{R}$ be positive and $f(A) < f(B)$ for $B \subset A$. If there is a $\mathbf{z} \in \mathbb{R}^n$ with $z_r > 0$ for all r, such that

$$\frac{f_{A\cup j}(i)}{f_{A\cup i}(j)} = \frac{z_i}{z_j} \text{ when } i \notin A \text{ and } j \notin A, \tag{2}$$

then f is **z**-*indexable*.

Note that the f being strictly decreasing implies that $f_{A\cup j}(i) < 0$ for all $i, j \notin A$, so that the left-hand side of (2) is well-defined and positive.

The term "indexable" is inspired by the use of the term in Bertsimas and Niño Mora (1996) to describe dynamic and stochastic scheduling problems whose solution is given by assigning an index to each job and, at every stage, choosing the job with the highest index.

We first observe that if f is indexable, then for $k = 1$, the best response problem for the Γ_f is indexable, in the sense of Bertsimas and Niño Mora (1996).

Theorem 1. *Suppose f is* **z**-*indexable, and consider a mixed Hider strategy $x \in \mathbb{R}^n$. The solution to the best response problem of Γ_f is to search the elements of S in non-increasing order of the index x_i/z_i.*

Proof. The proof is a standard interchange argument. For a fixed search σ, the expected payoff of the game is

$$P(x, \sigma) = \sum_{i=1}^{n} x_{\sigma(i)} f(S_i^\sigma).$$

Suppose the elements of S are searched according to some search σ which is not in non-increasing order of the index x_i/z_i. Let j be some element of S such that the index of the element i that is searched immediately after j is larger than that of j. That is, $x_i/z_i > x_j/z_j$. Let σ' be the same as σ, with the order of elements i and j transposed.

Let $A = S^\sigma_{\sigma^{-1}(j)} - \{j\}$ be the subset of locations searched up but not including location j. Then

$$\begin{aligned} P(x, \sigma) - P(x, \sigma') &= (x_j f(A \cup j) + x_i f(A \cup \{i, j\})) \\ &\quad - (x_i f(A \cup i) + x_j f(A \cup \{i, j\})) \\ &= x_i f_{A\cup i}(j) - x_j f_{A\cup j}(i) \\ &= z_j f_{A\cup j}(i) \left(\frac{x_i}{z_i} - \frac{x_j}{z_j} \right) \text{ (by (2))} \\ &> 0. \end{aligned}$$

Hence, σ is not optimal, a contradiction. The theorem follows. □

3.1. Examples of **z**-Indexable games

Here we describe some examples of games Γ_f that are **z**-indexable.

3.1.1. The search and rescue game

It is easy to verify that for the search and rescue game, the function f is **z**-indexable. Indeed, for $i, j \notin A$, we have

$$\frac{f_{A\cup j}(i)}{f_{A\cup i}(j)} = \frac{\left(p_i p_j \prod_{s\in A} p_s\right) - \left(p_j \prod_{s\in A} p_s\right)}{\left(p_i p_j \prod_{s\in A} p_s\right) - \left(p_i \prod_{s\in A} p_s\right)} = \frac{p_j(1-p_i)}{p_i(1-p_j)}.$$

Thus, we can take $z_i = (1 - p_i)/p_i$, and Theorem 1 implies that the solution to the best response problem for $k = 1$ is to search the locations in non-increasing order of the index $x_i/z_i = x_i p_i/(1 - p_i)$. This is consistent with the result of Agnetis et al. (2009).

We may also extend the game, as in Agnetis et al. (2009), by incorporating a discount factor on the probabilities x_i. More precisely, suppose that if there is a target at the location that appears in position t of a search, there is a probability γ^t that the target will be there when that location is searched, where $0 \le \gamma^t \le 1$. This could be the result of an increased likelihood that the target will not survive as time goes on, due to dangerous environmental factors or a malicious adversary.

This is equivalent to the original game with new probabilities $p'_i = \gamma p_i$, and we obtain

$$\frac{f_{A\cup j}(i)}{f_{A\cup i}(j)} = \frac{p'_j(1-p'_i)}{p'_i(1-p'_j)} = \frac{p_j(1-\gamma p_i)}{p_i(1-\gamma p_j)}.$$

Thus, we can take $z_i = (1 - \gamma p_i)/p_i$, and it follows from Theorem 1 that the solution to the best response problem for $k = 1$ is to search the locations in non-increasing order of $x_i/z_i = x_i p_i/(1 - \gamma p_i)$, as shown directly in Agnetis et al. (2009).

3.1.2. An additive search game

This example is taken from Lidbetter (2013b). In this game, each location i in S has a *search cost* $c_i > 0$ which the Searcher must pay to search it. The objective is to order the locations so as to minimize the sum of the search costs of all the locations searched until all k targets have been found. In order to fit our framework of a decreasing function f, we take $f(A) = \sum_{i\notin A} c_i$ to be the cost of locations *not* searched. In this case, for $i, j \notin A$, we have

$$\frac{f_{A\cup j}(i)}{f_{A\cup i}(j)} = \frac{(-c_i - c_j + \sum_{s\notin A} c_s) - (-c_i - \sum_{s\notin A} c_s)}{(-c_i - c_j - \sum_{s\notin A} c_s) - (-c_j - \sum_{s\notin A} c_s)} = \frac{c_i}{c_j},$$

and we can take $z_i = c_i$. Hence, by Theorem 1, the solution to the best response problem for $k = 1$ is to search the locations in non-increasing order of $x_i/z_i = x_i/c_i$.

The best response problem to this game is a well-known search problem, first considered by Bellman (1957) (Chapter III, Exercise 3, p.90), and the solution is nothing new. It can equivalently be framed as the single machine scheduling problem posed by Smith (1956), now notated in the scheduling literature by $1||\sum w_j C_j$. The rule that the locations should be searched in non-increasing order of x_i/z_i is known as *Smith's Rule*.

3.1.3. A search game on a graph with traveling and search costs

We describe here an application to a game introduced by Baston and Kikuta (2013). The game is played on a graph with vertices S, and the Hider hides at one of the vertices. The Searcher starts at any vertex of her choosing and follows a walk in the network (that is, a sequence of vertices, each one of which is adjacent to the previous vertex). Each time the Searcher traverses an edge e, she pays a cost $d(e)$, and when visiting a vertex i, she can choose to search it for a search cost of c_i. If the Hider is located at a vertex, the Searcher finds him if and only if she pays the search cost. The payoff is the total cost to find the Hider.

Baston and Kikuta (2013) solve the game for graphs that have a Hamiltonian, in the case that the traveling costs $d(e)$ are all equal to 1. Here, we generalize their game to allow multiple targets to be hidden at vertices of the graph, so that the Searcher wants to

minimize the cost of finding all the targets. In the case that the graph is complete (that is, there is an edge between every pair of vertices), the total cost of searching a subset A of vertices is $|A| - 1 + \sum_{i\in A} c_i$. In this case, the game can be modeled by Γ_f, where we let $f(A) = |S - A| + \sum_{i\notin A} c_i$, to ensure the Searcher is the maximizer. Then for $i, j \notin A$, we have

$$\frac{f_{A\cup j}(i)}{f_{A\cup i}(j)} = \frac{(|S-A| - c_i - c_j + \sum_{s\notin A} c_s) - (|S-A| - c_j + \sum_{s\notin A} c_s)}{(|S-A| - c_i - c_j + \sum_{s\notin A} c_s) - (|S-A| - c_i + \sum_{s\notin A} c_s)} = \frac{1+c_i}{1+c_j},$$

so we can put $z_i = 1 + c_i$.

Note that this game is actually equivalent to the game described in the previous subsection, if we take the cost of a location i to be equal to $c_i + 1$.

3.2. Solution to the Game Γ_f

We now present a solution to the game Γ_f when f is **z**-indexable. The solution mirrors the solution of the additive search game considered in Lidbetter (2013b). We will see that in the solution, the Searcher will randomize between mixed strategies which search some subset $A \in S^{(k)}$ first, in an arbitrary order, then choose the other elements of S in a uniformly random order. We denote such a strategy by s_A. Note that the order of the first k elements of the search do not matter, since the Searcher must search at least k locations before finding all k targets.

Lemma 1. *Consider the game Γ_f, for an arbitrary function $f : 2^S \to \mathbb{R}$. For any $A, B \in S^{(k)}$, we have $P_f(A, s_B) = P_f(B, s_A)$.*

Proof. First suppose the Hider uses strategy A and the Searcher uses strategy s_B, and we will calculate the expected payoff $P(A, s_B)$. Observe that the strategy s_B must search all the elements of B before finding the k targets (since these are the first k locations searched) and it must search all the elements of A before finding the targets, because this is where they are located. Each other element of S, contained in the complement $(A\cup B)^c$ is searched with some probability q before the k targets are found. This probability q must be the same for every element of $(A\cup B)^c$, since the elements of B^c are searched in a uniformly random order.

Let X be a random variable equal to $f(A\cup B\cup C)$, where the elements of C are chosen independently with probability q from the set $(A\cup B)^c$. Then, the expected payoff $P(A, s_B)$ is equal to the expectation $\mathbb{E}(X)$.

Clearly, by symmetry, the expected payoff $P(B, s_A)$ is also equal to $\mathbb{E}(X)$. □

We will use a simple but useful lemma, which we state without proof (see Lemma 2.6 of Lidbetter (2013b) for the proof).

Lemma 2. *Consider an arbitrary zero-sum game with a symmetric payoff matrix in which Player I has a mixed strategy x that makes Player II indifferent between all her pure strategies. Then x is an optimal strategy for both Player I and Player II.*

We can now solve the search and rescue game, but to state the solution we will use the following definition.

Definition 2. For a subset $A\subseteq S$, let

$$T_k(A) \equiv \sum_{B\in A^{(k)}} \prod_{i\in B} z_i.$$

Note that

$$T_k(A\cup i)z_j - T_k(A\cup j)z_i = z_j T_k(A) - z_i T_k(A). \tag{3}$$

Theorem 2. *Consider the search game Γ_f, and suppose f is **z**-indexable. Then it is optimal for the Hider to use the mixed strategy q, whereby a set $A \in S^{(k)}$ is chosen with probability q_A given by*

$$q_A \equiv \frac{\prod_{i\in A} z_i}{T_k(S)},$$

It is optimal for the Searcher to use the strategy s that chooses s_A with probability proportional to q_A.

Proof. Suppose the Hider uses the strategy q described in the statement of the theorem. We will show transposing any two adjacent elements of a given search σ leaves the expected payoff unchanged. Since any search can be obtained from σ by a sequence of such transpositions, this is sufficient to prove that all searches have the same expected cost against this Hider strategy.

Thus, suppose in a search σ, the element i comes immediately before j, and let σ' be the search that is the same as σ except that elements i and j are transposed. If j comes in position k or earlier in σ, then clearly transposing i and j leaves the expected payoff unchanged. Otherwise, we can compute the difference in the expected payoffs of the two searches against the Hider strategy q as follows. Let A denote the set of locations searched before i.

$$P(q,\sigma) - P(q,\sigma') = \left(\frac{T_{k-1}(A)z_i}{T_k(S)} f(A\cup i) + \frac{T_{k-1}(A\cup i)z_j}{T_k(S)} f(A\cup\{i,j\})\right) - \left(\frac{T_{k-1}(A)z_j}{T_k(S)} f(A\cup j) + \frac{T_{k-1}(A\cup j)z_i}{T_k(S)} f(A\cup\{i,j\})\right). \tag{4}$$

Considering the coefficient of $f(A\cup\{i,j\})$ in (4), and using (3) with k replaced by $k-1$, we get

$$P(q,\sigma) - P(q,\sigma') = \frac{T_{k-1}(A)}{T_k(S)}\left(z_j f_{A\cup j}(i) - z_i f_{A\cup i}(j)\right) = 0,$$

by (2).

The argument above shows that the value of the game is at most V, where $V = P(q,\sigma)$ is the payoff when the Hider uses the strategy q against any Searcher strategy σ. If we restrict the Searcher to strategies of the form s_A, then the value of the restricted game is equal to V and the strategies s and q are optimal, by Lemmas 1 and 2. Since the value, V of the restricted game is a lower bound for the value of the original (unrestricted) game, the strategies q and s must also be optimal in the original game. □

Theorem 2.1 of Lidbetter (2013b), which gives the solution of the additive search game, follows as a corollary of Theorem 2 of this paper. Our theorem also gives a solution to the search and rescue game. If $k = 1$, there is a particularly simple expression for the value of the game.

Theorem 3. *In the search and rescue game it is optimal for the Hider to choose a subset $A \in S^{(k)}$ with probability*

$$q_A \equiv \lambda_k \prod_{i\in A} \frac{1-p_i}{p_i}, \text{ where } \lambda_k \equiv \left(\sum_{B\in S^{(k)}} \prod_{i\in B} \frac{1-p_i}{p_i}\right)^{-1}.$$

It is optimal for the Searcher to choose a subset $A \in S^{(k)}$ of locations to search first with probability q_A, then search the remaining locations in a uniformly random order. For $k = 1$, the value V of the game is

$$V \equiv \lambda_1\left(1 - \prod_{i\in S} p_i\right) \tag{5}$$

Proof. The optimality of the Hider's and Searcher's strategies follows immediately from Theorem 2. To prove the correctness of (5), it is sufficient to show that against the given Hider strategy, some (and therefore, every) search has expected payoff V. Let σ be the

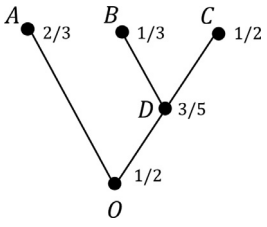

Fig. 1. A tree with probabilities p_v indicated.

search that chooses the locations in increasing order from 1 to n. Then

$$\begin{aligned} P(q,\sigma) &= \sum_{i=1}^{n} q_i \prod_{j\le i} p_j \\ &= \lambda_1 \sum_{i=1}^{n} \frac{1-p_i}{p_i} \prod_{j\le i} p_j \\ &= \lambda_1 \sum_{i=1}^{n} (1-p_i) \prod_{j<i} p_j. \end{aligned}$$

The sum is telescopic, and reduces to the expression on the right-hand side of (5). □

4. The search and rescue game on a graph

In this section we consider the search and rescue game played on a graph. That is, we assume that the set of hiding locations S are the vertices $V(G)$ of a graph G with edge set $E(G)$, which is a collection of unordered pairs of vertices (i, j), $i \neq j$. The Searcher begins at a specified vertex O of the graph, called the *root*, and can search the graph using an *expanding search*, which is a search paradigm introduced by Alpern and Lidbetter (2013). An expanding search of a graph G with root O is an ordering of the vertices S, starting with O, such that each vertex in the ordering is adjacent to some previous vertex.

Formally, a Searcher strategy is a ordering σ of S such that $\sigma(1) = O$ and if $i > 1$, then $(\sigma(i), \sigma(i-1)) \in E(G)$. In this section we will refer to such an ordering simply as a *search*. If α is the restriction of some search σ to some set $\{i, i+1, \dots, j\}$, then we call α a *subsearch* of the vertices $A = \sigma(\{i, i+1, \dots, j\})$. In other words, α describes the sequence of vertices in positions i through j in the search.

The only difference between the game played on a graph and the original version of the game described in Section 2 is that the Searcher's strategy set is restricted. In particular, a Hider strategy is still a probability distribution on S, given by a vector $\mathbf{x} \in \mathbb{R}^n$; the probability that the Searcher is not captured when she searches a vertex v is denoted by p_v; the payoff of the game is calculated in the same way. We relax the restriction $p_v \in (0, 1)$ slightly, by allowing p_v to take the value 1 as long as $v \neq O$ is not a leaf (a vertex of degree 1), otherwise it could be removed from the graph without changing the value of the game. The reason for this is that it will be convenient later to make the assumption that all vertices have degree at most 3, and we will justify this assumption adding vertices v with $p_v = 1$ to the tree to obtain an equivalent game on a tree with the desired property.

Playing the game on a network complicates things, and we therefore restrict ourselves to the case $k = 1$, leaving larger values of k for future work.

We illustrate expanding search with an example. Consider the tree in depicted in Fig. 1. The vertices are labeled with letters, and the probabilities p_i are shown next to the vertices. One possible expanding search on this network visits the vertices in the order O, D, A, B, C. If the target is located at B, say, then the payoff is $(1/2) \cdot (3/5) \cdot (2/3) \cdot (1/3) = 1/15$.

Note that if $p_O = 1$ and the graph is a star (that is, a tree where O is the only vertex of degree greater than 1), then an expanding search of the graph corresponds simply to an (unrestricted) ordering of all the leaf vertices. The search and rescue game played on such a graph is therefore equivalent to the game without any network structure. Equivalently, the game could be played on a complete graph.

In order to solve the search and rescue game played on a tree, we define something similar to the index z_i in Section 3. First we introduced some more notation. For a subset $A \subseteq S$, let $\pi(A)$ denote $\prod_{i \in A} p_i$. If G is a graph, we may write $\pi(G)$ to express $\pi(V(G))$. We also expand the definition of the payoff function P. For a subsearch α of vertices $\sigma(\{i, i+1, \dots, j\})$, and a fixed Hider strategy $\mathbf{x}$, let

$$P(\mathbf{x}, \alpha) = x_{\alpha(i)} p_{\alpha(i)} + x_{\alpha(i+1)} p_{\alpha(i)} p_{\alpha(i+1)} + \cdots + x_{\alpha(j)} p_{\alpha(i)} \cdots p_{\alpha(j)}.$$

In particular, if σ is a search of the whole of S, then $P(\mathbf{x}, \sigma)$ is the payoff of the game when the Hider uses $\mathbf{x}$. We will usually drop the $\mathbf{x}$ in $P(\mathbf{x}, \alpha)$, and simply write $P(\alpha)$ when there is no ambiguity.

We remark that the solution of the best response problem for the game played on a tree follows from the work of Monma and Sidney (1979), since the function P can be easily shown to satisfy what the authors call the *series-network decomposition property*. We give the solution here to the *game* played on a tree.

For a fixed Hider strategy $\mathbf{x}$ and subsearch α of vertices A with $\pi(A) \neq 1$, we define an index

$$I(\alpha) \equiv I_{\mathbf{x}}(\alpha) \equiv P(\alpha)/(1 - \pi(A)).$$

The restriction $\pi(A) \neq 1$ insures that $I(\alpha)$ is well defined. Note that if α consists of a single element i, then the index of α is equal to $x_i p_i/(1 - p_i)$, which is the same as the index determining the optimal search in the best response problem for the game played with no network structure. We now prove a more general lemma, which says that if two subsearches are disjoint and can be executed consecutively in some order, then the subsearch with the highest index should come first.

Lemma 3. *Let* $\mathbf{x}$ *be some fixed Hider strategy and let* σ *be a search. Suppose some subsearch* α *of* σ *searches a subset* $A \subseteq S$ *of vertices immediately before some other subsearch* β *searches a subset* $B \subseteq S$ *disjoint from* A*, with* $\pi(A), \pi(B) \neq 1$*. Let* σ' *be the same as* σ *except that the order of* α *and* β *are transposed. Then* $P(\sigma) \le P(\sigma')$ *if and only if*

$$I(\alpha) \ge I(\beta), \tag{6}$$

with $P(\sigma) = P(\sigma')$ *if and only if (6) holds with equality.*

Proof. Let $C \subseteq S$ be the set of locations searched immediately before α in σ. Then

$$\begin{aligned} P(\sigma') - P(\sigma) &= (\pi(C)P(\alpha) + \pi(C \cup A)P(\beta)) \\ &\quad - (\pi(C)P(\beta) + \pi(C \cup B)P(\alpha)) \\ &= \pi(C)(1 - \pi(A))(1 - \pi(B))(I(\alpha) - I(\beta)), \end{aligned}$$

by definition of $I(\alpha)$ and $I(\beta)$. The lemma follows immediately. □

Lemma 3 is a variation of the *Search Density Lemma*, found in various guises in studies of other search games, for example in Alpern (2010), Alpern and Lidbetter (2013) and Fokkink, Lidbetter, and Végh (2019).

We present a solution to the game on a tree. We may assume that the maximum degree of any vertex of the tree is 3. If not, then by successively adding vertices v with $p(v) = 1$, we can iteratively transform the tree into a tree with degree at most 3 such that the value of the game is the same on both trees and there is a one-to-one correspondence between strategies on one and on the other. Thus, any solution of the game for the transformed tree can be

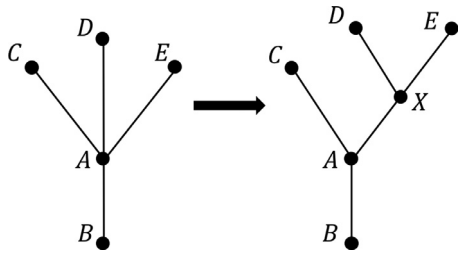

Fig. 2. Transformation of a degree 4 vertex.

mapped back onto the original tree. For example, suppose Fig. 2 depicts a subgraph of a particular tree: in particular, the degree 4 vertex A and its neighbors. Assume that B is closer to O than A. In other words, the path from A to O contains the vertex B, so that in any search of the tree, B must appear before A. Then the subgraph depicted on the left of the figure can be replaced with the subtree shown on the right, where $p_X = 1$, so that there is zero probability that the Searcher will be captured after she searches the vertex X. Strategies on the new tree map onto strategies on the original one in a natural way, with no alteration to the payoffs.

We recursively define a strategy $h \equiv h_G$ for the Hider on a tree G which we will later prove is optimal. We also define recursively a quantity V_G, which we will later prove is the value of the game. We first define the *branches* of a tree G with root O as the connected components of the tree obtained when O and its incident edges are removed from G. The roots of the branches are the neighbors of O. We call vertices v with degree 3 *branch vertices*. We also refer to O as a branch vertex if it has degree 2.

Definition 3 (Tree hiding strategy)**.** The Hider strategy h_G is defined recursively as follows for rooted trees G. If G has only has one vertex v, then let $h_G(v) = 1$ and $V_G = p_v$. If G has more than one vertex, there are two cases, depending on whether or not O is a branch vertex.

Case 1. The root O is not a branch vertex. In this case, let G' be the unique branch of G and let O' be its root. The Hider strategy h_G on G is given by $h_G(O) = 0$ and $h_G(v) = h_{G'}(v)$ for all vertices v in G'. Let $V_G = p_O V_{G'}$.

Case 2. The root O is a branch vertex. In this case, let G_1 and G_2 be the branches of G and let O_1 and O_2 be their roots. Then we set

$$h_G(G_i) = \lambda_G\left(\frac{1-\pi(G_i)}{V_{G_i}}\right), i = 1, 2,$$

where

$$\lambda_G = \left(\frac{1-\pi(G_1)}{V_{G_1}} + \frac{1-\pi(G_2)}{V_{G_2}}\right)^{-1}$$

is a normalizing factor. Then for a vertex v in G_i, we set

$$h_G(v) = h_G(G_i)h_{G_i}(v).$$

We also set

$$V_G = p_O\lambda_G(1-\pi(G_1)\pi(G_2)).$$

Note that the support of h is the set of leaves of G (excluding O, if it is a leaf). It is obvious that the support of any optimal strategy must be the set of leaves, because all other vertices are dominated by some leaf.

We illustrate the computation of h_G and V_G for the tree depicted in Fig. 1. For any vertex v, let $G(v)$ be the subtree of G containing all vertices whose path to O contains v. Using $V_{G(B)} = 1/3$ and $V_{G(C)} = 1/2$, we compute

$$\lambda_{G(D)} = \left(\frac{1-\pi(G(B))}{V_{G(B)}} + \frac{1-\pi(G(C))}{V_{G(C)}}\right)^{-1}$$
$$= \left(\frac{1-1/3}{1/3} + \frac{1-1/2}{1/2}\right)^{-1} = 1/3.$$

Hence, we find that

$$h_{G(D)}(B) = \lambda_{G(D)}\left(\frac{1-\pi(G(B))}{V_{G(B)}}\right) = (1/3)\left(\frac{1-1/3}{1/3}\right) = 2/3 \text{ and}$$
$$h_{G(D)}(C) = \lambda_{G(D)}\left(\frac{1-\pi(G(C))}{V_{G(C)}}\right) = (1/3)\left(\frac{1-1/2}{1/2}\right) = 1/3.$$

Also,

$$V_{G(D)} = p_D\lambda_{G(D)}(1-\pi(G(B))\pi(G(C)))$$
$$= (3/5)(1/3)(1-(1/3)(1/2)) = 1/6.$$

Now using $V_{G(A)} = 2/3$, we compute

$$\lambda_G = \left(\frac{1-\pi(G(A))}{V_{G(A)}} + \frac{1-\pi(G(D))}{V_{G(D)}}\right)^{-1}$$
$$= \left(\frac{1-2/3}{2/3} + \frac{1-(1/3)(1/2)(3/5)}{1/6}\right)^{-1} = 10/59.$$

Hence,

$$h_G(A) = \lambda_G\left(\frac{1-\pi(G(A))}{V_{G(A)}}\right) = (10/59)\left(\frac{1-2/3}{2/3}\right) = 5/59 \text{ and}$$
$$h_G(G(D)) = \lambda_G\left(\frac{1-\pi(G(D))}{V_{G(D)}}\right) = (10/59)\left(\frac{1-(1/3)(1/2)(3/5)}{1/6}\right)$$
$$= 54/59.$$

It follows that

$$h_G(B) = h_G(G(D))h_{G(D)}(B) = (54/59)(2/3) = 36/59 \text{ and}$$
$$h_G(C) = h_G(G(D))h_{G(D)}(C) = (54/59)(1/3) = 18/59.$$

The value of the game is

$$V_G = p_O\lambda_G(1-\pi(G(A))\pi(G(D)))$$
$$= (1/2)(10/59)(1-(2/3)(1/3)(1/2)(3/5)) = 14/177.$$

This completes the calculation of the tree hiding strategy for the tree in Fig. 1.

To show that the Hider strategy h_G guarantees an expected payoff of at most V_G, we first show that any depth-first search of G has expected payoff V_G against h_G, where the formal definition of a depth-first search is as follows.

Definition 4 (Depth-first search)**.** Let G be a tree with root O. A depth-first search of G is a search such that for any vertex v, all the other vertices of $G(v)$ appear in the search in some order immediately after v.

Lemma 4. *Suppose the Hider is located on a tree G according to the tree hiding strategy. Then any depth-first search σ of G has expected payoff $P(\sigma) = V_G$.*

Proof. Let σ be a depth-first search of G. The proof is by induction on the number of vertices of G. The lemma is trivially true when there is only one vertex, so suppose G has at least 2 vertices. There are two cases, depending on whether or not O is a branch vertex. First suppose O is not a branch vertex, in which case let G' be its one branch, with root O'. By the induction hypothesis, the expected payoff of any depth-first search of G' is V_G. Then clearly the expected payoff of σ is

$$P(\sigma) = p(O)V_{G'} \equiv V_G,$$

by definition of V_G.

In the other case, O is a branch vertex and let G_1 and G_2 be the branches of G, with roots O_1 and O_2, respectively. By the induction hypothesis, any depth-first search of G_i has expected payoff V_{G_i} for $i = 1, 2$. Suppose, without loss of generality, that σ performs successive depth-first searches σ_1 and σ_2 of G_1 and G_2 in that order, after searching O. Then the expected payoff of σ is

$$P(\sigma) = p_O(P(\sigma_1) + \pi(G_1)P(\sigma_2)).$$

By induction, $P(\sigma_i) = h_G(G_i)V_{G_i}$ for all $i = 1, 2$, and by definition of h, we have $h_G(G_i) = \lambda_G(1 - \pi(G_i))/V_{G_i}$. Hence,

$$\begin{aligned} P(\sigma) &= p_O\lambda_G((1 - \pi(G_1) + \pi(G_1)(1 - \pi(G_2))) \\ &= p_O\lambda_G(1 - \pi(G)) \equiv V_G. \end{aligned}$$

□

In order to show that the value of the game is bounded above by V_G, it is sufficient to prove that depth-first searches are best responses to the Hider strategy h_G. This follows from the fact that the indices of both branches of a branch vertex are equal. We prove both of these next.

Lemma 5. *Suppose the Hider is located on a tree G according to the strategy h_G.*

(i) *Suppose v is a branch vertex and let σ_1 and σ_2 be depth-first searches of the branches G_1 and G_2 of $G(v)$. Then $I(\sigma_1) = I(\sigma_2)$.*
(ii) *Any depth-first search is a best response to the tree hiding strategy h_G, and h_G ensures the expected payoff of the game is at most V_G.*

Proof. For part (i), by Lemma 4, we have that $P(\sigma_i) = h_G(G_i)V_{G_i} = h_G(G(v))h_{G(v)}(G_i)V_{G_i}$ for $i = 1, 2$. By definition of $h_{G(v)}(G_i)$, we have

$$P(\sigma_i) = h_{G(v)}\lambda_{G(v)}(1 - \pi(G_i)).$$

Hence,

$$I(\sigma_i) = \frac{P(\sigma)}{1 - \pi(G_i)} = h_{G(v)}\lambda_{G(v)}.$$

This expression is independent of i.

For part (ii), suppose there exists a best response σ to h that is not depth-first. Then there must exist branch vertex v such that the two branches G_1 and G_2 of $G(v)$ are not searched consecutively, but the subsearches of the branches are both depth-first. Without loss of generality, assume that G_1 is searched before G_2. We may also assume that v appears in the search immediately before G_1, since if some other vertex w were searched immediately before G_1, then the order of search of v and w could be swapped, and the expected payoff would not be any greater.

So there must exist consecutive subsearches σ_1, τ, σ_2, where σ_1 and σ_2 are depth-first searches of G_1 and G_2 and τ is a subsearch of some other subset A of vertices, disjoint from $G(v)$.

Since π is optimal, by Lemma 3, we must have

$$I(\sigma_1) \geq I(\tau) \geq I(\sigma_2) \tag{7}$$

But, by part (i), we have $I(\sigma_1) = I(\sigma_2)$, and it follows that all the inequalities in (7) hold with equality.

Therefore, by Lemma 3, the search σ' that results from the subsearches σ_1 and τ being swapped has the same expected payoff as σ, and is therefore a best response to h_G.

But σ' searches $G(v)$ in a depth-first manner, and applying this argument repeatedly, we can transform σ into a depth-first search which is also a best response to h_G. By Lemma 4, every depth-first search has the same expected payoff, and is therefore a best response to h_G, and the Hider can ensure that the value of the game is at most V_G. □

We now define the strategy mixed strategy $s = s_G$ that will turn out to be optimal for the Searcher. Similarly to the Hider's strategy, we define it recursively. We first need a technical lemma to ensure that the strategy s_G is well defined.

Lemma 6. *For any tree G, we have that $1 \geq V_G \geq \pi(G)$.*

Proof. We prove this by induction on the number of vertices of G. It is clearly true when G has only one vertex, so suppose G has more than one vertex and that the lemma is true for trees with fewer vertices than G.

First suppose that O is not a branch vertex and let O' be the neighbor of O. Then by the induction hypothesis, $1 \geq V_{G'} \geq \pi(G')$, so $V_G \equiv p(O)V_{G'} \geq p(O)\pi(G') = \pi(G)$. Also, clearly $V_G \leq p(O) \leq 1$.

Now suppose that O is a branch vertex, and let G_1 and G_2 be the two branches. Then by the induction hypothesis, $V_{G_1} \geq \pi(G_1)$ and $V_{G_2} \geq \pi(G_2)$, and it follows that

$$\begin{aligned} \lambda(G) &\geq \left(\frac{1 - \pi(G_1)}{\pi(G_1)} + \frac{1 - \pi(G_2)}{\pi(G_2)}\right)^{-1} \\ &= \frac{\pi(G_1)\pi(G_2)}{\pi(G_2)(1 - \pi(G_1)) + \pi(G_1)(1 - \pi(G_2))} \\ &\geq \frac{\pi(G_1)\pi(G_2)}{1 - \pi(G_1) + \pi(G_1)(1 - \pi(G_2))} \\ &= \frac{\pi(G_1)\pi(G_2)}{1 - \pi(G_1)\pi(G_2)}. \end{aligned}$$

It follows that $V_G \equiv p_O\lambda(G)(1 - \pi(G_1)\pi(G_2)) = \pi(G)$.

Also, by the induction hypothesis, $V_{G_1} \leq 1$ and $V_{G_2} \leq 1$, and it follows from the definition of $\lambda(G)$ that

$$\begin{aligned} \lambda(G) &\leq (1 - \pi(G_1) + 1 - \pi(G_2))^{-1} \\ &\leq (1 - \pi(G_1) + \pi(G_1)(1 - \pi(G_2)))^{-1} \\ &= (1 - \pi(G_1)\pi(G_2))^{-1}. \end{aligned}$$

If follows that $V_G \equiv p_O\lambda(G)(1 - \pi(G_1)\pi(G_2)) \leq 1$. □

Definition 5 (Tree searching strategy)**.** The tree searching strategy s_G is a probabilistic choice of depth-first searches of a tree G, and is fully described by specifying which branch is searched first at each branch vertex. Suppose v is such a vertex, and let G_1 and G_2 be the two branches. Then G_1 is searched first with probability

$$q_{G_1} = \lambda(G)\left(\frac{1}{V_{G_1}} - \frac{\pi(G_2)}{V_{G_2}}\right),$$

otherwise G_2 is searched first.

It is easy to check that $q_{G_1} + q_{G_2} = 1$, so to check that q_{G_1} and q_{G_2} are well defined probabilities, we just need to verify that they are both non-negative. To see that q_{G_1} is non-negative, note that by Lemma 6, we have $1/V_{G_1} \geq 1$ and $\pi(G_2)/V_{G_2} \leq 1$. Similarly for q_2.

Before proving the tree searching strategy guarantees an expected payoff of at least V_G for the Searcher, we illustrate the calculation of the probabilities that define the strategy by considering the tree in Fig. 1. The tree searching strategy is specified by the probability $q_{G(A)}$ of searching $G(A)$ before $G(D)$ and the probability $q_{G(B)}$ of searching $G(B)$ before $G(C)$. The first probability is

$$\begin{aligned} q_{G(A)} &= \lambda(G)\left(\frac{1}{V_{G(A)}} - \frac{\pi(G(D))}{V_{G(D)}}\right) \\ &= \left(\frac{10}{59}\right)\left(\frac{1}{2/3} - \frac{(1/3)(1/2)(3/5)}{1/6}\right) = \frac{9}{59}. \end{aligned}$$

The second probability is

$$q_{G(B)} = \lambda(G(D))\left(\frac{1}{V_{G(B)}} - \frac{\pi(G(C))}{V_{G(C)}}\right) = \left(\frac{1}{3}\right)\left(\frac{1}{1/3} - \frac{1/2}{1/2}\right) = \frac{2}{3}.$$

Lemma 7. *The tree searching strategy s_G ensures an expected payoff of at least V_G against any Hider pure strategy.*

Proof. The proof is by induction on the number of vertices of G (it is obviously true for graphs with one vertex). Suppose that G has more than one vertex, and first suppose that O is not a branch vertex, that O' is the vertex adjacent to O and G' is the tree rooted at O'. Then by induction, the expected payoff is at least $p_{O'}V_{G'} \equiv V_G$.

Now suppose that O is a branch vertex, and let G_1 and G_2 be the two branches of G, so that depth-first searches of these subtrees are performed in some order. Suppose, without loss of generality, that the Hider is located in G_1. Then, by induction, the expected payoff satisfies

$$\begin{aligned} P(s) &\geq q_{G_1}V_{G_1} + q_{G_2}\pi(G_2)V_{G_1} \\ &= \lambda(G)\left(\frac{1}{V_{G_1}} - \frac{\pi(G_2)}{V_{G_2}}\right)V_{G_1} + \lambda(G)\left(\frac{1}{V_{G_2}} - \frac{\pi(G_1)}{V_{G_1}}\right)\pi(G_2)V_{G_1} \\ &= \lambda(G)(1-\pi(G)) \equiv V_G. \end{aligned}$$

This completes the proof. □

Theorem 4. *The value of the game is $V_G \equiv \lambda(G)(1-\pi(G))$. An optimal strategy for the Hider is the tree hiding strategy h_G and an optimal strategy for the Searcher is the tree searching strategy $s(G)$.*

Proof. Suppose the Hider uses the strategy h_G. Then, by Lemma 5, the expected payoff is at most V_G, so this is an upper bound for the value of the game.

On the other hand, if the Searcher uses the strategy s_G then by Lemma 7, the expected payoff is at least V_G, and it follows that the value is at least V_G.

Putting together these two bounds on the value, the theorem follows. □

5. Conclusion

We have introduced a new search game to model search and rescue operations in which there is a threat the Searcher will be captured herself. We solved the game in which an arbitrary number of targets must be captured, and in the case of one target we solved the game when played on the vertices of a graph. There are many open questions and variations of the game that must be left to future work. What is the solution to the game for $k > 1$ on trees, or for $k = 1$ on other classes of graphs? What happens if we relax the assumption that the Searcher starts at a fixed vertex, as in Baston and Kikuta (2013, 2015)? We could also consider a variation of the game in which there is more than one Searcher, similarly to the scheduling problem in Agnetis et al. (2009). This paper scratches the surface, but we believe this opens up an interesting and potential fruitful avenue of new study.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1909446.

References

Agnetis, A., Detti, P., Pranzo, M., & Sodhi, M. S. (2009). Sequencing unreliable jobs on parallel machines. *Journal of Scheduling, 12*(1), 45–54.
Alpern, S. (2010). Search games on trees with asymmetric travel times. *SIAM Journal on Control and Optimization, 48*(8), 5547–5563.
Alpern, S. (2019). Search for an immobile hider in a known subset of a network. *Theoretical Computer Science, 794*, 20–26.
Alpern, S., Baston, V., & Gal, S. (2008). Network search games with immobile hider, without a designated searcher starting point. *International Journal of Game Theory, 37*(2), 281–302.
Alpern, S., & Gal, S. (2003). The theory of search games and rendezvous. In *Kluwer international series in operations research and management science (Kluwer, Boston)* (p. 319).
Alpern, S., & Lidbetter, T. (2013). Mining coal or finding terrorists: The expanding search paradigm. *Operations Research, 61*(2), 265–279.
Alpern, S., & Lidbetter, T. (2015). Optimal trade-off between speed and acuity when searching for a small object. *Operations Research, 63*(1), 122–133.
Alpern, S., & Lidbetter, T. (2019). Approximate solutions for expanding search on general networks. *Annals of Operations Research, 275*(2), 259–279.
Angelopoulos, S., Dürr, C., & Lidbetter, T. (2019). The expanding search ratio of a graph. *Discrete Applied Mathematics, 260*, 51–65.
Baston, V., & Kikuta, K. (2013). Search games on networks with travelling and search costs and with arbitrary searcher starting points. *Networks, 62*(1), 72–79.
Baston, V., & Kikuta, K. (2015). Search games on a network with travelling and search costs. *International Journal of Game Theory, 44*(2), 347–365.
Bellman, R. (1957). *Dynamic programming*. Princeton, NJ: Princeton University Press.
Bertsimas, D., & Niño Mora, J. (1996). Conservation laws, extended polymatroids and multiarmed bandit problems; A polyhedral approach to indexable systems. *Mathematics of Operations Research, 21*(2), 257–306.
Dagan, A., & Gal, S. (2008). Network search games, with arbitrary searcher starting point. *Networks, 52*(3), 156–161.
Fokkink, R., Lidbetter, T., & Végh, L. (2019). On submodular search and machine scheduling. *Mathematics of Operations Research, 44*(4), 1431–1449.
Gal, S. (1979). Search games with mobile and immobile hider. *SIAM Journal on Control and Optimization, 17*(1), 99–122.
Gal, S. (2001). On the optimality of a simple strategy for searching graphs. *International Journal of Game Theory, 29*(4), 533–542.
Gal, S. (2011). Search games. In J. J. Cochran, L. A. Cox, Jr, P. Keskinocak, J. P. Kharoufeh, & J. C. Smith (Eds.), *Wiley encyclopedia of operations research and management science*. Wiley.
Gal, S., & Casas, J. (2014). Succession of hide-seek and pursuit-evasion at heterogeneous locations. *Journal of the Royal Society Interface, 11*(94), 20140062.
Hohzaki, R. (2016). Search games: Literature and survey. *Journal of the Operations Research Society of Japan, 59*(1), 1–34.
Lidbetter, T. (2013a). On the approximation ratio of the random chinese postman tour for network search. *European Journal of Operational Research, 263*(3), 782–788.
Lidbetter, T. (2013b). Search games with multiple hidden objects. *SIAM Journal on Control and Optimization, 51*(4), 3056–3074.
Lin, K. Y., & Singham, D. I. (2016). Finding a hider by an unknown deadline. *Operations Research Letters, 44*(1), 25–32.
Monma, C. L., & Sidney, J. B. (1979). Sequencing with series-parallel precedence constraints. *Mathematics of Operations Research, 4*(3), 215–224.
Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly, 3*(1-2), 59–66.
von Neumann, J. (1928). Zur theorie der gesellschaftsspiele. *Mathematische Annalen, 100*(1), 295–320.