Spyros Angelopoulos, Christoph Dürr and Thomas Lidbetter

The expanding search ratio of a graph

Book section (Published version) (Refereed)

Original citation: Originally published in Ollinger, Nicolas and Vollmer, Heribert, (eds.) 33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016). Leibniz International Proceedings in Informatics (LIPIcs). (47). Wadern, Germany : Schloss Dagstuhl - Leibniz-Zentrum für Informatik GmbH, 2016, pp. 9:1-9:14
DOI: 10.4230/LIPIcs.STACS.2016.9

The Expanding Search Ratio of a Graph*

Spyros Angelopoulos[1], Christoph Dürr[2], and Thomas Lidbetter[3]

1 **Sorbonne Universités, Université Pierre et Marie Curie Paris 06, CNRS, LIP6, Paris, France**
spyros.angelopoulos@lip6.fr
2 **Sorbonne Universités, Université Pierre et Marie Curie Paris 06, CNRS, LIP6, Paris, France**
christoph.durr@lip6.fr
3 **Department of Mathematics, London School of Economics, UK**
t.r.lidbetter@lse.ac.uk

Abstract

We study the problem of searching for a hidden target in an environment that is modeled by an edge-weighted graph. Most of the previous work on this problem considers the *pathwise* cost formulation, in which the cost incurred by the searcher is the overall time to locate the target, assuming that the searcher moves at unit speed. More recent work introduced the setting of *expanding search* in which the searcher incurs cost only upon visiting previously unexplored areas of the graph. Such a paradigm is useful in modeling problems in which the cost of re-exploration is negligible (such as coal mining).

In our work we study algorithmic and computational issues of expanding search, for a variety of search environments including general graphs, trees and star-like graphs. In particular, we rely on the deterministic and randomized *search ratio* as the performance measures of search strategies, which were originally introduced by Koutsoupias and Papadimitriou [ICALP 1996] in the context of pathwise search. The search ratio is essentially the best competitive ratio among all possible strategies. Our main objective is to explore how the transition from pathwise to expanding search affects the competitive analysis, which has applications to optimization problems beyond the strict boundaries of search problems.

1998 ACM Subject Classification F.1.2 Modes of Computation, F.2.2 Nonnumerical Algorithms and Problems, I.2.8 Problem Solving, Control Methods and Search

Keywords and phrases Search games, randomized algorithms, competitive analysis, game theory

Digital Object Identifier 10.4230/LIPIcs.STACS.2016.9

1 Introduction

Searching for a hidden target is a common task in everyday life and a significant computational problem with important applications. The game-theoretic approach to searching considers the search *environment* (e.g., a network represented by an undirected, edge-weighted graph), and defines a game between two players: the *Hider*, for whom a *pure strategy* is a hiding point in the environment; and the *Searcher*, for whom a pure strategy is some choice of how to navigate through the environment. We assume that the Searcher is initially located to a starting point called the *root*, that the Hider is static, and that the environment (e.g., the search graph) is fully known to the Searcher. Note that unlike the setting of pursuit-evasion

* This work was supported by project ANR-11-BS02-0015 "New Techniques in Online Computation" (NeTOC).

games, we assume that the Searcher has no knowledge of the Hider's position. In a zero-sum game formulation of the general search problem, given a pure strategy S of the Searcher and a pure strategy H of the Hider, we define the *cost function* $c(S, H)$ as the total effort incurred by the Searcher in locating the Hider. Since the game is zero-sum, this cost also represents the gain of the Hider. A strategy for the Searcher may be evaluated by calculating the highest cost $c(S) = \sup_H c(S, H)$ paid to locate a Hider at point H, and we define the *minimax value* of the game as the smallest value $\inf_S c(S)$ that this cost can attain. The extension to games where the players may use randomized (or *mixed*) strategies follows the standard game-theoretic framework.

This simple, yet inclusive formulation has provided the mathematical framework for a study of several variants of search games (the textbooks [22, 4, 2] provide a comprehensive summary of important developments in the field over the past three decades). This is due to the versatility in defining the cost (i.e., payoff) function, which makes the game-theoretic framework applicable to many settings. For instance, in much previous work, the cost function is defined as the *search time* $T(S, H)$, that is the time taken for the Searcher following a strategy S to find a target located at a point H. An alternative way of defining the cost function, which we will use in this paper, is the so-called *normalized search time*, defined formally as $\hat{T}(S, H) = T(S, H)/d(H)$, where $d(H)$ is the minimum time for the Searcher to reach H, assuming this point is known to him. This concept of the normalized search time is very useful when searching in unbounded domains, in which $T(S, H)$ can be arbitrarily large. Moreover, this cost formulation is equally applicable to bounded domains (e.g., finite graphs). In particular, it gives rise to the *competitive ratio* measure, also called *search ratio* in this context, which was first addressed by Koutsoupias and Papadimitriou [29]. The name itself is a reference to the well-known competitive analysis of online algorithms, since the position of the Hider is unknown to the Searcher and the normalization parameter $d(H)$ can be seen as the "optimal" cost for locating the Hider assuming full information.

Another aspect of the cost function is related to when the searcher incurs cost, which may vary according to the setting at hand. The usual search paradigm when seeking a target on a network is what we now call *pathwise search*, in which the Searcher follows a continuous, unit-speed path until the target is reached. Very recently, Alpern and Lidbetter [5] introduced a new search paradigm termed *expanding search*, in which, informally, the Searcher may restart the search at any time from any previously reached point. As a concrete application, [5] mentions the problem of mining for coal: here, digging into a new site is far more costly than moving through an area that has already been dug.

In [5] expanding search games were studied assuming non-normalized measures. In this paper, we study the competitive ratio of expanding search, which, following earlier work of Koutsoupias and Papadimitriou [29] on pathwise search we refer to as the search ratio; namely, we assume the normalized measure as defined earlier.

Related work

Following Gal's formalization of network search games [21] in the framework of pathwise search with un-normalized search time, the problem has had considerable attention, for example [35, 34, 23, 16, 1, 10, 11]. Expanding search was introduced in [5] in the setting in which the payoff is the total (un-normalized) cost of finding the Hider. Among other results, [5] solved the game in the case that the network is either a tree, or 2-edge-connected. This model was extended in [30] to a setting in which the Searcher must locate multiple hidden objects.

The competitive ratio of pathwise search was studied in [29], who showed that the problem of computing the optimal search ratio in a given undirected graph is NP-complete (and

MAX-SNP hard to approximate). They also gave a search strategy based on repeated executions of DFS traversals that achieves a constant approximation of the (deterministic) competitive ratio. Similar results can be obtained concerning the *randomized competitive ratio* (assuming that the Searcher randomizes over its strategy space). Connections between graph searching and other classic optimization problems such as the Traveling Salesman problem and the Minimum Latency problem were shown in [8]. The setting in which the search graph is revealed as the search progresses was studied in [18].

A specific search environment that has attracted considerable attention in the search literature is the *star-like* environment. More specifically, in the unbounded variant, the search domain consists of a set of infinite lines which have a common intersection point (the root of the Searcher); this problem is also known as *ray searching*. Ray searching is a natural generalization of the well-known *linear search* problem [13, 12] (informally called the "cow-path problem"). Optimal strategies were initially given by Gal [20] as well as by Baeza-Yates *et al.* [9] and Jaillet and Stafford [24]. Other related work includes the study of randomization [36], [27], multi-Searcher strategies [32], searching with turn cost [17], the variant in which some probabilistic information on target placement is known [24], [25], and the related problem of designing *hybrid algorithms* [26].

Bounded star search, namely the setting in which distance of the target from the root is bounded was studied in [31, 15]. New performance measures were introduced in [28, 33]. The problem of locating a certain number among the many Hiders was studied in [7].

It must be emphasized that star search has applications that are not confined to locating a target (which explains its significance and popularity). Some concrete applications include drilling for oil in a number of different locations [33], as well as the design of algorithms that return acceptable solutions even if interrupted during their execution [14, 6].

Contribution

In this work we study expanding search by means of competitive analysis, assuming a variety of search graphs such as stars, trees, and general edge-weighted, undirected graphs. Our main motivation is to explore how the transition from pathwise to expanding search affects the deterministic and the randomized search ratios. As in [5], we address both the *discrete* and the *continuous* settings. In the discrete setting, the Hider can hide only on the vertices of the graph. In contrast, in the continuous setting the search space is a network with arcs, and the Hider can hide anywhere across an arc.

We begin in Section 2 with the definitions of the (expanding) search ratio and randomized search ratio, both in the continuous and discrete settings. In Section 3 we give a simple optimal algorithm for the deterministic search ratio in the continuous setting, and show that this is also a 2-approximation of the randomized search ratio. In the discrete setting, we show that the problem of finding the optimal (deterministic) search ratio is NP-hard (using a substantially more complicated reduction than for pathwise search in [29]). Applying well-known iterative deepening techniques, we obtain a $4\ln(4) \approx 5.55$ approximation.

Our main technical results, presented in Section 4, apply to the discrete setting where the search graph is a star. Here, it is easy to show that an optimal deterministic search strategy searches the edges of the star in non-decreasing order of length. This strategy is also a 2-approximation of the randomized search ratio. We thus turn our attention to obtaining a better *randomized* search strategy. More precisely, we give a randomized strategy that approximates the randomized search ratio within a factor of $5/4$, representing a significant improvement over the afore-mentioned 2-approximation. Improved approximations via randomization are usually not easy to achieve (see, e.g. [29]). Our result confirms the

intuitive expectation that randomization has significant benefits. Moreover, using game-theoretic techniques we show a tight bound on the randomized search ratio of any n-edge star graph; namely, we show it is at most $(n+1)/2$, with equality if and only if all the edges have the same length. We accomplish this by analyzing an explicit randomized strategy.

As argued earlier, star-search problems have applications that transcend searching. This is indeed the case in expanding search. Consider the following problem: we are given a collection of n boxes, among which only one contains a prize. We can open a box i at cost d_i. We seek a (randomized) strategy for locating the prize, and the randomized search ratio of the strategy is the total expected cost of all opened boxes, divided by the cost of the box that holds the prize. This problem is equivalent to the problem of finding the (randomized) search ratio of a star graph.

Lastly, we show in Section 5 that the principle of searching vertices in non-decreasing order of their distance from the root extends from stars to trees and unweighted graphs, and gives the optimal deterministic search ratio and a 2-approximation for the optimal randomized search ratio.

Since our main objective is to study the algorithmic and computational impact of *re-exploration* due to the transition from pathwise search to expanding search, it is important to compare our results to the best-known bounds in the context of pathwise search (and, specifically, in the discrete model). More precisely, for unweighted graphs, [29] gives asymptotic approximations of the deterministic and randomized search ratios equal to 6 and 8.98, respectively, but its techniques appear to be applicable also to general graphs, at the expense of somewhat larger, but constant approximations. Furthermore, [29] notes that the problems of computing the search ratios of trees are "surprisingly hard". In contrast, for expanding search of unweighted graphs and (weighted) trees we obtain optimal algorithms and a 2-approximation of the deterministic and randomized search ratios, respectively. Going beyond the threshold of 2 requires more sophisticated strategies even for simple environments such as a star. For general graphs, we note that our 5.55 approximation is strict, and not asymptotic. As a last observation, we note that the pathwise and expanding search algorithms appear to depend crucially on the approximability of TSP and the Steiner Tree problem, respectively.

2 Preliminaries

Continuous setting

We begin by defining an *expanding search* on a connected network Q with root O, as introduced in [5]. The network Q consists of nodes and edges, and Q is endowed with Lebesgue measure corresponding to length. The measure of a subset A of Q is denoted by $\lambda(A)$. Let $\mu = \lambda(Q)$ be the total measure of Q.

▶ **Definition 1.** An expanding search on a network Q with root O is a family of connected subsets $S(t) \subset Q$ (for $0 \le t \le \mu$) satisfying: (i) $S(0) = O$; (ii) $S(t) \subset S(t')$ for all $t \le t'$; and (iii) $\lambda(S(t)) = t$ for all t.

Since we will only consider expanding searches in this paper, we refer to a given expanding search as a *search strategy*. For a point $H \in Q$ we write $d(H)$ for the length of the shortest path from O to H. For a given expanding search S of Q and a point $H \in Q$, let $T(S,H) = \min\{t : H \in S(t)\}$ be the *search time* of H under S. For $H \neq O$, let $\hat{T}(S,H)$ be the ratio $T(S,H)/d(H)$ of the search time of H to the distance of H from the root. We refer to $\hat{T}(S,H)$ as the *normalized search time*.

▶ **Definition 2.** The (deterministic) search ratio $\sigma_S = \sigma_S(Q)$ of a search strategy S for a network Q is given by $\sigma_S(Q) = \sup_{H \in Q - \{O\}} \hat{T}(S, H)$. The (deterministic) search ratio, $\sigma = \sigma(Q)$ of Q is given by $\sigma(Q) = \inf_S \sigma_S(Q)$, where the infimum is taken over all search strategies S. If $\sigma_S = \sigma$ we say that S is optimal.

We will also consider randomized search strategies: that is, search strategies that are chosen according to some probability distribution. We denote randomized strategies by lower case letters, and for such a strategy s and point $H \in Q$ we denote the *expected search time* by $T(s, H)$ and the *expected normalized search time* by $\hat{T}(s, H) = T(s, H)/d(H)$.

▶ **Definition 3.** The randomized search ratio $\rho_s = \rho_s(Q)$ of a randomized search strategy s for a network Q is given by $\rho_s(Q) = \sup_{H \in Q - \{O\}} \hat{T}(s, H)$. The randomized search ratio, $\rho = \rho(Q)$ of Q is given by $\rho(Q) = \inf_s \rho_s(Q)$,, where the infimum is taken over all possible randomized search strategies s. If $\rho_s = \rho$ we say that s is optimal.

We can view the randomized search ratio of a network as the value of the following zero-sum game $\Gamma(Q, O)$. A strategy S for the Searcher is a search strategy as described above and a strategy H for the Hider is a point on Q. The payoff of the game is the normalized search time $\hat{T}(S, H)$. For *mixed* (randomized) strategies s and h of the Searcher and Hider, respectively, the expected payoff is denoted by $\hat{T}(s, h)$.

In [5] the authors considered a similar zero-sum game in which the player's strategy sets are the same but the payoff is the unnormalized search time $T(S, H)$. They showed that the strategy sets are compact with respect to the uniform Hausdorff metric and that $T(S, H)$ is lower semicontinuous in S for fixed H. Since $d(H)$ is a constant for fixed H, it follows that $\hat{T}(S, H) = T(S, H)/d(H)$ is also lower semicontinuous in S for fixed H, and by the Minimax Theorem of Alpern and Gal [3], we have the following theorem.

▶ **Theorem 4.** *Let Q be a network with root O. The game $\Gamma(Q, O)$ has a value V, which is equal to the randomized search ratio $\rho(Q)$. The Searcher has an optimal mixed strategy (with search ratio $\rho(Q)$) and the Hider has ϵ-optimal mixed strategies.*

Theorem 4 allows us to find lower bounds for the randomized search ratio, since for any mixed Hider strategy h, we have $\rho(Q) \geq \inf_S \hat{T}(S, h)$.

Discrete setting

In the discrete setting the search environment consists of an undirected, edge-weighted graph $G = (\mathcal{E}, \mathcal{V})$, with $|\mathcal{V}| = n$, and a distinguished root vertex $O \in \mathcal{V}$; moreover, the Hider is always located on some vertex of G. The weight or *length* of edge e, denoted by $\lambda(e)$, represents the time required to search that edge (we assume, via normalization, that $\lambda(e) \geq 1$)). We will call a graph of unit edge weights *unweighted*.

A *search strategy* on G is a sequence of edges, starting from the root, chosen so that the set of edges that have been searched is a connected, increasing set. More precisely:

▶ **Definition 5.** An expanding search S on a graph G is a sequence of edges $e_1, \ldots, e_{n-1}$ such that every *prefix* $\{e_1, \ldots, e_k\}$, $k = 1, \ldots, n-1$ is a subtree of G rooted at O.

For a given vertex $v \in \mathcal{V}$ and a given search strategy $S = (e_1, \ldots, e_{n-1})$, denote by S_v the first prefix $\{e_1, \ldots, e_k\}$ that covers v. The *search time*, $T(S, v)$ of v is the total time $\sum_{e \in S_v} \lambda(e)$ taken to search all the edges before v is discovered. Let $d(v)$ denote the length of the shortest path from O to v, which is the minimum time for the Searcher to discover v. For

$v \neq O$ the *normalized search time* is denoted by $\hat{T}(S, v) = T(S, v)/d(v)$. The deterministic and randomized search ratios are then defined along the lines of Definitions 2 and 3.

We will view the randomized search ratio ρ through the lens of a zero-sum game between a Searcher and a Hider. Unlike the continuous setting, the game is finite. The Searcher's pure strategy set, $\mathcal{S}$ is the set of search strategies and the Hider's pure strategy set is the set $\mathcal{V} - O$ of non-root vertices of G. For a Hider strategy $v \in \mathcal{V}$ and a Searcher strategy $S \in \mathcal{S}$, the payoff of the game is $\hat{T}(S, v)$, which the Hider wants to maximize and the Searcher wants to minimize. By the standard minimax theorem for zero-sum games, the value of the game is equal to the randomized search ratio and an optimal randomized search strategy is an optimal strategy for the Searcher in the game. A mixed strategy for the Hider is a probability distribution h over the vertices of G, and for mixed strategies h and s of the Hider and Searcher respectively, we write $T(s, h)$ and $\hat{T}(s, h)$ for the corresponding expected search time and expected normalized search time.

3 General graphs

3.1 Continuous setting

The optimal deterministic search ratio is simple to compute in the continuous case, using a "uniformly-expanding", BFS strategy. For any $r \geq 0$, we denote the closed disc of radius r around O by $D(r) = \{x \in Q : d(x) \leq r\}$. Consider the real function $f : \mathbb{R}^+ \to \mathbb{R}$ given by $f(r) = \lambda(D(r))$, so $f(r)$ is the measure of the set of points at distance no more than r from the root. The function f is strictly increasing so has an inverse g. The interpretation is that $g(t)$ is the unique radius r for which $D(r)$ has measure t.

For a network Q with root O, consider the expanding search S^* defined by $S^*(t) = D(g(t))$. Thus, $S^*(t)$ is an expanding disc of radius $g(t)$. It is easy to verify that S^* is indeed an expanding search. First we note that $S^*(t)$ is connected, since $D(r)$ is always connected. It also trivially satisfies (i) and (ii) from Definition 1, and (iii) is also satisfied since $\lambda(S^*(t)) = \lambda(D(g(t))) = f(g(t)) = t$.

It is very easy to see that S^* can be implemented in polynomial time. We will show that S^* is optimal. First note that the search time of a point $H \in Q$ under S^* is the unique time t such that $S^*(t) = D(d(H))$, so $T(S^*, H) = \lambda(D(d(H))) = f(d(H))$. Hence

$$\sigma_{S^*} = \sup_{H \in Q - \{O\}} \frac{f(d(H))}{d(H)} = \sup_{r>0} \frac{f(r)}{r} = \frac{1}{\inf_{t>0} \frac{g(t)}{t}}. \tag{1}$$

▶ **Theorem 6.** *The search ratio σ of a network Q with root O is given by $\sigma = \sup_{r>0} \frac{f(r)}{r}.$, for f defined as above. Therefore, the expanding search S^* is optimal.*

We further show that the randomized search ratio is always at least half of the search ratio, which implies that S^* is a 2-approximation of the optimal randomized search strategy.

▶ **Proposition 7.** *For a network Q with root O, the randomized search ratio ρ satisfies $\sigma/2 \leq \rho \leq \sigma$. Furthermore, the bounds are tight.*

3.2 Discrete setting

In this section we show that the problem of computing the (deterministic) search ratio is NP-hard. We also give a search strategy that achieves a $4\ln(4) \approx 5.55$ approximation ratio.

For a given graph G, suppose that S is a search strategy which searches the edges in the order $e_1, \ldots, e_{n-1}$. For a subgraph H of G, denote by $\lambda(H)$ the sum of all the lengths $\lambda(e)$ of edges e in H.

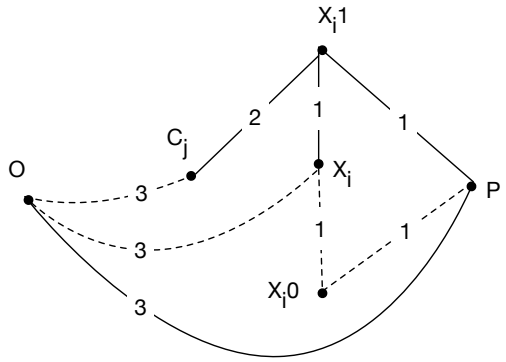

Figure 1 A schematic view of the graph G used in the reduction of Theorem 8.

.

▶ **Theorem 8.** *Given a graph G with root O and a constant $R \geq 0$, it is NP-Complete to decide whether $\sigma(G) \leq R$.*

Proof. The proof is based on a reduction from 3-SAT. Given a 3-SAT instance consisting of n variables and m clauses with $m \geq n$, we construct an instance of our problem.

We construct the graph G consisting of vertices O, P, a vertex C_j for every clause (the *clause vertices*), vertices X_i (the *variable vertices*) and vertices X_i^0, X_i^1 (the *literal vertices*) for every variable. For every $i = 1, \ldots, n$ there are unit length edges of the form $(X_i, X_i^0), (X_i, X_i^1), (P, X_i^0), (P, X_i^1)$. For every variable x_i appearing positively in the j-th clause there is an edge (C_j, X_i^1) of length 2 and for every variable x_i appearing negatively in the j-th clause there is an edge (C_j, X_i^0) of length 2. For every $j = 1, \ldots, m$ there is an edge (O, C_j) of length 3 and for every $i = 1, \ldots, n$ there is an edge (O, X_i) of length 3. Finally, there is an edge (O, P) of length 3. We fix $R = 1 + \frac{2}{3}(n + m)$. The construction is shown in Figure 1.

Note that the vertices can be partitioned according to their distance from O. In particular, vertex P, as well as variable and clause vertices have distances 3, whereas literal vertices have distance 4.

We must show that there exists a boolean assignment to the variables satisfying all clauses if and only if the search ratio of G is at most R.

For the easy direction of the proof, consider a boolean assignment $b \in \{0, 1\}^n$ to the variables satisfying all clauses. We will show that there is a search strategy with search ratio at most R. First we construct a tree H covering all distance 3 vertices with total length $3R$. The tree consists of the edge (O, P), the edges $(P, X_i^{b_i}), (X_i, X_i^{b_i})$ for every $i = 1, \ldots, n$, and for every clause C_j an edge from C_j to the literal vertex corresponding to a literal satisfying the clause. We denote the tree constructed from b by H^b. The total length of H^b is $3 + 2n + 2m$ which is exactly $3R$ by the choice of R. To turn the tree into a search strategy S we order the edges from H by increasing distance from 0. This sequence S is completed in arbitrary order with the remaining edges of the form (X_i, X_i^0) and (X_i, X_i^1). We have $\rho_S(P) = 1, \rho_S(C_j) \leq 3R/3, \rho_S(X_i) \leq 3R/3$ and $\rho(X_i^x) \leq (3R + n)/4 \leq R$ for every i, j, which shows that the search ratio of G is at most R.

For the hard direction, assume that there is a search strategy with search ratio at most R. Let H be its shortest prefix covering all distance 3 vertices. By the definition of the search ratio we know that $\lambda(H) \leq 3R$. Through a sequence of transformations we turn H into a tree of the form H^b with $\lambda(H^b) \leq \lambda(H)$. This will show that b is a satisfying assignment for the formula and complete the proof of the theorem.

- If (O, P) does not belong to H we add it. This must create a cycle, containing an edge of the form (O, v) with $v \neq P$. Now we remove this edge, and obtain a tree of the same length.
- If there is an edge (O, C_j) in H for some j, then we replace this edge by the edges $(C_j, v), (v, P)$, where v is a vertex corresponding to a literal from the j-th clause. Some of the added edges might already have been present. The result is a tree of no greater length.
- If there is an edge of the form (O, X_i) in H for some i, then we replace this edge by the edges $(X_i, X_i^0), (X_i^0, P)$. Again, the result is a tree with of no greater length.
- At this stage we know that O is only connected to P in the tree.
- If there is a vertex C_j connected to several vertices $v_1, \ldots, v_k$ for $k \geq 2$, then we remove the edges $(C_j, v_1), \ldots, (C_j, v_k)$. Hence, the tree now contains k components, each containing some distinct vertex v_i, and only one of them also containing P. Without loss of generality suppose that v_1 and P are in the same component. Then we add (C_j, v_1) back to H and add for each vertex v_i $(i = 2, \ldots, k)$, a length 2 path to P, going through any literal vertex to which v_i is connected. This way we maintain a tree, and do not increase its length (it might even decrease if some of the added edges were already present).
- At this stage we know that every C_j vertex is adjacent to exact one length 2 edge. Also for every $i = 1, \ldots, n$, among the vertices $\{X_i, X_i^0, X_i^1, P\}$ there are at least two edges, one adjacent to X_i and one adjacent to P. The last edge is necessary since otherwise there would be no connection from the vertices $\{X_i, X_i^0, X_i^1\}$ to P, since by the previous point we know that such a path could not go through a clause vertex. Let k be the total number of additional edges that could exist among the vertex sets $\{X_i, X_i^0, X_i^1, P\}$ over all $i = 1, \ldots, n$. Then the total length of H is $3 + 2m + 2n + k$, which by assumption is at most $3R$. By the choice of R we have equality and thus $k = 0$. This shows that H is a tree of the form H^b for some $b \in \{0, 1\}^n$, which a satisfying assignment. ◀

Using an approach similar to the doubling heuristic of [29], we obtain a constant-approximation algorithm. It is worth pointing out that the algorithm doubles the radius, and explores the resulting graph by computing a Steiner tree of the corresponding vertex set (in contrast to pathwise search, in which the resulting graph is simply explored depth-first).

▶ **Theorem 9.** *There is a polynomial-time search algorithm that approximates $\sigma(G)$ within a factor of $4\ln(4) + \epsilon < 5.55$.*

4 Star search

In this section we consider problems related to the search ratio and randomized search ratio of a star graph in the discrete setting. Note that, unlike the generalizations of the cow-path problem [19], our star environment in finite and discrete. Suppose that G is a star graph consisting of n edges $e_1, \ldots, e_n$ of lengths $d_1, \ldots, d_n$ with each e_i incident to the root, O and to a vertex v_i. We assume without loss of generality that $1 = d_1 \leq d_2 \cdots \leq d_n$. An expanding search of such a graph corresponds simply to a permutation of edges (or vertices).

Computing the optimal deterministic search ratio is relatively simple; in particular, it suffices to search the edges in non-decreasing order of length.

▶ **Proposition 10.** *The deterministic search ratio of a star graph G is*

$$\sigma(G) = \max_{j \leq n} \frac{\sum_{i \leq j} d_i}{d_j}. \tag{2}$$

In addition, searching the vertices in the order $v_1, v_2, \ldots, v_n$ is an optimal strategy.

4.1 Randomized approximation of the randomized search ratio

We now turn to the randomized search ratio, $\rho = \rho(G)$ of the star G. Here, the Hider's pure strategy set is the set $\{v_1, \ldots, v_n\}$ of n leaves and the Searcher's pure strategy set is the set of orderings of the edges. We start with a lower bound for ρ which helps us obtain optimal randomized stategies for 2-edge stars (but is inefficient for general stars). For $j = 1, \ldots, n$, let $\mu_j = \sum_{i=1}^{j} d_i$ be the total length of the first j edges and let $D_j = \sum_{i=1}^{j} d_i^2$ be the sum of the squares of the lengths of the first j edges.

▶ **Lemma 11.** *For each $k = 1, \ldots, n$, consider the Hider strategy which chooses a vertex v_j, where $j \leq k$ with probability $p_j = d_j^2/D_k$. Then for any pure Searcher strategy S, we have*

$$\rho_S \geq \hat{T}(S, p) = \frac{1}{2}\left(1 + \frac{\mu_k^2}{D_k}\right). \tag{3}$$

Let $\pi_k = \frac{1}{2}\left(1 + \frac{\mu_k^2}{D_k}\right)$ be the right-hand side of (3). A natural question is whether the randomized search ratio for star graphs is exactly π_n. For instance, it is easy to see that if the star only has two edges, this is the indeed the case and the bound helps us obtain optimal randomized strategies. However, it is not true in general that the randomized search ratio of an n-edge star is π_n, even for $n = 3$.

An immediate consequence of (3) is that the pure search strategy that simply searches all the edges in the order $e_1, \ldots, e_n$ is a 2-approximation of the optimal strategy. However, this relies on a search strategy that is deterministic, so a reasonable question is whether it is possible to obtain a better approximation using a randomized search strategy. This is indeed the case, as we shall give a search strategy that has approximation ratio $5/4$. The idea behind the strategy is to randomize between edges of similar size; moreover, this partition is also determined at random.

For the purpose of the analysis, let t be the smallest integer such that the longest edge has length less than 2^t. Consider the partition of the edges into subsets $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_t$ where the set $\mathcal{A}_i$ consists of all edges of lengths d with $2^{i-1} \leq d < 2^i$, $i \in \mathbb{N}^+$.

Before we define formally our strategy, we observe that the strategy s that randomizes uniformly between all edges in each of the $\mathcal{A}_i$ successively is not efficient. For example, consider the star with edges lengths $d_1 = 1, d_2 = \cdots = d_{n-1} = 2 - \epsilon, d_n = 2$. We have $\hat{T}(s, v_n) \approx n$, whereas the strategy that searches v_1 first before randomizing uniformly between the remaining edges has a randomized search ratio of approximately $n/2$.

▶ **Definition 12** (Randomized deepening strategy)**.** For each $i = 1, \ldots, t$ choose some x_i uniformly at random between 2^{i-1} and 2^i and let $x_0 = 1$ and $x_{t+1} = 2^t$. For $i = 0, \ldots, t$, let $\mathcal{B}_i$ be the set of edges with length in the interval $[x_i, x_{i+1})$. The **randomized deepening strategy**, s randomizes uniformly between all the edges in each $\mathcal{B}_i$ in the order $\mathcal{B}_0, \ldots, \mathcal{B}_t$.

Let α_i be the measure of the edges in $\cup_{j \leq i} \mathcal{A}_j$ and let $\hat{\alpha}_i$ be the measure of the edges in $\mathcal{A}_i$. Let k_i be the sum of the squares of the lengths of the edges in $\cup_{j \leq i} \mathcal{A}_j$ and let $\hat{k}_i$ be the sum of the squares of all the lengths of the edges in $\mathcal{A}_i$. We start with two simple lemmas which will help us bound the expected search time of a vertex.

▶ **Lemma 13.** *The expected measure of the edges in $\mathcal{A}_i$ with length less than x_i is $2\hat{\alpha}_i - \hat{k}_i/2^{i-1}$.*

▶ **Claim 14.** *For any $i = 1, \ldots, k$ we have $2^{i-1}\hat{\alpha}_i \leq \hat{k}_i \leq 2^i \hat{\alpha}_i$.*

▶ **Theorem 15.** *The approximation ratio of the randomized deepening strategy s, as defined in Definition 12 is $5/4$. Namely, $\rho_s \leq (5/4)\rho$.*

Proof. Consider a vertex v at distance d from O. We calculate the expected search time $T(s,v)$ of v. To simplify the analysis, we scale the length of all the edges in the star so that $2 \le d < 4$ and $e \in \mathcal{A}_2$. This means that the shortest edge v_1 of the star may now have length less than 1, and we write $\mathcal{A}_0$ for the set of edges with length less than 1.

Let T_1, T_2, T_3 be the expected time spent searching edges in $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$, respectively, before reaching v. So $T(s,v) = \alpha_0 + T_1 + T_2 + T_3$.

To calculate the expected time T_1, observe that if x_2 is less than d (which happens with probability $(d-2)/2$) then the time is $T_1 = \hat{\alpha_1}$. If x_2 is at least d (which happens with probability $(4-d)/2$) then the expected time spent searching $\mathcal{A}_1$ is the sum of the expected measure of the edges in $\mathcal{A}_1$ with length less than x_1 and half the expected measure of the edges in $\mathcal{A}_1$ with length at least x_1. By Lemma 13, this is $(2\hat{\alpha_1} - \hat{k_1}) + (1/2)(\hat{k_1} - \hat{\alpha_1}) = 3\hat{\alpha_1}/2 - \hat{k_1}/2$. Hence $T_1 = \left(\frac{d-2}{2}\right)\hat{\alpha_1} + \left(\frac{4-d}{2}\right)(3\hat{\alpha_1}/2 - \hat{k_1}/2) = \left(2 - \frac{d}{4}\right)\hat{\alpha_1} - \left(1 - \frac{d}{4}\right)\hat{k_1}$.

To calculate T_3, observe that if x_2 is greater than d then no arcs in $\mathcal{A}_3$ are searched before e. Otherwise, if x_2 is no greater than d (which happens with probability $(d-2)/2$), the expected time spent searching $\mathcal{A}_3$ is half of the expected measure of the edges in $\mathcal{A}_3$ with length less than x_3. So by Lemma 13, $T_3 = \left(\frac{d-2}{2}\right)(\hat{\alpha_3} - \hat{k_3}/8) = \left(\frac{d}{2} - 1\right)\hat{\alpha_3} - \left(\frac{d}{16} - \frac{1}{8}\right)\hat{k_3}$.

Let $f(x)$ be the measure of edges in $\mathcal{A}_2$ with length no greater than x. Then $T_2 = \frac{d}{2} + \int_{x=2}^{d}(f(x) + \frac{1}{2}(\hat{\alpha_2} - f(x))) \cdot \frac{1}{2}dx + \int_{x=d}^{4}\frac{1}{2}f(x) \cdot \frac{1}{2}dx = \frac{d}{2} + \frac{1}{4}(d-2)\hat{\alpha_2} + \int_2^4 \frac{1}{4}f(x)dx = \frac{d}{2} + \left(\frac{d}{4} + \frac{1}{2}\right)\hat{\alpha_2} - \left(\frac{1}{4}\right)\hat{k_2}$. Combining these and rearranging we get: $T(s,v) = \frac{d}{2} + \left(1 - \frac{d}{4}\right)(k_0 - \alpha_0) + \left(\frac{3}{4} - \frac{d}{4}\right)(2\alpha_1 - k_1) + \left(\frac{3}{8} - \frac{d}{16}\right)(4\alpha_2 - k_2) + \left(\frac{d}{16} - \frac{1}{8}\right)(8\alpha_3 - k_3)$. The second of these five terms is negative since $d \le 4$ and $k_0 \le \alpha_0$ (by Claim 14).

Dividing by d, we obtain an expression for the randomized search ratio ρ_s of s:

$$\rho_s \le \frac{1}{2} + \left(\frac{3}{4d} - \frac{1}{4}\right)(2\alpha_1 - k_1) + \left(\frac{3}{8d} - \frac{1}{16}\right)(4\alpha_2 - k_2) + \left(\frac{1}{16} - \frac{1}{8d}\right)(8\alpha_3 - k_3).$$

Let $\rho^{(1)}, \rho^{(2)}, \rho^{(3)}$ denote the first, second and third terms in the above expression, respectively. The following lemma bounds these terms with respect to ρ.

▶ **Lemma 16.** *For $\rho^{(1)}, \rho^{(2)}, \rho^{(3)}$ defined as above we have $\frac{1/2+\rho^{(3)}}{\rho} \le 2 - 4/d$, and $\frac{\rho^{(2)}}{\rho} \le 3/d - 1/2$, and $\frac{\rho^{(1)}}{\rho} \le 3/(2d) - 1/2$.*

Using Lemma 16 we obtain $\frac{\rho_s}{\rho} \le \frac{\rho^{(1)}}{\rho} + \frac{\rho^{(2)}}{\rho} + \frac{1/2+\rho^{(3)}}{\rho} \le (2 - 4/d) + (3/d - 1/2) + (3/(2d) - 1/2) = 1 + 1/(2d) \le 5/4$ (with equality holding when $d = 2$). ◀

4.2 Tight bounds for the randomized search ratio of a star graph

We now consider how large the deterministic and randomized search ratios can be for a star graph G with n edges. From (2), we have $\sigma(G) = \max_{j \le n} \frac{\sum_{i \le j} d_i}{d_j} \le \max_{j \le n} \frac{j d_j}{d_j} = n$. This upper bound is tight if and only if all edges have the same length. In this case it is very easy to see that the randomized search ratio ρ is $(n+1)/2$, and the optimal search strategy is to search the vertices in a uniformly random order. In contrast, it is not so easy to see that $(n+1)/2$ is the largest value that the randomized search ratio can attain for *any* star graph with n edges. We will show that this is indeed the case by inductively defining a particular randomized search strategy whose randomized search ratio is bounded above by $(n+1)/2$. We thus prove that the bound is tight.

For a given star graph G we inductively define a randomized search strategy s_k on the star graph G_k consisting of only the edges $e_1, \ldots, e_k$ with total length μ_k. Having defined the strategy s_k, we will define s_{k+1} as a randomized mix of two strategies, s_{k+1}^+ and s_{k+1}^-.

Table 1 Maximum value of $\hat{T}(s, v)$.

Search strategy, s	Vertex, v	
	v_i for some $i \leq k$	v_{k+1}
s^+_{k+1}	ρ_{s_k}	$\mu_k/d_{k+1} + 1$
s^-_{k+1}	$\rho_{s_k}(1 + d_{k+1}/\mu_k)$	$\mu_k/(2d_{k+1}) + 1 - D_k/(2\mu_k d_{k+1})$

▶ **Definition 17.** Suppose s_k has been defined for some $k = 1, \ldots, n-1$. Let s^+_{k+1} and s^-_{k+1} be randomized search strategies on G_{k+1} defined by:

(i) s^+_{k+1}: follow the strategy s_k on G_k and then search edge e_{k+1}.

(ii) s^-_{k+1}: choose a time t uniformly at random in $[0, \mu_k]$ and denote the edge that is being searched at time t by e. Follow the strategy s_k, but search edge e_{k+1} immediately before searching e.

Before giving the precise definition of s_k, we evaluate the normalized expected search times $\hat{T}(s^+_{k+1}, v_i)$ and $\hat{T}(s^-_{k+1}, v_i)$ in terms of ρ_{s_k} for the vertices v_i with $i = 1, \ldots, k+1$.

First suppose $i \leq k$. Then clearly $\hat{T}(s^+_{k+1}, v_i) \leq \rho_{s_k}$ (with equality for some $i \leq k$). Under s^-_{k+1}, with probability $T(s_k, v_i)/\mu_k$ edge e_{k+1} is searched before e_i, so the expected search time of v_i is $T(s_k, v_i) + (T(s_k, v_i)/\mu_k)d_{k+1}$. Hence $\hat{T}(s^-_{k+1}, v_i) = \frac{T(s_k,v_i)+(T(s_k,v_i)/\mu_k)d_{k+1}}{d_i} = \hat{T}(s_k, v_i)(1 + d_{k+1}/\mu_k) \leq \rho_{s_k}(1 + d_{k+1}/\mu_k)$. Now suppose $i = k+1$. Under s^+_{k+1}, the time taken to find the Hider is $\mu_k + d_{k+1}$, so $\hat{T}(s^+_{k+1}, v_{k+1}) = \mu_k/d_{k+1} + 1$. Under s^-_{k+1}, the expected search time is $\mu_k/2 + d_{k+1}$ minus a random correction error which depends upon which edge e is being searched under s_k at the random time t chosen uniformly in $[0, \mu_k]$. The edge e is e_i with probability d_i/μ_k, and in this case the expected value of the correction error is $d_i/2$. Hence the expected value of this correction error is $\sum_{i=1}^{k}(d_i/\mu_k) \cdot (d_i/2) = D_k/(2\mu_k)$. So we have $\hat{T}(s^-_{k+1}, v_{k+1}) = \frac{\mu_k/2+d_{k+1}-D_k/(2\mu_k)}{d_{k+1}} = \mu_k/(2d_{k+1}) + 1 - D_k/(2\mu_k d_{k+1})$.

To sum up, the expected search ratio for each combination of strategies can be bounded above by the payoffs in Table 1. We can then proceed to define s_n.

▶ **Definition 18.** Let s_1 be the only strategy available on G_1. Suppose s_k has already been defined on G_k for some $k = 1, \ldots, n-1$. The strategy s_{k+1} is an optimal mixture of s^+_{k+1} and s^-_{k+1} in the zero-sum game with payoff matrix given by Table 1.

The search ratio of s_n can be calculated iteratively, since the search ratio $\rho_{s_{k+1}}$ of s_{k+1} is the value of the game with payoff matrix given by Table 1, for each $k = 1, \ldots, n-1$. We use this to show that $\rho_{s_n} \leq (n+1)/2$.

▶ **Theorem 19.** *The randomized search ratio ρ of star network G with n edges is at most $(n+1)/2$, with equality if and only if all the edges have the same length.*

Proof. We have already pointed out that $\rho = (n+1)/2$ for the star whose edges all have the same length. To show that $\rho \leq (n+1)/2$ we use induction on the number of edges to show that $\rho_{s_n} \leq (n+1)/2$. It is clear that for $k = 1$, we have $\rho_{s_k} = 1 = (k+1)/2$, so assume that $\rho(s_k) \leq (k+1)/2$ for some $k > 1$ and we will show that $\rho_{k+1} \leq (k+2)/2 = k/2 + 1$.

First observe that if $d_{k+1} \geq 2\mu_k/k$ then the Searcher can ensure a payoff of no more than $k/2 + 1$ in the game in Table 1 just by using strategy s^+_{k+1}. This is because the payoff ρ_{s_k} against a vertex v_i with $i \leq k$ is no more than $(k+1)/2$ by the induction hypothesis and the payoff against v_{k+1} is $\mu_k/d_{k+1} + 1 \leq k/2$.

So assume that $d_{k+1} \leq 2\mu_k/k$, and note also that $d_{k+1} \geq \mu_k/k$, since the lengths of the edges are non-decreasing and d_{k+1} must be at least the average length of edges $e_1, \ldots, e_k$.

Table 2 Upper bounds for $\hat{T}(s, v)$.

Search strategy, s	Vertex, v	
	v_i for some $i \leq k$	v_{k+1}
s^+_{k+1}	$(k+1)/2$	$\mu_k/d_{k+1} + 1$
s^-_{k+1}	$(k+1)(1 + d_{k+1}/\mu_k)/2$	$\mu_k/(2d_{k+1}) + 1 - \mu_k/(2kd_{k+1})$

By the induction hypothesis, $\rho_{s_k} \leq (k+1)/2$, so the value of the game with payoff matrix given by Table 1 cannot decrease if we replace ρ_{s_k} with $(k+1)/2$ in the table. The value also does not decrease if we replace $-D_k$ by the maximum value it can take, which is $-\mu^2/k$ (that is, its value when $d_1, \ldots, d_k$ are all equal). In summary, $\rho_{s_{k+1}}$ is no more than the value of the game given in Table 2.

By assumption, against strategy s^+_{k+1}, the best response of the Hider (that is, the highest payoff) is given by choosing vertex v_{k+1}. We show that against strategy s^-_{k+1}, the Hider's best response is to choose a vertex v_i with $i \leq k$. This follows from writing the difference, δ between the payoffs in entries $(2,1)$ and $(2,2)$ of Table 2 as $\delta = (k-1)\frac{\mu_k}{2d_{k+1}}\left(\left(\frac{k+1}{k-1}\right)\left(\frac{d_{k+1}}{\mu_k}\right)^2 + \frac{d_{k+1}}{\mu_k} - 1/k\right)$. The quadratic in (d_{k+1}/μ_k) inside the parentheses is increasing for positive values of d_{k+1}/μ_k, and when $d_{k+1}/\mu_k = 1/k$ the quadratic is positive. Since $d_{k+1}/\mu_k \geq 1/k$, we must have $\delta \geq 0$.

Hence the Hider does not have a dominating strategy in the game in Table 2. It is also clear that the Searcher does not have a dominating strategy, since it is better to search e_{k+1} last if and only if the Hider is at some v_i with $i \leq k$. Therefore the game in Table 2 has a unique equilibrium in proper mixed strategies (that is, the players both play each of their strategies with positive probability). The search ratio $\rho_{s_{k+1}}$ of s_{k+1} is bounded above by the value V of the game, which is easily verified to be $V = k/2 + 1 - \frac{\frac{k}{2}(d_{k+1}/\mu_k - 1/k)^2}{(d_{k+1}/\mu_k)^2 + 1/k}$. This is clearly at most $k/2 + 1$, with equality if and only if $d_{k+1}/\mu_k = k$. Equality is only possible if $d_1 = d_2 = \ldots = d_{k+1} = \mu_k/k$. ◀

5 Trees and unweighted graphs

We conclude with the cases in which the graph is either a tree or an unweighted graph (in the discrete setting). If G is a graph with root O, for any $r > 0$ let G_r be the sub-graph of G with vertex set $\mathcal{V}_r$ consisting of all the vertices in G of distance no more than r from the root and with edge set $\mathcal{E}_r$ consisting of all the edges in $\mathcal{E}$ adjacent to some vertex in $\mathcal{V}_r$. The following proposition generalizes Proposition 10.

▶ **Proposition 20.** *Let G be a rooted graph and suppose that G is a tree or an unweighted graph. Then the search ratio σ is given by $\sigma = \sup_{r>0} \frac{\lambda(G_r)}{r}$. An optimal search strategy is to search the vertices in non-decreasing order of their distance from the root.*

We also generalize Proposition 7 and make it applicable to the discrete setting:

▶ **Proposition 21.** *The randomized search ratio of a tree or unweighted graph satisfies $\sigma/2 \leq \rho \leq \sigma$.*

References

1 S. Alpern, V. Baston, and S. Gal. Network search games with immobile hider, without a designated searcher starting point. *International Journal of Game Theory*, 37(2):281–302, 2008.

2 S. Alpern, R. Fokkink, L. Gąsieniec, R. Lindelauf, and V. S. Subrahmanian, editors. *Search theory. A game theoretic perspective.* New York, NY: Springer, 2013.

3 S. Alpern and S. Gal. A mixed strategy minimax theorem without compactness. *SIAM Journal on Control and Optimization*, 26(6):1357–1361, 1988.

4 S. Alpern and S. Gal. *The theory of search games and rendezvous.* Kluwer Academic Publishers, 2003.

5 S. Alpern and T. Lidbetter. Mining coal or finding terrorists: The expanding search paradigm. *Operations Research*, 61(2):265–279, 2013.

6 S. Angelopoulos. Further connections between contract-scheduling and ray-searching problems. In *Proc. of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1516–1522, 2015.

7 S. Angelopoulos, A. López-Ortiz, and K. Panagiotou. Multi-target ray searching problems. *Theoretical Computer Science*, 540:2–12, 2014.

8 G. Ausiello, S. Leonardi, and A. Marchetti-Spaccamela. On salesmen, repairmen, spiders, and other traveling agents. In *Algorithms and Complexity, 4th Italian Conference, CIAC 2000, Rome, Italy, March 2000, Proceedings*, pages 1–16, 2000. URL: `http://dx.doi.org/10.1007/3-540-46521-9_1`, `doi:10.1007/3-540-46521-9_1`.

9 R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Cmputation*, 106:234–244, 1993.

10 V. Baston and K. Kikuta. Search games on networks with travelling and search costs and with arbitrary searcher starting points. *Networks*, 62(1):72–79, 2013.

11 V. Baston and K. Kikuta. Search games on a network with travelling and search costs. *International Journal of Game Theory*, 44(2):347–365, 2015.

12 A. Beck. On the linear search problem. *Naval Research Logistics*, 2:221–228, 1964.

13 R. Bellman. An optimal search problem. *SIAM Review*, 5:274, 1963.

14 D.S. Bernstein, T. J. Perkins, S. Zilberstein, and L. Finkelstein. Scheduling contract algorithms on multiple processors. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI)*, pages 702–706, 2002.

15 P. Bose, J. De Carufel, and S. Durocher. Searching on a line: A complete characterization of the optimal solution. *Theoretical Computer Science*, 569:24–42, 2015.

16 A. Dagan and S. Gal. Network search games, with arbitrary searcher starting point. *Networks*, 52(3):156–161, 2008.

17 E.D. Demaine, S.P. Fekete, and S. Gal. Online searching with turn cost. *Theoretical Computer Science*, 361:342–355, 2006.

18 R. Fleischer, T. Kamphans, R. Klein, E. Langetepe, and G. Trippen. Competitive online approximation of the optimal search ratio. *SIAM Journal on Computing*, 38(3):881–898, 2008.

19 S. Gal. A general search game. *Israel Journal of Mathematics*, 12:32–45, 1972.

20 S. Gal. Minimax solutions for linear search problems. *SIAM J. on Applied Math.*, 27:17–30, 1974.

21 S. Gal. Search games with mobile and immobile hider. *SIAM Journal on Control and Optimization*, 17(1):99–122, 1979.

22 S. Gal. *Search Games.* Academic Press, 1980.

23 S. Gal. On the optimality of a simple strategy for searching graphs. *International Journal of Game Theory*, 29(4):533–542, 2001.

24 P. Jaillet and M. Stafford. Online searching. *Operations Research*, 49:234–244, 1993.

25 M-Y. Kao and M.L. Littman. Algorithms for informed cows. In *Proceedings of the AAAI 1997 Workshop on Online Search*, 1997.

26 M-Y. Kao, Y. Ma, M. Sipser, and Y.L. Yin. Optimal constructions of hybrid algorithms. *Journal of Algorithms*, 29(1):142–164, 1998.

27 M-Y. Kao, J.H. Reif, and S.R. Tate. Searching in an unknown environment: an optimal randomized algorithm for the cow-path problem. *Inform. and Comp.*, 131(1):63–80, 1996.

28 D. G. Kirkpatrick. Hyperbolic dovetailing. In *Proceedings of the 17th Annual European Symposium on Algorithms (ESA)*, pages 616–627, 2009.

29 E. Koutsoupias, C.H. Papadimitriou, and M. Yannakakis. Searching a fixed graph. In *Proc. of the 23rd Int. Colloq. on Automata, Languages and Programming (ICALP)*, pages 280–289, 1996.

30 T. Lidbetter. Search games with multiple hidden objects. *SIAM Journal on Control and Optimization*, 51(4):3056–3074, 2013.

31 A. López-Ortiz and S. Schuierer. The ultimate strategy to search on m rays? *Theoretical Computer Science*, 261(2):267–295, 2001.

32 A. López-Ortiz and S. Schuierer. On-line parallel heuristics, processor scheduling and robot searching under the competitive framework. *Theor. Comp. Sci.*, 310(1–3):527–537, 2004.

33 A. McGregor, K. Onak, and R. Panigrahy. The oil searching problem. In *Proc. of the 17th European Symposium on Algorithms (ESA)*, pages 504–515, 2009.

34 L. Pavlovic. A search game on the union of graphs with immobile hider. *Naval Research Logistics*, 42(8):1177–1199, 1995.

35 J. Reijnierse and J. Potters. Search games with immobile hider. *International Journal of Game Theory*, 21:385–394, 1993.

36 S. Schuierer. A lower bound for randomized searching on m rays. In *Computer Science in Perspective*, pages 264–277, 2003.