On submodular search and machine scheduling

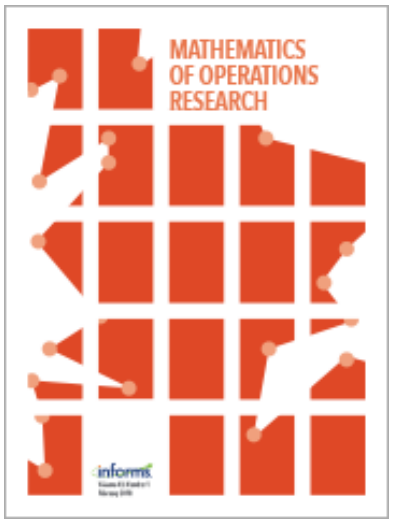
Fokkink, Robbert; Lidbetter, Thomas; Végh, László A.

DOI
10.1287/moor.2018.0978

Publication date
2019

Document Version
Final published version

Published in
Mathematics of Operations Research



Mathematics of Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

On Submodular Search and Machine Scheduling

Robbert Fokkink, Thomas Lidbetter, László A. Végh

To cite this article:
Robbert Fokkink, Thomas Lidbetter, László A. Végh (2019) On Submodular Search and Machine Scheduling. Mathematics of Operations Research 44(4):1431-1449. https://doi.org/10.1287/moor.2018.0978

http://pubsonline.informs.org/journal/moor

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 44, No. 4, November 2019, pp. 1431–1449
ISSN 0364-765X (print), ISSN 1526-5471 (online)

On Submodular Search and Machine Scheduling

Robbert Fokkink,[a] Thomas Lidbetter,[b] László A. Végh[c]

[a] Department of Applied Mathematics, Delft University of Technology, 2628 CD Delft, Netherlands; [b] Department of Management Science and Information Systems, Rutgers Business School, Newark, New Jersey 07102; [c] Department of Mathematics, London School of Economics, London WC2A 2AE, United Kingdom

Contact: r.j.fokkink@tudelft.nl (RF); tlidbetter@business.rutgers.edu, http://orcid.org/0000-0001-6111-2899 (TL); l.vegh@lse.ac.uk, http://orcid.org/0000-0003-1152-200X (LAV)

Received: July 26, 2016
Revised: November 27, 2017; June 9, 2018
Accepted: September 19, 2018
Published Online in Articles in Advance: August 1, 2019

MSC2000 Subject Classification: Primary: 91A43, 90B40, 90B35
OR/MS Subject Classification: Primary: search/surveillance, games/group decisions, production/scheduling

https://doi.org/10.1287/moor.2018.0978

Abstract. Suppose that some objects are hidden in a finite set S of hiding places that must be examined one by one. The cost of searching subsets of S is given by a submodular function, and the probability that all objects are contained in a subset is given by a supermodular function. We seek an ordering of S that finds all the objects with minimal expected cost. This problem is NP-hard, and we give an efficient combinatorial 2-approximation algorithm, generalizing analogous results in scheduling theory. We also give a new scheduling application where a set of jobs must be ordered subject to precedence constraints to minimize the weighted sum of some concave function of the completion times of *subsets* of jobs. We go on to give better approximations for submodular functions with low *total curvature*, and we give a full solution when the problem is what we call *series-parallel decomposable*. Next, we consider a zero-sum game between a cost-maximizing hider and a cost-minimizing searcher. We prove that the equilibrium mixed strategies for the hider are in the base polyhedron of the cost function, suitably scaled, and we solve the game in the series-parallel decomposable case, giving approximately optimal strategies in other cases.

Funding: L. A. Végh was supported by the Engineering and Physical Sciences Research Council [Grant EP/M02797X/1].

Keywords: search games • scheduling • game theory • submodular functions

1. Introduction

Consider a search problem with a finite, nonempty set S of hiding locations, a *cost function* $f : 2^S \to [0, \infty)$, and a *weight function* $g : 2^S \to [0, \infty)$. An ordering, or *search*, π of S must be chosen. For a given ordering π and an element j of S, we denote by $S_j = S_j^\pi$ the union of j and all the locations that precede j in the ordering π. The *search cost* of j under π is $f(S_j^\pi)$. We assume that a hider has hidden some objects in these locations such that if they are searched according to the ordering π, the probability that all the objects are in S_j^π is $g(S_j^\pi)$.

We study two variants of the problem. In the *optimization setting*, the searcher knows the probability distribution used by the hider; that is, she has oracle access to the function g. In the *game setting*, the objects are adversarially hidden, and thus we consider a two-person zero-sum game between the searcher and the hider. In this paper, we restrict our attention to cases where f is submodular and nondecreasing and g is supermodular and nondecreasing.

1.1. The Optimization Setting

The searcher can minimize her expected cost by finding an ordering π that minimizes the expected search cost with respect to f and g, which we write as

$$c(\pi) = \sum_{j=1}^{n} (g(S_j^\pi) - g(S_j^\pi - j)) f(S_j^\pi).$$

We call this problem the *submodular search problem*, and if π minimizes $c(\pi)$, we say that π is *optimal*. The equivalent *submodular ordering problem* was introduced by Pisaruk [40], who showed that in the worst case, the problem takes exponential time, and he gave a 2-approximation algorithm. Our first main result, proved in Section 2, provides a simpler and more direct 2-approximation algorithm. Our key new insight is extending Smith's [49] rule for optimal scheduling to this setting (Theorem 1). This implies that *any* optimal search respects a generalized version of a Sidney [47] decomposition; furthermore, any search that respects this decomposition is a 2-approximation for an optimal search.

We give stronger approximation guarantees for special classes of functions. In Section 2.2, we show that our algorithm performs well when the functions f and g are close to being modular. In particular, we show that the algorithm performs well for low values of the *total curvature* of f and $g^{\#}$ (roughly speaking, the extent to which they differ from modular functions). Here $g^{\#}(A) = g(S) - g(\overline{A})$ is the dual function of g, and $\overline{A}$ denotes the complement of A. Later, in Section 2.3, we introduce the concept of a *series-parallel decomposability* and show how to find an optimal search if the problem is series-parallel decomposable.

1.2. The Game Setting

We then restrict our attention to the case where g is modular. This corresponds to the case of hiding a single object at one of the locations according to a probability distribution $\mathbf{x} \in [0,1]^S$. Thus, $g(A) = \mathbf{x}(A) := \sum_{j \in A} x_j$. We consider the finite zero-sum game between a searcher and a cost-maximizing hider, introduced by Fokkink et al. [19]. A pure strategy for the searcher is a permutation π of S, and a pure strategy for the hider is an element $j \in S$. The payoff is $f(S_j^{\pi})$. We call this the *submodular search game*. Because the strategy sets are finite, the game has a value and optimal mixed strategies. However, the size of the searcher's strategy set complicates the problem of computing these optimal strategies. This is a *search game with an immobile hider in discrete locations*, which is a type of game that has been well studied (see [2, 21, 22]). It is customary to study such games on graphs, and the cost is given by the time taken for the searcher to reach the hider's location from a giving starting point. The alternative approach in our paper is to ignore the graph and focus on the cost function.

We analyze the submodular search game in Section 3, showing that every optimal hider strategy lies in the *base polyhedron* of the scaled cost function $\frac{1}{f(S)} f$ and that any such strategy approximates an optimal strategy by a factor of 2. We go on to give $1/(1-\kappa_f)$-approximate strategies, where κ_f is the total curvature of f (defined precisely in Section 2.2). Finally, we define a notion of series-parallel decomposability for the submodular search game and give a solution in this case.

We do not know the computational complexity of finding equilibrium strategies in the game, and we leave this as an open problem.

1.3. Motivation, Examples, and Previous Work

Here we present a wide range of examples of submodular search in the context of search games and scheduling. Several further applications are described in Pisaruk [40].

1.3.1. Modular Search. To give some intuition for the models, we first consider the submodular search problem in the case that f and g are both modular (call this the *modular search problem*). In this case, for subsets $A \subset S$, we can write $g(A) = \mathbf{x}(A)$ and $f(A) = \mathbf{c}(A)$, for some vectors $\mathbf{x}, \mathbf{c} \in \mathbb{R}^S$. (Note that we are using the symbol $\subset$ to indicate nonstrict set inclusion.) The modular search problem was considered in Bellman [11, chapter III, exercise 3, p. 90], and the solution is easily shown to be that S should be searched in nonincreasing order of the indices x_j/c_j. Blackwell (reported in [35]) considered the more general problem in which each location has an *overlook probability*, that is, the probability that when a location containing the hider is inspected, the hider is not found. An alternative route to the solution of this more complicated problem can be found using Gittins indices for multiarmed bandit processes [24]. Two different solutions to a game-theoretic version of the modular search problem can be found in the more recent works of Alpern and Lidbetter [4] and Lidbetter [33].

1.3.2. Smith's Rule. The modular search problem is equivalent to a single-machine scheduling problem, considered in Smith [49], in which S is a set of jobs, p_j is the *processing time*, and w_j is the *weight* of job j. For a given ordering π of the jobs, the *completion time* C_j of a job j is the sum of its own processing time and the processing times of all jobs that precede it in π. The objective is to order the jobs to minimize the sum $\sum_j w_j C_j$ of the weighted completion times of the jobs. This problem is usually denoted by $1||\sum w_j C_j$, and by writing $p_j = c_j$ and $x_j = w_j$, it clearly fits into the framework of the modular search problem. The solution that the jobs should be completed in nonincreasing order of the indices w_j/p_j is known as *Smith's rule*. Theorem 1 of this paper is a generalization of Smith's rule and says that any optimal search in the submodular search problem must begin with a subset A that maximizes $g(A)/f(A)$.

Smith's rule has also appeared in a different guise in the field of reliability theory. In particular, Gluss [25] and Mitten [38] consider a least-cost fault-detection problem in which n tests can be performed, each of which has a given cost and a given probability of detecting a fault. The object is to order the tests to minimize the expected cost of detecting the fault.

1.3.3. Search on Graphs with a Single Object. Now consider a generalization of the modular search problem that takes place on a graph on vertex set $S \cup \{r\}$. The searcher is initially located at r and the hider is in S according to the probability distribution $\mathbf{x} \in [0,1]^S$. Each edge of the graph has a cost. An *expanding search* of the graph is a sequence of edges, the first of which is incident to r, whereas each other edge is adjacent to some previously chosen edge. For a particular expanding search, the *search cost* of a vertex j is the sum of the costs of each of the edges chosen up to and including the first edge that is incident to j, and the object is to find an expanding search that minimizes the expected search cost. This problem, which we call the *expanding search problem*, was introduced in Alpern and Lidbetter [4]. This paper also considered a game-theoretic version of the problem, which we shall refer to the *expanding search game*, in which an adversary chooses a worst-case distribution $\mathbf{x}$. The expanding search paradigm is motivated by scenarios in which there is negligible cost to resume searching from some previously reached point of the search space, for example, when mining for coal. See Alpern and Lidbetter [4] for further motivations of expanding search.

Consider the expanding search problem on a tree with root r. For a subset A of nonroot vertices, let $f(A)$ be the sum of the costs of all the edges in the minimum cardinality subtree containing $A \cup \{r\}$, and let $g(A) = \mathbf{x}(A)$. Then f is nondecreasing and submodular, g is nondecreasing and modular, and the expanding search problem is equivalent to the submodular search problem for this f and g. The expanding search game is equivalent to the submodular search game. In fact, the problem is series-parallel decomposable, so solutions of both follow immediately from this work.

1.3.4. Single-Machine Scheduling with Precedence Constraints. Both the expanding search problem and the expanding search game on a tree were solved in Alpern and Lidbetter [4], but in fact the expanding search problem is a special case of the single-machine scheduling problem $1|\text{prec}|\sum w_jC_j$ (see, e.g., [32]). This scheduling problem is a generalization of $1||\sum w_jC_j$ for which the ordering of the jobs S must respect some precedence constraints given by a partial order $\prec$ on S so that a job cannot be processed until all the jobs that precede it in the ordering have been completed. Sidney [47] generalized Smith's rule, showing that an optimal schedule must begin with an *initial set* A of jobs that maximizes the ratio $w(A)/p(A)$ (where A is an initial set if for each job $j \in A$, and all jobs preceding j in the precedence ordering are also in A). Applying this principle repeatedly to the remaining jobs in $\overline{A}$ gives rise to what has become known as a *Sidney decomposition* $S = A_1 \cup \ldots A_k$, where if $i < j$, all jobs in A_i must be scheduled before all jobs in A_j.

One usually depicts the partial order on the jobs with a Hasse diagram, which is a directed acyclic graph with vertex set S and edges (s,t) if $s \prec t$ and s is an immediate predecessor of t. In the case that this graph is a tree, Sidney [47] showed that his decomposition theorem could be used to find an optimal schedule (which was rediscovered in the context of the search problem in Alpern and Lidbetter [4]). It was later shown that an optimal schedule can be found in polynomial time for generalized series-parallel graphs [1, 31], as we explain in Section 2.4, and our result in Section 2.3 for series-parallel decomposable problems generalizes this idea.

The connection to the submodular search problem was pointed out in Pisaruk [40]. Define the cost $f(A)$ of a subset A of jobs as the sum $p(\widetilde{A})$ of the processing times of all the jobs in the precedence closure $\widetilde{A}$ of A, and define $g(A) = \sum_{j \in A} w_j$. Then f is nondecreasing and submodular, and g is nondecreasing and modular, and the problem $1|\text{prec}|\sum w_jC_j$ is equivalent to the submodular search problem for this f and g.

The problem $1|\text{prec}|\sum w_jC_j$ is well known to be *NP*-hard [23, 32], which implies that the submodular search problem is NP-hard, and there are many 2-approximation algorithms [6, 12, 13, 26, 34, 41, 45]. Almost all 2-approximations are consistent with a Sidney decomposition, as shown in Correa and Schulz [15]. In particular, any ordering of the jobs consistent with a Sidney decomposition approximates an optimal schedule by a factor of 2. It is also known that there is no polynomial time approximation scheme for the problem unless NP-complete problems can be solved in randomized subexponential time [7]. Furthermore, for any $\varepsilon > 0$, there is no $(2-\varepsilon)$-approximation to the problem unless a slightly stronger version of the unique games conjecture fails [9].

1.3.5. Scheduling with More General Costs. We may also consider the generalization of $1|\text{prec}|\sum w_jC_j$, denoted by $1|\text{prec}|\sum w_jh(C_j)$, in which the object is to minimize the weighted sum of some monotonically increasing function h of the completion times of the jobs. This problem was considered recently in Schulz and Verschae [46], where the authors found an expression in terms of h for the approximation ratio for an arbitrary schedule that is consistent with a Sidney decomposition for the original problem $1|\text{prec}|\sum w_jC_j$. They also showed that for any concave h, this approximate ratio is at most 2. The concavity of the function h corresponds to the machine benefiting from a learning effect or from a continuous upgrade of its resources. However, the authors also noted that an optimal schedule may not follow a Sidney decomposition of this type. For concave h, $1|\text{prec}|\sum w_jh(C_j)$ fits into the submodular search framework, taking $f(A)$ to be $h(p(\widetilde{A}))$ for a subset A of jobs, and

$g(A) = \sum_{j\in A} w_j$. Thus, we find a 2-approximation different from that in Schulz and Verschae [46] and a Sidney decomposition that is necessarily consistent with every optimal schedule. It should also be mentioned that Schulz and Verschae [46] give $(2+\varepsilon)$-approximate algorithms for the more general problem of $1|\text{prec}|\sum h_j(C_j)$.

For arbitrary functions h, nothing is known about the problem $1|\text{prec}|\sum w_j h(C_j)$. Indeed, without any restrictions on h, it is difficult to believe anything can be said in general. If there are no precedence constraints and $h(C_j) = C_j^\beta, \beta \geq 0$, this is the problem $1||\sum w_j C_j^\beta$, as studied in Bansal et al. [10], in which it is shown that the problem of minimizing total weighted completion time plus total energy requirement (see [17, 37]) can be reduced to $1||\sum w_j C_j^\beta, \beta \in (0,1)$. We discuss the problem $1||\sum w_j h(C_j)$ further in Section 2.4, in which we bound the approximation ratio of our algorithm by a simple expression in terms of h.

1.3.6. Expanding Search with Multiple Objects. We now extend the expanding search problem to the setting where multiple objects are hidden. Consider a graph on vertex set $S \cup \{v\}$, with several objects hidden inside S, so that for a subset $A \subset S$, objects are hidden at each of the vertices in A with probability $q(A)$, where $\sum_{A\subset S} q(A) = 1$. The objective is to find an expanding search to minimize the expected time to find *all* the objects. A game-theoretic version of this problem was introduced in Lidbetter [33], but nothing is known about the problem of minimizing the expected time to find multiple objects hidden according to a known distribution. When the graph is a tree, as before, we can define $f(A)$ to be the sum of the costs of all the edges in the minimum cardinality subtree containing $A \cup \{r\}$, and this time define $g(A)$ to be $\sum_{B\subset A} q(B)$. Then g is nondecreasing and supermodular. Thus, this is a submodular search problem, and therefore, we obtain a 2-approximation algorithm.

1.3.7. Scheduling with Subset Weights. There is an analogous extension to the scheduling problem $1|\text{prec}|\sum w_j C_j$. Instead of giving a weight to each job, we give a weight $w_A \geq 0$ to each *subset* A of jobs, and the object is to minimize the sum of the weighted completion times $\sum_{A\subset S} w_A C_A$ of the subsets of the jobs, where C_A is the first time that all the jobs in A have been completed. The motivation for this problem is the prospect that completing certain subsets of jobs could have additional utility. Denote this problem by $1|\text{prec}|\sum w_A C_A$. If the number of nonzero weights w_A is polynomial in n, then $1|\text{prec}|\sum w_A C_A$ can be reduced to $1|\text{prec}|\sum w_j C_j$. Indeed, given an instance of the former problem, for each subset A with positive weight, we can create a dummy job with processing time 0 and weight w_A that is preceded by all jobs in A. The same holds for the further generalization $1|\text{prec}|\sum w_A h(C_A)$, where h is a monotone increasing, concave function of the completion times.

If there is a superpolynomial number of nonzero weights, then the problem $1|\text{prec}|\sum w_A h(C_A)$ still fits into our framework: as before, take $f(A) = h(\widetilde{p}(A))$, and this time let $g(A) = \sum_{B\subset A} w_B$. Note that this requires the assumption that the values $g(A)$ are given by an oracle.

This problem can also be interpreted in the context of searching a directed acyclic graph (given by the Hasse diagram of the partial order). For each subset A of edges, objects are hidden at each of the edges in A with probability $w(A)$ (where $w(S)$ is normalized to be equal to 1). An edge can be searched only if all the edges preceding it in the precedence ordering have been searched, and the cost of searching an edge corresponding to a job j is equal to the processing time p_j. The objective is to minimize the total expected cost of finding all the hidden objects.

The assumption of an oracle could be reasonable if, for example, k objects are hidden uniformly at random on the edges of a directed acyclic graph so that $w(A) = 1/\binom{n}{k}$ if $|A| = k$ and $w(A) = 0$ otherwise. In this case, g is given by $g(A) = \binom{|A|}{k}/\binom{n}{k}$. Equivalently, in the scheduling setting, equal utility could be derived from completing all subsets of k jobs.

1.3.8. The Minimum Linear Ordering Problem. The *minimum linear ordering problem* was studied in Iwata et al. [30]. The problem is to find a permutation π to minimize the sum $\sum_{j=1}^n f(S_j^\pi)$ for a function $f: 2^S \to \mathbb{R}^+$. For a monotone increasing and submodular f, an algorithm is given that finds a permutation that approximates an optimal one within a factor of $2 - 2/(n+1)$. This corresponds to the submodular search problem for $g(A) = |A|$ for all A. The approach of Iwata et al. [30] is quite different from ours and that of Pisaruk [40] and is based on rounding the convex programming relaxation based on the Lovász extension. This technique does not seem to extend easily to the more general setting.

2. The Submodular Search Problem

Let $S = \{1, \ldots, n\}$ be a finite set. A function $f: 2^S \to \mathbb{R}$ is *submodular* if

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$$

for all sets $A, B \subset S$. A function $g: 2^S \to \mathbb{R}$ is *supermodular* if and only if $-g$ is submodular.

We consider the submodular search problem, defined in Section 1, with nondecreasing, nonnegative submodular cost function f and nondecreasing, nonnegative supermodular weight function g. Although we often think of g as defining probabilities, it is simpler not to make the assumption that $g(S) = 1$. An optimal search remains optimal if we add a constant to f, and submodularity is preserved, so we may assume that $f(\emptyset) = 0$ (in other words, f is a polymatroid set function). Similarly, we assume that $g(\emptyset) = 0$. Furthermore, we assume that $f(A) > 0$ for all $A \neq \emptyset$ because it is clear that sets with zero cost must be searched first, and we assume that $g(A) < g(S)$ for all $A \neq S$ because any A with $g(A) = g(S)$ would be searched first. We denote the expected cost of a search π with respect to functions f and g by $c_{f,g}(\pi)$, though we shall usually suppress the subscripts.

A key concept we will use in the paper is that of the *search density* (or simply *density*) of a set $A \subset S$, which is defined as the ratio of the probability that the hider is located in A and the cost of searching A if A is searched first. Search density is a concept that often appears in the theory of search games (see [3, 4, 5]), and a general principle that arises is that it is best to search regions of higher density first. The corresponding inverse ratio of the processing time to the weight of jobs also arises naturally in scheduling theory, particularly in Smith's [49] well-known rule for minimizing the weighted completion time in single-machine scheduling of jobs without precedence constraints. The rule says that the jobs should be executed in nondecreasing order of this ratio. Our 2-approximation for the submodular search problem relies on a key result that there is an optimal search that begins with a maximum density subset of S. Sidney [47] observed this to be the case for the scheduling problem $1|\text{prec}|\sum w_j C_j$.

The proof of our result and the resulting 2-approximation is inspired by the proof of the analogous result in Chekuri and Motwani [12], of which this is a generalization. We emphasize that the 2-approximation found in Chekuri and Motwani [12] was obtained independently by Margot et al. [34]. We also note that the 2-approximation result generalizes a similar result from Fokkink et al. [19], which says that *any* search strategy is a 2-approximation for the equilibrium search strategy in the submodular search game.

Definition 1. The *search density* (or simply *density*) of a nonempty subset $A \subset S$ is defined as

$$\rho(A) = \frac{g(A)}{f(A)}.$$

We denote $\max\{\rho(A) : A \subset S\}$ by ρ^*, and if $\rho(A) = \rho^*$, then we say that A has *maximum search density*, or simply *maximum density*. We put $\rho(\emptyset) = \rho^*$.

Recall that $\mathcal{F} \subset 2^S$ is a *lattice* if $A, B \in \mathcal{F}$ implies that $A \cup B \in \mathcal{F}$ and $A \cap B \in \mathcal{F}$. A nonempty $A \in \mathcal{F}$ is an *atom* if the only proper subset of A in $\mathcal{F}$ is the empty set. Atoms are disjoint, and each element of $\mathcal{F}$ is a union of atoms.

If f_1 and f_2 are set functions on disjoint sets S_1 and S_2, then the *direct sum* $f_1 \oplus f_2$ of f_1 and f_2 over S_1 and S_2 is the set function on $S_1 \cup S_2$ defined by

$$(f_1 \oplus f_2)(A) = f_1(S_1 \cap A) + f_2(S_2 \cap A).$$

The restriction of f to a subset A is denoted by $f|_A$, and that for g is denoted similarly.

In the proof of Lemma 1, and later in the proof of Lemma 9, we use the following observations: if $a, c \geq 0$ and $b, d > 0$, then $\frac{a}{b} \leq \frac{c}{d}$ implies the following:

(i) $\frac{a}{b} \leq \frac{a+c}{b+d} \leq \frac{c}{d}$. Furthermore, if one of these three inequalities is an equality, then all the inequalities are equalities.

(ii) $(a - c)\frac{d}{c} \leq (b - d)$.

Lemma 1. *Let $\mathcal{M}$ be the family of subsets of maximum density, and let M be the union of all the atoms of $\mathcal{M}$. Then $\mathcal{M}$ is a lattice, and the functions $f|_M$ and $g|_M$ are both direct sums over the atoms.*

Proof. If $A, B \in \mathcal{M}, A \neq B$, then $\rho^* = g(A)/f(A) = g(B)/f(B)$ and

$$\rho^* = \frac{g(A) + g(B)}{f(A) + f(B)} \leq \frac{g(A \cup B) + g(A \cap B)}{f(A \cup B) + f(A \cap B)},$$

by the submodularity of f and the supermodularity of g. This inequality is in fact an equality because $\rho(A \cup B)$ and $\rho(A \cap B)$ are both bounded above by ρ^*. It follows that both $A \cup B$ and $A \cap B$ have maximum density. If A and B are atoms, then $A \cap B = \emptyset$, and the equality implies that $f(A) + f(B) = f(A \cup B)$ and $g(A) + g(B) = g(A \cup B)$, so $f|_{A \cup B}$ and $g|_{A \cup B}$ are both direct sums over A and B. Therefore, $\mathcal{M}$ is a lattice, and $f|_M$ and $g|_M$ are direct sums over the atoms. □

We now prove that optimal searches must start with a subset of maximum density, generalizing the analogous result for machine scheduling, as first shown in Sidney [47].

The proof of the theorem relies on the following lemma. For a subset A of S and $s \in A$, we write $d_s g(A)$ for $g(A) - g(A - \{s\})$, for convenience of presentation, so that, for instance,

$$c(\pi) = \sum_{j=1}^{n} d_j g(S_j^\pi) f(S_j^\pi).$$

Lemma 2. *Let $f : 2^S \to \mathbb{R}$ be nondecreasing, and let $g : 2^S \to \mathbb{R}$ be supermodular. If π and π' are two permutations of S, then*

$$c(\pi) \geq \sum_{j=1}^{n} d_j g(S_j^{\pi'}) f(S_j^\pi).$$

Proof. We prove Lemma 2 using an adjacent pairwise interchange argument. Suppose the element $i \in S$ appears before $h \in S$ in π, and suppose that σ and τ are any two permutations of S that are identical except that in σ, the element i appears immediately before h, and in τ, the element h appears immediately before i. In this case, we say that τ can be obtained from σ by a *down-switch*. Let k be the immediate predecessor of i in σ and of h in τ, and let $T = S_k^\tau = S_k^\sigma$. Then

$$\begin{aligned}\sum_{j=1}^{n} d_j g(S_j^\sigma) f(S_j^\pi) - \sum_{j=1}^{n} d_j g(S_j^\tau) f(S_j^\pi) &= (d_i g(S_i^\sigma) - d_i g(S_i^\tau)) f(S_i^\pi) + (d_h g(S_h^\sigma) - d_h g(S_h^\tau)) f(S_h^\pi) \\ &= (g(T \cup \{i,h\}) - g(T \cup \{i\}) - g(T \cup \{h\}) + g(T))(f(S_h^\pi) - f(S_i^\pi)). \qquad (1)\end{aligned}$$

By the monotonicity of f and the supermodularity of g, the left-hand side of (1) is nonnegative.

It is easy to see that every permutation π' can be derived from π by performing a finite number of down-switches, and this proves the lemma. □

We say that A is an *initial segment* of a search strategy π if $A = \{\pi(1), \ldots, \pi(|A|)\}$.

Theorem 1. *Let M be the element of $\mathcal{M}$ of largest cardinality. Then any optimal search π has initial segment M. Furthermore, if $A \in \mathcal{M}$, then there exists an optimal search π' such that A is an initial segment.*

Proof. Let A be any subset of maximum search density. Suppose that an optimal search π starts by searching sets $B_1, A_1, B_2, A_2, \ldots, B_k, A_k \subset S$ in that order before searching the rest of S, where $A_i \subset A$ and $B_i \subset \overline{A}$ for all i, the union $A_1 \cup \ldots \cup A_k$ is equal to A, and B_1 may be the empty set. Let $A^j = A_1 \cup \ldots \cup A_j$ and define B^j similarly.

Define a new search π' that starts by searching $A_1, \ldots, A_k, B_1, \ldots, B_k$ before searching the rest of S in the same order. Within each A_i and B_i, the new search follows the same order as π. For a subset T of S, let $\Delta(T)$ be the difference between the terms corresponding to elements of T in $c(\pi)$ and in $c(\pi')$. We will show that $\Delta \equiv \Delta(S) = 0$.

First, consider any $s \in A_j$. The difference $\Delta(\{s\})$ is

$$\begin{aligned}\Delta(\{s\}) &= d_s g(S_s^\pi) f(S_s^\pi) - d_s g(S_s^{\pi'}) f(S_s^{\pi'}) \\ &= d_s g(S_s^{\pi'})(f(S_s^\pi) - f(S_s^{\pi'})) + (d_s g(S_s^\pi) - d_s g(S_s^{\pi'})) f(S_s^\pi) \\ &\geq d_s g(S_s^{\pi'})(f(A \cup B^j) - f(A)) + (d_s g(S_s^\pi) - d_s g(S_s^{\pi'})) f(S_s^\pi),\end{aligned}$$

by the submodularity of f and the monotonicity of g. Summing over all $s \in A_j$ gives

$$\begin{aligned}\Delta(A_j) &\geq (g(A^j) - g(A^{j-1}))(f(A \cup B^j) - f(A)) + \sum_{s \in A_j} (d_s g(S_s^\pi) - d_s g(S_s^{\pi'})) f(S_s^\pi) \\ &\geq \frac{1}{\rho^*}(g(A^j) - g(A^{j-1}))(g(A \cup B^j) - g(A)) + \sum_{s \in A_j} (d_s g(S_s^\pi) - d_s g(S_s^{\pi'})) f(S_s^\pi). \qquad (2)\end{aligned}$$

The second inequality uses the inequality (ii) above, noting that $1/\rho^* = f(A)/g(A)$. Now consider any $t \in B_j$. The difference $\Delta(\{t\})$ is

$$\begin{aligned}\Delta(\{t\}) &= d_t g(S_t^\pi) f(S_t^\pi) - d_t g(S_t^{\pi'}) f(S_t^{\pi'}) \\ &= d_t g(S_t^{\pi'})(f(S_t^\pi) - f(S_t^{\pi'})) + (d_t g(S_t^\pi) - d_t g(S_t^{\pi'})) f(S_t^\pi) \\ &\geq d_t g(S_t^{\pi'})(f(A^{j-1}) - f(A)) + (d_t g(S_t^\pi) - d_t g(S_t^{\pi'})) f(S_t^\pi)\end{aligned}$$

by the submodularity of f. Summing over all $t \in B_j$ gives

$$\Delta(B_j) \geq (g(A \cup B^j) - g(A \cup B^{j-1}))(f(A^{j-1}) - f(A)) + \sum_{t \in B_j} (d_t g(S_t^\pi) - d_t g(S_t^{\pi'}))f(S_t^\pi)$$
$$\geq \frac{1}{\rho^*}(g(A \cup B^j) - g(A \cup B^{j-1}))(g(A^{j-1}) - g(A)) + \sum_{t \in B_j} (d_t g(S_t^\pi) - d_t g(S_t^{\pi'}))f(S_t^\pi), \tag{3}$$

again using inequality (ii). We now sum these estimates on $\Delta(A_j)$ and $\Delta(B_j)$ over all j. Adding the two sums in the right-hand sides of (2) and (3) and summing over j, we obtain

$$\sum_{j=1}^{n} d_j g(S_j^\pi) f(S_j^\pi) - \sum_{j=1}^{n} d_j g(S_j^{\pi'}) f(S_j^\pi),$$

which is nonnegative by Lemma 2. Hence, Δ, which is equal to the sum over j of the right-hand sides of (2) and (3), satisfies

$$\begin{aligned}\rho^* \Delta &\geq \sum_{j=1}^{k} \big((g(A^j) - g(A^{j-1}))(g(A \cup B^j) - g(A)) + (g(A \cup B^j) - g(A \cup B^{j-1}))(g(A^{j-1}) - g(A))\big) \\ &= \sum_{j \leq k} (g(A^j) - g(A^{j-1})) \sum_{i \leq j} (g(A \cup B^i) - g(A \cup B^{i-1})) \\ &\quad + \sum_{j \leq k} (g(A \cup B^j) - g(A \cup B^{j-1})) \sum_{i \geq j} (g(A^i) - g(A^{i-1})) \\ &= 0,\end{aligned}$$

by swapping the order of summation of one of the double sums.

Therefore, the ordering π' is optimal. Hence, it must be true that $\Delta = 0$ and all inequalities above are equalities. It follows that $\rho(A \cup B^j) = \rho(A) = \rho^*$ for all j, and, in particular, $\rho(A \cup B) = \rho(A \cup B^k) = \rho^*$. We have thus established that if A has maximum search density, then it is a subset of an initial segment $A \cup B$ of maximum density. Therefore, every optimal strategy π searches M first. We have also established that there exists an optimal search that has A as an initial segment. □

2.1. A 2-Approximation

Theorem 1 suggests an approach to constructing an optimal strategy akin to a Sidney [47] decomposition for machine scheduling. First, find a nonempty subset $A \subset S$ of maximum density. By Theorem 1, there is an optimal strategy that begins with the elements of A. Now consider the subproblem of finding an optimal search of $\bar{A}$ with cost function f_A defined for $B \subset \bar{A}$ by $f_A(B) = f(A \cup B) - f(A)$ and weight function g_A defined by $g_A(B) = g(A \cup B) - g(A)$. The function f_A is called the *contraction* of f by A and is well known to be submodular [20, p. 45]. Similarly, the contraction g_A is supermodular. It is easy to see that a search of S that begins with the elements of A is optimal only if it defines an optimal search of $\bar{A}$ with cost function f_A and weight function g_A. We summarize the observation below.

Lemma 3. *Suppose there is an optimal search of S with initial segment A. Then an optimal search of S can be found by combining an optimal search of A with respect to cost function $f|_A$ and weight function $g|_A$ with an optimal search of $\bar{A}$ with respect to cost function f_A and weight function g_A.*

We now repeat the process on $\bar{A}$ with cost function f_A and weight function g_A, finding a subset of maximum density, and so on. The result is a partition of S into subsets $A = A_1, A_2, \ldots, A_k$ such that there exists an optimal search strategy that respects the ordering of those subsets. This is a generalization of the notion of a Sidney [47] decomposition for optimal scheduling. If each subset A_j is chosen to be the maximal set of maximum density, then Theorem 1 implies that the resulting decomposition must be respected by any optimal search strategy.

We show that, in fact, *any* search that respects the ordering of such a decomposition $A_1, \ldots, A_k$ described above approximates an optimal search by a factor of 2, generalizing the analogous result for scheduling that can be found in Chekuri and Motwani [14] and Margot et al. [34]. We first show that if S itself has maximum density, then any search approximates an optimal search by a factor of 2.

Lemma 4. *Suppose that S has maximum search density. Then every search strategy has an expected cost in between $g(S)f(S)/2$ and $g(S)f(S)$.*

Proof. Let π be any search, and without loss of generality, suppose that $\pi(j) = j$ so that $S_j = S_j^\pi = \{1, \dots, j\}$. Write $x_j = g(S_j) - g(S_{j-1}), j = 1 \dots, n$, and note that $g(S_j) = \sum_{i \le j} x_i$. Then the expected cost of π is

$$\begin{aligned} c(\pi) &= \sum_j x_j f(S_j) \\ &\ge \frac{1}{\rho^*} \sum_j x_j g(S_j) \text{ (because } \rho(S_j) \le \rho^*) \\ &= \frac{1}{\rho^*} \left(\sum_j x_j^2 + \sum_{i<j} x_i x_j \right) \\ &= \frac{1}{2\rho^*} \left(\left(\sum_j x_j \right)^2 + \sum_j x_j^2 \right) \\ &= \frac{1}{2\rho^*} \left(g(S)^2 + \sum_j x_j^2 \right) \\ &\ge \frac{g(S)f(S)}{2}, \end{aligned}$$

because $g(S)/f(S) = \rho^*$. The cost of *any* search is at most $g(S)f(S)$. It follows that if S has maximum search density, then $g(S)f(S)/2 \le c(\pi) \le g(S)f(S)$. □

Our 2-approximation relies on being able to find a maximum density subset efficiently. The problem of maximizing the ratio of a supermodular function to a positive submodular function was considered in Iwata et al. [29, section 6], where it was shown that the problem can be solved in strongly polynomial time. For completeness, we present below a simple version of this algorithm that exploits the fact that f is nondecreasing:

1. Set $\lambda = \rho(S)$.
2. Maximize the supermodular function $g(X) - \lambda f(X)$ over subsets $X \subset S$. Let A be a maximizer.
3. If $\rho(A) = \rho(S) = \lambda$, return S as a maximum density subset.
4. Otherwise, set $S = A$ and go back to step 1.

Before we prove the correctness of this algorithm, first note that the total number of iterations is at most n, and each iteration involves a minimization of submodular functions, which can be performed in strongly polynomial time, using Schrijver's [44] algorithm or the Iwata–Fleischer–Fujishige algorithm [20].

To prove the algorithm does indeed return a maximum density subset, first note that if A maximizes $g(X) - \lambda f(X)$ and $\rho(A) = \rho(S) = \lambda$, then for any set $B \subset S$, we have $g(B) - \lambda f(B) \le g(A) - \lambda f(A) = 0$, so $\rho(B) \le \lambda = \rho(S)$, and S has maximum density.

So we just need to show that if A is a maximizer of $g(X) - \lambda f(X)$, then A contains a maximum density subset. Indeed, suppose that B has maximum density. Then, by the supermodularity of $g - \lambda f$ and the fact that A maximizes $g - \lambda f$, it follows that $g(B) - \lambda f(B) \le g(A \cap B) - \lambda f(A \cap B)$. This can be rewritten as

$$(\rho(B) - \lambda) f(B) \le (\rho(A \cap B) - \lambda) f(A \cap B).$$

Because B has maximum density and f is nondecreasing, it follows that $\rho(A \cap B) = \rho(B)$ and $f(A \cap B) = f(B)$, so $A \cap B$ is nonempty and has maximum density.

Theorem 2. *Suppose that the submodular function f and the supermodular function g are given by value oracles. Then there is a 2-approximation for an optimal search strategy to the submodular search problem that can be computed in strongly polynomial time.*

Proof. As discussed above, a subset $A \subset S$ of maximum density can be computed in strongly polynomial time. If A is the entire set, then any search is a 2-approximation by Lemma 4. If A is a proper subset, then there exists an optimal search with initial segment A.

Let π^* be an optimal search of S, let π_A be an optimal search of A with respect to functions $f|_A$ and $g|_A$, and let $\pi_{\overline{A}}$ be an optimal search of $\overline{A}$ with respect to functions f_A and g_A. Then

$$c_{f,g}(\pi^*) = c_{f|_A, g|_A}(\pi_A) + g_A(\overline{A}) f(A) + c_{f_A, g_A}(\pi_{\overline{A}}).$$

By induction, if we have a 2-approximation for π_A and $\pi_{\overline{A}}$, then we have one for π^*. □

The algorithm produces a partition $A_1, \ldots, A_k$ of S such that each A_i has maximum density in the complement of $\cup_{j<i} A_j$. The resulting search strategy π orders each A_i in an undetermined manner. The search strategy π is fully determined only if each A_i is a singleton. This happens only in very specific cases, for instance, if f and g are modular. The maximum density first algorithm then produces an optimal search strategy. As mentioned in the introduction, this corresponds to Smith's [49] rule for optimal scheduling or the result of Bellman [11] in the context of search theory.

We note that Pisaruk's [40, 41] algorithm also produces a Sidney decomposition of S. The important addition we have made here is Theorem 1, which implies that *every* optimal search follows a Sidney decomposition. Theorem 1 is also important in the next subsection, where we give a more refined expression for the approximation ratio of our algorithm.

2.2. Improved Approximation for Functions of Low Curvature

Define the *dual* $g^{\#}: 2^S \to \mathbb{R}$ of the set function g by $g^{\#}(A) = g(S) - g(\bar{A})$ (see [20, p. 36]). It is easy to see that $(g^{\#})^{\#} = g$. Also, g is nondecreasing and submodular with $g(\emptyset) = 0$ if and only if $g^{\#}$ is nondecreasing and supermodular with $g(\emptyset) = 0$.

Observe that for a search π, we have $c_{f,g}(\pi) = c_{g^{\#},f^{\#}}(\pi')$, where π' is the reverse of π. Indeed,

$$\begin{aligned} c_{f,g}(\pi) &= \sum_{j=1}^{n} f(S_j^{\pi})(g(S_j^{\pi}) - g(S_j^{\pi} - j)) \\ &= \sum_{j=1}^{n} (f^{\#}(S) - f^{\#}(\overline{S_j^{\pi}}))(g^{\#}(\overline{S_j^{\pi} - j}) - g^{\#}(\overline{S_j^{\pi}})) \\ &= \sum_{j=1}^{n} g^{\#}(S_j^{\pi'})(f^{\#}(S_j^{\pi'}) - f^{\#}(S_j^{\pi'} - j)) \\ &= c_{f^{\#},g^{\#}}(\pi'). \end{aligned}$$

It follows that $\min_{\pi} c_{f,g}(\pi) = \min_{\pi} c_{g^{\#},f^{\#}}(\pi)$, and we will use this *duality* later.

We now show that the algorithm of Section 2.1 performs better when the cost function f and the dual function $g^{\#}$ have *total curvature* less than 1. The *total curvature* κ of a set function f on S such that $f(\emptyset) = 0$ and $f(s) > 0$ for all $s \in S$ is

$$\kappa = 1 - \min_{s \in S} \frac{f_{S-s}(s)}{f(s)} = \max_{s \in S} \frac{f(s) + f(S - s) - f(S)}{f(s)}.$$

This was first defined in Conforti and Cornuéjols [14]; see also Vondrak [54]. When f is monotone nondecreasing, $\kappa \le 1$. When it is submodular, $\kappa \ge 0$ (with equality if and only if f is modular), and the value $f_X(s)$ decreases as $X \subset S$ increases, but it always exceeds $(1 - \kappa)f(s)$. Note that if $\kappa_{g^{\#}}$ is the total curvature of $g^{\#}$ for a supermodular function g with $g(\emptyset) = 0$, then $\kappa_{g^{\#}} = (g(S) - g(s) - g(S - s))/(g(S) - g(S - s))$.

Lemma 5. *Suppose that S has maximum search density, f has total curvature κ_f, and $g^{\#}$ has total curvature κ_g. Then, for all $A \subset S$, the density $\rho(A)$ satisfies*

$$(1 - \kappa_f)(1 - \kappa_{g^{\#}})\rho^* \le \rho(A) \le \rho^*.$$

Proof. The second inequality follows from the fact that S has maximum density ρ^*. To prove the first inequality, first observe that

$$\frac{f(S) - f(\bar{A})}{f(A)} \ge \frac{\sum_{s \in A} f_{S-s}(s)}{\sum_{s \in A} f(s)} \ge \frac{\sum_{s \in A}(1 - \kappa_f)f(s)}{\sum_{s \in A} f(s)} = 1 - \kappa_f.$$

Similarly, $g^{\#}(S) - g^{\#}(\bar{A}) \ge (1 - \kappa_{g^{\#}})g^{\#}(A)$ or, equivalently, $g(A) \ge (1 - \kappa_{g^{\#}})(g(S) - g(\bar{A}))$. Hence,

$$(1 - \kappa_f)(1 - \kappa_{g^{\#}})\rho^* f(A) \le (1 - \kappa_{g^{\#}})\rho^*(f(S) - f(\bar{A})) \le (1 - \kappa_{g^{\#}})(g(S) - g(\bar{A})) \le g(A),$$

where the second inequality comes from the fact that S has maximum density. The first inequality of the lemma follows. □

We can now revisit the proof of Lemma 4 by deriving a tighter upper bound on the expected cost of any search strategy π and a tighter lower bound on an optimal search strategy π^* when S has maximum density. This is based on Edmonds's [18] well-known greedy algorithm (see, e.g., [20, section 3.2]). The submodular base polyhedron is defined as

$$\mathbb{B}(f) = \{\mathbf{x} \in \mathbb{R}^S : \mathbf{x}(A) \le f(A) \text{ for all } A \subset S, \mathbf{x}(S) = f(S)\}. \tag{4}$$

Lemma 6 (Edmonds [18]). *For a submodular function f, an optimal solution to $\max \mathbf{w}^T\mathbf{x}$ subject to $\mathbf{x} \in \mathbb{B}(f)$ is given by*

$$x_j = f(S_j^\pi) - f(S_j^\pi - j), \quad \forall j = 1, \dots, n,$$

where π is a permutation that orders S in nonincreasing order of w_j.

Lemma 7. *Suppose that $f : 2^S \to \mathbb{R}^+$ is submodular and $g \to \mathbb{R}^+$ is supermodular, and let κ_f and $\kappa_{g^\#}$ be the total curvatures of f and $g^\#$, respectively. Define a function $\varepsilon = \varepsilon_{f,g}$ on permutations π of S by*

$$\varepsilon(\pi) = \sum_{j=1}^n (f(S_j^\pi) - f(S_j^\pi - j))(g(S_j^\pi) - g(S_j^\pi - j)) = \sum_{j=1}^n d_j f(S_j^\pi) d_j g(S_j^\pi).$$

Let π_1 be a permutation that orders the elements in nonincreasing order of $f(j)$, and let π_2 be a permutation of S that orders the elements in nonincreasing order of $g^\#(j)$. Then
(i) $(1 - \kappa_f)\varepsilon(\pi_1) \le \min_\pi \varepsilon(\pi)$ *and*
(ii) $(1 - \kappa_{g^\#})\varepsilon(\pi_2) \le \min_\pi \varepsilon(\pi)$.

Proof. For part (i), let us fix the cost function $w_j = f(j)$ for $j = 1, \dots, n$. Then Lemma 6 implies that π_1 minimizes the function

$$\varepsilon'(\pi) = \sum_{j=1}^n w_j (g(S_j^\pi) - g(S_j^\pi - j)).$$

It follows that for any permutation π,

$$(1 - \kappa_f)\varepsilon(\pi_1) \le (1 - \kappa_f)\varepsilon'(\pi_1) \le (1 - \kappa_f)\varepsilon'(\pi) \le \varepsilon(\pi).$$

The third inequality follows from the definition of κ_f.

Part (ii) follows using the similar argument or by observing that $\varepsilon_{f,g}(\pi) = \varepsilon_{g^\#,f^\#}(\pi')$, where π' is the reverse permutation of π (so that $\pi'(i) = \pi(n + 1 - i)$). Indeed,

$$\begin{aligned}
\varepsilon_{f,g}(\pi) &= \sum_{j=1}^n (f(S_j^\pi) - f(S_j^\pi - j))(g(S_j^\pi) - g(S_j^\pi - j)) \\
&= \sum_{j=1}^n (f^\#(\overline{S_j^\pi - j}) - f^\#(\overline{S_j^\pi}))(g^\#(\overline{S_j^\pi - j}) - g^\#(\overline{S_j^\pi})) \\
&= \sum_{j=1}^n (f^\#(S_j^{\pi'}) - f^\#(S_j^{\pi'} - j))(g^\#(S_j^{\pi'}) - g^\#(S_j^{\pi'} - j)) \\
&= \varepsilon_{f^\#,g^\#}(\pi'). \quad \square
\end{aligned}$$

Theorem 3. *Suppose that the submodular function f and the supermodular function g are given by a value oracle, f has total curvature $\kappa_f < 1$, and $g^\#$ has total curvature $\kappa_{g^\#} < 1$. Then there is a search strategy that can be computed in strongly polynomial time and approximates an optimal search strategy for the submodular search problem with approximation ratio $\frac{2}{1+\delta}$, where*

$$\delta = \min\left\{\theta, \frac{2\theta \max\{1 - \kappa_f, 1 - \kappa_{g^\#}\}}{1 + \theta}\right\},$$

and $\theta = (1 - \kappa_f)(1 - \kappa_{g^\#})$. If either f or g is modular, then the approximation ratio is $\frac{2}{1+\theta}$.

Proof. First suppose that S has maximum density. We normalize f and g so that $f(S) = g(S) = 1$; thus, $\rho^* = \rho(S) = 1$.

Recall that by duality, $\min_\pi c_{f,g}(\pi) = \min_\pi c_{g^\#,f^\#}(\pi)$. Note that S has maximum density with respect to f and g if and only if it has maximum density with respect to $g^\#$ and $f^\#$.

Hence, by Lemma 5, for any $A \subset S$,

$$\theta \le \frac{g(A)}{f(A)} \le 1 \text{ and } \theta \le \frac{f^\#(\bar{A})}{g^\#(\bar{A})} \le 1.$$

This means, in particular, that

$$f(A) \le \min\left\{\frac{g(A)}{\theta}, 1 - \theta + \theta g(A)\right\}. \tag{5}$$

For any search π, we can write

$$\begin{aligned} c(\pi) &= \sum_{j=1}^{n} (g(S_j) - g(S_j - j)) f(S_j) \\ &= \frac{1}{2}\varepsilon(\pi) + \sum_{j=1}^{n} \frac{1}{2}(g(S_j) - g(S_{j-1}))(f(S_j) + f(S_{j-1})). \end{aligned} \tag{6}$$

The sum in (6) is the area under the piecewise linear curve in $\mathbb{R}^2$ connecting the points $(g(S_j), f(S_j)), j = 0, 1, \ldots, n$. By (5), this is at most the area under the curve $y = \min\{x/\theta, 1 - \theta + \theta x\}, x \in [0,1]$, which can be easily calculated to be $1/(1+\theta)$.

Because the expected cost is always bounded above by 1, it follows that

$$c(\pi) \le \min\left\{\frac{1}{1+\theta} + \frac{1}{2}\varepsilon(\pi), 1\right\}.$$

Now consider an optimal search π^*. For this search, the sum in (6) is at least 1/2 because $f(A) \ge g(A)$ for any $A \subset S$. By Lemma 7, we can choose π to be some search such that $\varepsilon(\pi^*) \ge \max\{1 - \kappa_f, 1 - \kappa_{g^\#}\}\varepsilon(\pi)$. So (6) implies that

$$c(\pi^*) \ge \frac{1}{2} + \frac{1}{2}\max\{1 - \kappa_f, 1 - \kappa_{g^\#}\}\varepsilon(\pi).$$

Hence,

$$\frac{c(\pi)}{c(\pi^*)} \le \frac{\min\left\{\frac{1}{1+\theta} + \frac{1}{2}\varepsilon(\pi), 1\right\}}{\frac{1}{2} + \frac{1}{2}\max\{1 - \kappa_f, 1 - \kappa_{g^\#}\}\varepsilon(\pi)}.$$

This is maximized either at $\varepsilon(\pi) = \frac{2\theta}{1+\theta}$ or $\varepsilon = 0$, giving the first bound in the statement of the theorem.

If either f or g is modular, then $\delta = \min\left\{\theta, \frac{2\theta}{1+\theta}\right\} = \theta$.

If S does not have maximum density, then an induction argument similar to that of Theorem 2 completes the proof. □

We note that we would be able to improve the approximation ratio in Theorem 3 to $\frac{2}{1+\theta}$ for arbitrary submodular f and supermodular g if we could find an exact solution to the problem of minimizing $\varepsilon(\pi)$ of Lemma 7, and we leave this as an open problem.

2.3. An Optimal Search for Series-Parallel Decomposable Problems

In this section, we show how Theorem 1 may be used to determine an optimal search for problems we call *series-parallel decomposable*. The idea for series-parallel decomposability is motivated by the following example of expanding search on a tree, considered in Alpern and Lidbetter [4]. Let S be the vertex set of a tree $T = (S, E)$ with edge set E, and each $e \in E$ has weight $w(e)$. Let $r \in S$ be the root of the tree and restrict attention to searches that begin at r. For a set of edges A, define $f(A)$ to be the sum of the edge weights in the tree that is spanned by $\{r\} \cup A$. It is clear that if r has degree 1, then every search begins with the edge incident to r. We generalize this principle by defining *f-initial sets* below. If r has degree greater than 1, then T is the union of two edge-disjoint subtrees with root r, and it is easy to show that there is a maximal density subset of S whose elements are the vertices of one of these subtrees. So the problem of finding an optimal search can be decomposed. We generalize this principle using the concept of separators. We say that a proper nonempty

subset $B \subset S$ is a *separator* of f if f is the direct sum of $f|_B$ and $f|_{\overline{B}}$. To check whether B is an f-separator, we need to verify only that $f(S) = f(B) + f(\overline{B})$ (see [16, proposition 5]).

2.3.1. The f-Initial Sets. For a set $A \subset S$, we define the *f-closure* $cl(A)$ of A as the maximal set B containing A such that $f(B) = f(A)$; there is a unique such set. We say that a proper subset $I \subset S$ is an *f-initial set* if $I \subset cl(s)$ for every $s \in \overline{I}$. In the case that f corresponds to the special case of precedence-constrained scheduling, f-initial sets and f-closures correspond to the usual notions of initial sets and closures with respect to the precedence constraints. We leave it to the reader to check that a set I is an f-initial set if and only if for any subset $A \subset S$ that contains some element of $\overline{I}$ we have $f(A \cup I) = f(A)$.

Note that if there is an f-initial set, then the total curvature of f is 1 (i.e., the worst possible). Therefore, our approximation given in Theorem 3 is not helpful. However, it is easy to show that there is an optimal search with initial segment I.

Lemma 8. *If I is an f-initial set or $\overline{I}$ is a $g^{\#}$-initial set, then there exists an optimal search with initial segment I.*

Proof. First, suppose that I is an f-initial set and that there are no optimal searches with initial segment I. Let σ be an optimal search that has been chosen to minimize $\sum_{s\in I} \sigma^{-1}(s)$. Because I is not an initial segment of σ, there must be some $t \notin I$ that directly precedes some $s \in I$. Let A be the set of all elements preceding t, and let τ be the search obtained by switching the order of s and t.

Then the difference in expected costs between σ and τ is

$$\begin{aligned} c(\sigma) - c(\tau) &= f(A \cup \{t\}) d_t g(A \cup \{t\}) + f(A \cup \{s,t\}) d_s g(A \cup \{s,t\}) \\ &\quad - f(A \cup \{s\}) d_s g(A \cup \{s\}) - f(A \cup \{s,t\}) d_t g(A \cup \{s,t\}) \\ &\geq f(A \cup \{s,t\})(d_t g(A \cup \{t\}) + d_s g(A \cup \{s,t\}) - d_t g(A \cup \{s,t\})) - f(A \cup \{s\}) d_s g(A \cup \{s\}) \\ &= d_t f(A \cup \{s,t\}) d_s g(A \cup \{s\}) \\ &\geq 0, \end{aligned}$$

where the first inequality comes from the fact that $f(A \cup \{t\}) = f(A \cup \{t\} \cup I) \geq f(A \cup \{s,t\})$ because I is an initial set and by monotonicity. Hence, τ is an optimal search with $\sum_{s\in I} \tau^{-1}(s) < \sum_{s\in I} \sigma^{-1}(s)$, contradicting the definition of σ. So there must be an optimal search with initial segment I.

If $\overline{I}$ is a $g^{\#}$-initial set, the fact that there is an optimal search beginning with initial segment I follows immediately from duality. □

Therefore, if an f-initial set I exists, then it follows from Lemma 3 that to find an optimal search of S, it is sufficient to find an optimal search of I with respect to $f|_I$ and $g|_I$ and an optimal search of $\overline{I}$ with respect to $f_{\overline{I}}$ and $g_{\overline{I}}$. The process is similar if $\overline{I}$ is $g^{\#}$-initial. This is one way in which the problem can be decomposed.

Finding an f-initial set can be performed in polynomial time, as we now explain. For any $s \in S$, let I_s be the largest f-initial set not containing s. (If no such f-initial set exists, let $I_s = \emptyset$.) If we find a nonempty I_s for any $s \in S$, we can return it as an f-initial set. In case $I_s = \emptyset$ for every $s \in S$, we conclude that there is no f-initial set.

To find I_s, we maintain a candidate T, starting with $T = cl(s) - \{s\}$. We know that $I_s \subset T$ and that for every $t \in S - T$ we have $I_s \subset cl(t)$. Hence, we take an arbitrary $t \in S - T$ and update T as $T \cap cl(t)$.

We iterate this process: While there exists a $t \in S - T$ that we have not yet examined, we update T as $T \cap cl(t)$. We examine every t at most once. At termination, if $T \neq \emptyset$, then we must have $T \subset cl(t)$ for every $t \in S - T$, showing that T is an f-initial set. It is also clear from the construction that $T = I_s$, the largest f-initial set disjoint from s.

2.3.2. Separators. The other way that the problem can be decomposed is by finding a separator, as we now explain. For a lattice $\mathcal{L}$ and $B \subset S$, the restriction to B is $\mathcal{L}|_B = \{A \in \mathcal{L} : A \subset B\}$. If S and T are disjoint subsets, and if $\mathcal{L}$ is a lattice in S and $\mathcal{N}$ is a lattice in T, then the direct sum of these lattices is $\{A \cup A' : A \in \mathcal{L},\ A' \in \mathcal{N}\}$. It is a lattice in $S \cup T$.

Lemma 9. *If B is a separator of both f and g, then $\mathcal{M}$ is the direct sum of $\mathcal{M}|_B$ and $\mathcal{M}|_{\overline{B}}$.*

Proof. If A has maximum density, then the inequality

$$\rho^* = \rho(A) = \frac{g(A \cap B) + g(A \cap \overline{B})}{f(A \cap B) + f(A \cap \overline{B})} \leq \max\{\rho(A \cap B), \rho(A \cap \overline{B})\}$$

is in fact an equality. It follows that if $A \cap B \neq \emptyset$, then $\rho(A \cap B) = \rho^*$, and if $A \cap \overline{B} \neq \emptyset$, then $\rho(A \cap \overline{B}) = \rho^*$. □

It follows from Lemma 9 that if B is a separator, then either B or $\overline{B}$ (or both) must contain a subset A of density ρ^*. So, by Theorem 1, there exists an optimal search of S with initial segment A where A is a proper subset of S. In that case, we can again apply Lemma 3 to decompose the problem of finding an optimal search into two subproblems on A and $\overline{A}$.

If there exists a separator of both f and g, then it is possible to find one in strongly polynomial time, using the following method. The *connectivity function* of f is defined as $d_f(B) = f(B) + f(\overline{B}) - f(S)$. It is a symmetric nonnegative submodular function [16], as is the connectivity function of $h = f - g$. We say that a nonempty subset $B \subset S$ is a *split* if $d_h(B)$ is minimal and $d_h(A) > d_h(B)$ for all nonempty $A \subset B$. Obviously, a split is a separator of both f and g if and only if $d_h(B) = 0$, and because a split can be computed from a submodular function minimization, this can be carried out in strongly polynomial time by Queyranne's [42] algorithm for minimizing symmetric submodular functions.

We can now define series-parallel decomposability, which is an extension of an idea from theorem 3 of Fokkink et al. [19].

Definition 2. We say the submodular search problem is *series-parallel decomposable* if
(i) (Series Decomposable) there exists some set I such that I is f-initial or $\overline{I}$ is $g^{\#}$-initial; or
(ii) (Parallel Decomposable) there exists some set B that is a separator of both f and g.

We say that f is *series-parallel decomposable* if it can be repeatedly decomposed until all remaining search segments are singletons.

We have shown that if the submodular search problem is series-parallel decomposible, then by decomposing it we can determine an optimal search strategy. We summarize this result with a theorem.

Theorem 4. *Suppose that a submodular function f and the supermodular function g are given by a value oracle. Then we can decide in strongly polynomial time whether the submodular search problem is series-parallel decomposable, and if so, we can find an optimal search.*

As pointed out at the end of Section 2.1, the submodular search problem can be solved if both f and g are modular. In this case, the problem is series-parallel decomposable (by repeated parallel decompositions). The example from Alpern and Lidbetter [4] described at the beginning of this subsection is series-parallel decomposable as well. Theorem 4 also has an interpretation in scheduling with regard to scheduling jobs whose precedence constraints are given by a series-parallel graph. This is explained in more detail in Section 2.4.

We could extend the concept of series-parallel decomposition to a more general notion of *logarithmic decomposition*, meaning that the problem can be repeatedly decomposed until all remaining search segments A have cardinality $|A| \leq p \log n$ for some (small) constant p. Then each search subproblem in such a subset A can be solved by brute force (in $|A|!$ time) or by dynamic programming (in $|A|2^{|A|}$ time; see [27]), resulting in an overall time of $O(n^{p+2})$.

2.4. Applications to Scheduling

As we outlined in Section 1.3, the submodular search problem has a natural application to single-machine scheduling. Theorem 4 generalizes the well-known result that the problem $1 \mid \text{prec} \mid \sum w_j C_j$ can be solved in polynomial time if the Hasse diagram defined by the precedence constraints on the jobs is a generalized series-parallel graph [1, 31]. We define *generalized series parallel* here for completeness.

Denote a Hasse diagram by a pair $\{N, E\}$ of nodes N and directed edges $E \subset N^2$. For disjoint vertex sets N_1 and N_2, we let $N_1 \times N_2 = \{(u, v) : u \in N_1, v \in N_2\}$ denote the complete directed bipartite graph from N_1 to N_2. Then we have the following:

1. If $G_1 = \{N_1, E_1\}$ and $G_2 = \{N_2, E_2\}$ are graphs on disjoint vertex sets, then $\{N_1 \cup N_2, E_1 \cup E_2\}$ is the *parallel composition* of G_1 and G_2.

2. If $G_1 = \{N_1, E_1\}$ and $G_2 = \{N_2, E_2\}$ are graphs on disjoint vertex sets, then $\{N_1 \cup N_2, E_1 \cup E_2 \cup (N_1 \times N_2)\}$ is the *series composition* of G_1 and G_2.

The graph $\{\{i\}, \emptyset\}$ containing a single node is generalized series parallel, and any graph that can be obtained by a finite number of applications of parallel composition or series composition of generalized series-parallel graphs is generalized series parallel.

Recall that in the framework of the submodular search problem, the problem $1|\text{prec}|\sum w_j C_j$ has cost function f such that $f(A)$ is the sum of the processing times of the jobs in the precedence closure of a set A of jobs. If the Hasse diagram corresponding to some precedence constraints is a parallel composition, then clearly the corresponding submodular cost function f has a separator. If the Hasse diagram is a series composition, then f has an

f-initial set. Thus, the concept of a series-parallel decomposable submodular search problem generalizes the problem $1|\text{prec}|\sum w_jC_j$ when the precedence constraints are given by a generalized series-parallel graph.

It is also possible that the concept of series-parallel decomposability could be used to generalize work in the machine scheduling literature that extends the solution of $1|\text{prec}|\sum w_jC_j$ in the generalized series-parallel case, for example, Sidney and Steiner [48] and Monma and Sidney [39]. However, this work does not correspond directly with Theorem 4, and we leave for future work the question of how to link these extensions to our submodular framework.

Theorem 2 can also be applied to the problem $1 \mid \text{prec} \mid \sum w_A h(C_A)$, in which the object is to minimize the weighted sum of some concave function h of the completion times of *subsets* of jobs, where h is given by a value oracle. This problem also has a natural interpretation in terms of searching for multiple hidden objects, where w_A is the probability that there are objects hidden in the locations contained in A. We summarize this in the following theorem.

Theorem 5. *Suppose that the nondecreasing concave real function h and the function $g : A \mapsto \sum_{B\subset A} w_B$ are given by a value oracle, where the weights w_A are nonnegative. Then there is an algorithm running in strongly polynomial time in n that computes a 2-approximation for $1 \mid \text{prec} \mid \sum w_A h(C_A)$.*

Proof. This is an immediate consequence of Theorem 2 because the composition of the concave function h with the submodular function $f : A \mapsto p(\widetilde{A})$ is itself submodular, and the function $g : A \mapsto \sum_{B\subset A} w_B$ is supermodular. □

It is worth noting that because h is applied to costs in this model, a concave h corresponds to larger losses having a decreasing marginal disutility. This reflects a risk-seeking attitude, a less common assumption than risk aversion.

We may also consider the more specific problem $1\|\sum w_j h(C_j)$ for some nondecreasing function h. In Megow and Verschae [37], a polynomial time approximation scheme is given for the problem. If $h(C_j) = C_j^\beta$ for $\beta \neq 1$, then it is the problem considered in Bansal et al. [10]. It is unknown whether there exists a polynomial time algorithm to compute an optimal schedule for this problem or if it is NP-hard (see Bansal et al. [10] and the references therein). For a concave or convex h, it is shown in Stiller and Wiese [50] that Smith's rule (for the original problem $1\|\sum_j w_jC_j$) yields a $(\sqrt{3}+1)/2 \approx 1.37$-approximation to the problem $1\|\sum_j w_j h(C_j)$, whereas Höhn and Jacobs [28] give explicit formulas for the exact approximation ratio of Smith's rule for this problem in terms of a maximization over two continuous variables.

Of course, our algorithm gives rise to a Sidney decomposition for $1\|\sum_j w_j h(C_j)$ that is not necessarily consistent with the application of Smith's rule for the problem $1\|\sum_j w_jC_j$. Furthermore, Theorem 1 implies that every optimal schedule must follow a Sidney decomposition of our type. If h defines a submodular cost function f with low total curvature, we can use Theorem 3 to express the approximation ratio of our algorithm in a simple form that may be better than the 1.37-approximation in Stiller and Wiese [50]. For example, if $\kappa_f = 1/2$ and $\kappa_{g^\#} = 0$, then our approximation ratio is $2/(1+1/2) = 1.33$. Note that the curvature κ_f of f is given by

$$1-\kappa_f = \min_A \frac{h(p(S))-h(p(A))}{h(p(S)-p(A))} \geq \inf_{y\in[0,p(S)]} \frac{h(p(S))-h(y)}{h(p(S)-y)},$$

where $p : 2^S \to \mathbb{R}$ denotes processing time. Because the fraction on the right is the ratio of a decreasing concave function to a decreasing convex function, its infinum is achieved in the limit as $y \to p(S)$. Because $g^\#$ is modular in this problem, applying Theorem 3, we obtain the following.

Theorem 6. *Suppose that $h : \mathbb{R}^+ \to \mathbb{R}^+$ is a nondecreasing concave real function. Then there is a schedule for $1\|\sum w_j h(C_j)$ that can be computed in strongly polynomial time and approximates an optimal schedule with approximation ratio $\frac{2}{2-\kappa_f}$. The parameter κ_f is given by*

$$1-\kappa_f = \lim_{y\to p(S)} \frac{h(p(S))-h(y)}{h(p(S)-y)} = \frac{h'(p(S))}{h'(0)},$$

where the second equality holds by l'Hôpital's rule if h is differentiable at 0 and $p(S)$.

By way of an example, we scale so that $p(S) = 1$ and take the standard log utility function [52]: $h(y) = \log(1+ay)$, for some positive constant a. Then Theorem 6 implies that we can find a schedule that approximates an optimal schedule for $1\|\sum w_j h(C_j)$ with approximation ratio $1 + a/(2+a)$.

Another classic example is the function $h(y) = (1-e^{-ry})/r$ (with discount rate $r > 0$) for continuously discounted search (or wait) time, as in [43]. For this choice of h, our approximation ratio for $1\|\sum w_j h(C_j)$ is $2/(1+e^{-r})$.

We conclude this section by arguing that the submodular search problem really is more general than $1 \mid \text{prec} \mid \sum w_A h(C_A)$. Indeed, consider the problem with $S = \{1,2,3\}$ and $f(1) = f(2) = f(3) = 1$, $f(1,2) = f(1,3) = 2$, $f(2,3) = 3/2$, and $f(1,2,3) = 2$. Suppose that f is defined by some partial order on S, some processing times p_j, and some concave function h of completion times, as in the proof of Theorem 5. Then the partial order on the jobs must be an antichain (i.e., no jobs are comparable) because otherwise we would have $f(A) = f(B)$ for some $1 = |A| \subsetneq |B|$. It must also be the case that $p_1 = p_2 = p_3$ because, if not, by the concavity of h and because $f(1) = f(2) = f(3) = 1$ it would have to be the case that $f(A) = 1$ for all $A \neq \emptyset$. But then we would have $2 = f(1,2) = h(p_1 + p_2) = h(p_2 + p_3) = f(2,3) = 3/2$, a contradiction.

3. The Submodular Search Game

We now turn to the submodular search game, and we seek optimal mixed strategies for the players, settling a question from Fokkink et al. [19]. Here each hider's mixed strategy (probability distribution of S) $\mathbf{x}$ defines a modular function g, where $g(A) = \mathbf{x}(A)$ for all $A \subset S$. We use $c_{f,\mathbf{x}}$ to denote the search cost for such a g. A mixed strategy for the searcher is some $\mathbf{p}$ that assigns a probability $\mathbf{p}(\pi)$ to each pure strategy π. We denote by $C_f(\mathbf{p}, \mathbf{x})$ the expected search cost for mixed strategies $\mathbf{p}$ and $\mathbf{x}$, where $\mathbf{p}$ and $\mathbf{x}$ are independent; that is,

$$C_f(\mathbf{p}, \mathbf{x}) = \sum_{\pi} \mathbf{p}(\pi) c_{f,\mathbf{x}}(\pi).$$

We suppress the subscript f when the context is clear.

Recall the definition of the *submodular base polyhedron* $\mathbb{B}(f)$ in (4). We apply Theorem 1 to settle a question from Fokkink et al. [19].

Theorem 7. *Every equilibrium strategy for the hider in the submodular search game is in the scaled base polyhedron* $\frac{1}{f(S)}\mathbb{B}(f)$.

Proof. By contradiction. Suppose that $\mathbf{x}$ is an equilibrium hider strategy, but it is not in the base polyhedron. Then there exists a subset A such that $\mathbf{x}(A) > f(A)/f(S)$ so that $\rho(A) > 1/f(S) = \rho(S)$. It follows that the largest subset M of maximum density is a proper subset of S. Any pure strategy best response π to $\mathbf{x}$ searches M first, by Theorem 1, and hence an optimal mixed strategy of the searcher assigns positive probability only to orderings of S with initial segment M. Now, informally, an optimal response to these searcher strategies is to hide in $\overline{M}$, which cannot be an equilibrium strategy.

More formally, we observe that every pure search strategy in the support of an equilibrium strategy $\mathbf{p}$ is a best response to $\mathbf{x}$, so it must start by searching the whole of M. Let $k \in \overline{M}$. Then we must have $f(M \cup \{k\}) > f(M)$; otherwise, M cannot be maximal. Define $\mathbf{y}$ by $\mathbf{y}(M) = 0$, $\mathbf{y}(k) = \mathbf{x}(k) + \mathbf{x}(M)$ and $\mathbf{y}(j) = \mathbf{x}(j)$ for any $j \notin M \cup \{k\}$. Then we have

$$C(\mathbf{p}, \mathbf{y}) - C(\mathbf{p}, \mathbf{x}) \geq \mathbf{x}(M)(f(M \cup \{k\}) - f(M)) > 0,$$

so $\mathbf{x}$ cannot be a best response to $\mathbf{p}$: a contradiction. Hence, we must have $M = S$, and $\mathbf{x}$ is in the base polyhedron of $\frac{1}{f(S)} f$. □

Combining Theorem 7 with Lemma 4, the value of the submodular search game must lie between $f(S)/2$ and $f(S)$. Also, any hider strategy $\mathbf{x}$ in the base polyhedron of $\frac{1}{f(S)} f$ is a 2-*approximation for the hider's equilibrium strategy* in the sense that $\min_{\mathbf{p}} C(\mathbf{p}, \mathbf{x}) \geq V/2$, where V is the value of the game. Furthermore, *any* searcher strategy $\mathbf{p}$ is a 2-*approximation for the searcher's equilibrium strategy* in the sense that $\max_{\mathbf{x}} C(\mathbf{p}, \mathbf{x}) \leq 2V$.

We can also find strategies that are better approximations for the equilibrium strategies for cost functions with total curvature less than $1/2$. Define the modular function $h(A) = \sum_{s \in A} f(s)$. Then $f(A) \leq h(A)$ and $f(A) \geq (1 - \kappa)h(A)$ for all $A \subset S$. It follows that for any mixed strategies $\mathbf{p}$ and $\mathbf{x}$, we have $C_h(\mathbf{p}, \mathbf{x}) \geq C_f(\mathbf{p}, \mathbf{x}) \geq (1 - \kappa)C_h(\mathbf{p}, \mathbf{x})$.

Two different solutions to the search game with modular cost function h can be found in Alpern and Lidbetter [4] and Lidbetter [33]. The equilibrium strategy for the hider (shown to be unique in [4]) is given by $\mathbf{x}^h(s) = h(s)/h(S) = f(s)/h(S)$. The equilibrium strategy $\mathbf{p}^h$ for the searcher given in [33] is to begin with an element s with probability $\mathbf{x}^h(s)$ and to search the remaining elements in a uniformly random order. Note that this not an extreme point solution to the linear program defining an equilibrium searcher strategy. The equilibrium strategy for the searcher given in [4] is less concise to describe and is iteratively constructed, much like the searcher strategy of Theorem 8, which generalizes it.

Proposition 1. *Suppose that f has total curvature $\kappa < 1/2$. Then the equilibrium strategies $\mathbf{x}^h$ and $\mathbf{p}^h$ in the submodular search game with cost function h are $\left(\frac{1}{1-\kappa}\right)$-approximations for the equilibrium strategies in the submodular search game with cost function f.*

Proof. Let V_f and V_h be the values of the game with cost function f and the game with cost function h, respectively. Then

$$V_h = \max_{\mathbf{x}} C_h(\mathbf{p}^h, \mathbf{x}) \geq \max_{\mathbf{x}} C_f(\mathbf{p}^h, \mathbf{x}) \geq V_f \geq \min_{\mathbf{p}} C_f(\mathbf{p}, \mathbf{x}^h) \geq (1-\kappa) \min_{\mathbf{p}} C_h(\mathbf{p}, \mathbf{x}^h) = (1-\kappa)V_h.$$

It follows that $\max_{\mathbf{x}} C_f(\mathbf{p}^h, \mathbf{x}) \leq V_h \leq V_f/(1-\kappa)$ and $\min_{\mathbf{p}} C_f(\mathbf{p}, \mathbf{x}^h) \geq (1-\kappa)V_h \geq (1-\kappa)V_f$. □

We now define a notion of series-parallel decomposition for the submodular search game, similar to Definition 2 for the submodular search problem. We say the game is *series-parallel decomposable* if there is an f-initial set or if there is a separator of f. Note that this is equivalent to saying that the problem of finding a best response to a given hider strategy $\mathbf{x}$ is series-parallel decomposable.

In the case where the game is series-parallel decomposable, we can improve on Proposition 1.

Theorem 8. *Suppose that the submodular search game is series-parallel decomposable. Then, if f is given by a value oracle, an equilibrium strategy $\mathbf{x}^f$ for the hider can be computed in strongly polynomial time. An equilibrium searcher strategy can also be computed in strongly polynomial time. The value V of the game is*

$$V = \frac{1}{2}(f(S) + \Phi), \tag{7}$$

where $\Phi = \sum_{s \in S} \mathbf{x}^f(s) f(s)$.

Proof. The theorem is proved by induction on the number of hiding locations $n = |S|$. We write $V = V^f$ and $\Phi = \Phi^f$ to indicate the dependence on f. We will define both the hider's equilibrium strategy $\mathbf{x}^f$ and an equilibrium strategy $\mathbf{p}^f$ for the searcher recursively.

The base case, $n = 1$, is immediate because for $S = \{s\}$, the players each have only one available strategy: $\mathbf{x}^f(s) = 1$ for the hider and $\mathbf{p}^f(\pi)$ for the searcher, where π is the unique permutation of S. Then $\Phi^f = f(s) = f(S)$ and $f(S) = V^f = \frac{1}{2}(f(S) + \Phi^f)$.

For the induction step, there are two cases. The first case is that there is an f-initial set I. In this case, we claim that $V^f = f(I) + V^{f_{\bar{I}}}$. Indeed, the searcher can ensure that $V^f \leq f(I) + V^{f_{\bar{I}}}$ by using the strategy $\mathbf{p}^f$, which searches I in any order and then searches $\bar{I}$ according to the mixed strategy $\mathbf{p}^{f|_{\bar{I}}}$. By Lemma 8, the hider can ensure that $V^f \geq f(I) + V^{f_{\bar{I}}}$ by using the strategy $\mathbf{x}^f$ given by $\mathbf{x}^f(s) = \mathbf{x}^{f_{\bar{I}}}(s)$ for $s \in \bar{I}$ and $\mathbf{x}^f(s) = 0$ for $s \in I$. Hence, by induction, the strategies $\mathbf{x}^f$ and $\mathbf{p}^f$ are equilibrium strategies and can be calculated in strongly polynomial time. Furthermore,

$$\begin{aligned} V^f &= f(I) + \frac{1}{2}\left(f_{\bar{I}}(\bar{I}) + \Phi_{f_{\bar{I}}}(\bar{I})\right) \\ &= f(I) + \frac{1}{2}\left((f(S) - f(I)) + (\Phi_f(S) - f(I))\right) \\ &= \frac{1}{2}(f(S) + \Phi). \end{aligned}$$

The second case is that f has a separator A. Then we define $\mathbf{x}^f$ on A by $\mathbf{x}^f(s) = \frac{f(A)}{f(S)} \mathbf{x}^{f|_A}(s)$ and on $\bar{A}$ by $\mathbf{x}^f(s) = \frac{f(\bar{A})}{f(S)} \mathbf{x}^{f|_{\bar{A}}}(s)$. By Lemma 9, there must be a set of maximum density contained in A, and by induction and Theorem 7, A has maximum density. So, by Theorem 1, there is a best response π to $\mathbf{x}^f$ that starts with A. By induction, we must have

$$\begin{aligned} C(\pi, \mathbf{x}^f) &\geq \mathbf{x}^f(A) V_{f|_A}(A) + \mathbf{x}^f(\bar{A})(f(A) + V_{f|_{\bar{A}}}(\bar{A})) \\ &= \frac{f(A)}{f(S)} \cdot \frac{1}{2}(f|_A(A) + \Phi_{f|_A}(A)) + \frac{f(\bar{A})}{f(S)} \cdot \left(f(A) + \frac{1}{2}(f|_{\bar{A}}(\bar{A}) + \Phi_{f|_{\bar{A}}}(\bar{A}))\right) \\ &= \frac{1}{2}\left(\frac{(f(A) + f(\bar{A}))^2}{f(S)} + \mathbf{x}^f(A)\Phi_{f|_A}(A) + \mathbf{x}^f(\bar{A})\Phi_{f|_{\bar{A}}}(\bar{A})\right) \\ &= \frac{1}{2}(f(S) + \Phi), \end{aligned}$$

where the final equality comes from the fact that $\Phi_{f|_A}(A) = \sum_{s \in A} \mathbf{x}^{f|_A}(s) f|_A(s) = \sum_{s \in A} \frac{\mathbf{x}^f(s)}{\mathbf{x}^f(A)} f(s)$—and similarly for $\Phi_{f|_{\bar{A}}}(A)$.

Now we turn to the searcher's strategy $\mathbf{p}^f$, which, with probability q, searches A first according to $\mathbf{p}^{f_A}$ and otherwise searches $\bar{A}$ first according to $\mathbf{p}^{f_{\bar{A}}}$, where

$$q = \frac{1}{2} + \frac{\Phi_{f|_A}(A) - \Phi_{f|_{\bar{A}}}(\bar{A})}{2f(S)}.$$

We prove by induction that this strategy ensures an expected search cost of at most V, where V is given by Equation (7). Let $s \in A$. Then, by induction, the expected search cost $C(\mathbf{p}^f, s)$ satisfies

$$\begin{aligned} C(\mathbf{p}^f, s) &\le V_{f|_A}(A) + (1-q)f(B) \\ &= \frac{1}{2}\left(f(A) + \Phi_{f|_A}(A)\right) + \left(\frac{1}{2} + \frac{\Phi_{f|_B}(B) - \Phi_{f|_A}(A)}{2f(S)}\right) f(B) \\ &= \frac{1}{2}\left(f(A) + f(B) + \mathbf{x}^f(A)\Phi_{f|_A}(A) + \mathbf{x}^f(\bar{A})\Phi_{f|_{\bar{A}}}(\bar{A})\right) \\ &= \frac{1}{2}(f(S) + \Phi) = V. \end{aligned}$$

This shows that the value is at most V. The case $s \in \bar{A}$ is similar, exchanging the roles of A and $\bar{A}$. □

Theorem 8 generalizes results in Alpern and Lidbetter [4] on expanding search on a rooted tree, where it is shown that the equilibrium hider strategy is unique and can be computed efficiently, as can an equilibrium searcher strategy. Expanding search on a tree is a series-parallel decomposable submodular search game.

We may consider the submodular search game in the context of scheduling jobs with processing times and precedence constraints. One player chooses an ordering of jobs, and the other player chooses a job; the payoff is the completion time of the chosen job. We can interpret this as a robust approach to scheduling, in which one job, unknown to the scheduler, has particular importance, and the scheduler seeks a randomized schedule that minimizes the expected completion time of that job in the worst case. This has a natural application to planning a research project or an innovation process, in which there are many directions the project can take, but it is unknown which task will be fruitful. Theorem 8 gives a solution of this scheduling game on series-parallel graphs. An interesting direction for future research would be to study the game on more general partial orders.

4. Final Remarks

We have shown that the notion of series-parallel decomposability is useful for solving both the submodular search problem and the submodular search game. A direction for future research could be to find some measure that captures the "distance" from being series-parallel decomposable and show that better approximations can be found when the problem is close to being series-parallel decomposable. It is shown in Ambühl et al. [8] that better approximations to the single-machine scheduling problem $1|\text{prec}|\sum w_jC_j$ can be found when the precedence constraints have low *fractional dimension*. It would be interesting to see whether this idea could be generalized to our setting.

The submodular search game that we have studied in this paper is a zero-sum game between one searcher and one hider. In search games on networks, one usually restricts attention to one searcher only because more searchers can divide up the space efficiently [2, p. 15]. However, in a submodular search game, such a division is impossible, and it would be interesting to study games with multiple searchers, which should relate to multimachine scheduling problems. Another extension would be to consider search games with selfish hiders. Selfish loading games have been studied, and an overview can be found in Vöcking [51]. These are games between one searcher and multiple hiders and with a modular payoff function, similar to the scheduling problem $1\|\sum w_jC_j$. A study of submodular search games with selfish hiders would extend this to $1 \mid \text{prec} \mid \sum w_jh(C_j)$, for concave h.

We end with a question. It is known that the complexity of determining an equilibrium searcher strategy in a specific search game on a network is NP-hard [53] (see also [36]). What is the complexity of determining equilibrium strategies in the submodular search game?

Acknowledgments

The authors thank Christoph Dürr for pointing out the connection between expanding search and scheduling. They also thank three anonymous reviewers, whose comments and suggestions inspired them to greatly improve this paper. In particular, they thank one reviewer for drawing their attention to the work of Pisaruk [40] and another reviewer for suggesting a more general notion of series-parallel decomposability and for corrections to the proof of Theorem 8.

References

[1] Adolphson D (1977) Single machine job sequencing with precedence constraints. *SIAM J. Comput.* 6(1):40–54.
[2] AlpernS, Gal S (2003) *The Theory of Search Games and Rendezvous*. Price CC, ed. Kluwer International Series in Operations Research and Management Sciences, vol. 55 (Kluwer, Boston).
[3] Alpern S, Howard JV (2000) Alternating search at two locations. *Dynam. Control* 10(4):319–339.
[4] Alpern S, Lidbetter T (2013) Mining coal or finding terrorists: The expanding search paradigm. *Oper. Res.* 61(2):265–279.
[5] Alpern S, Lidbetter T (2014) Searching a variable speed network. *Math. Oper. Res.* 39(3):697–711.
[6] Ambühl C, Mastrolili M (2009) Single machine precedence constrained scheduling is a vertex cover problem. *Algorithmica* 53(4):488–503.
[7] Ambühl C, Mastrolili M, Mutsanas N, Svensson O (2011) On the approximability of single-machine scheduling with precedence constraints. *Math. Oper. Res.* 36(4):653–669.
[8] Ambühl C, Mastrolili M, Svensson O (2007) Inapproximability results for sparsest cut, optimal linear arrangement, and precedence constrained scheduling. *Proc. 48th Annual IEEE Sympos. Foundations Comput. Sci.* (Institute of Electrical and Electronics Engineers, Washington, DC), 329–337.
[9] Bansal N, Khot S (2009) Optimal long code test with one free bit. *Proc. 50th Annual IEEE Sympos. Foundations Comput. Sci.* (Institute of Electrical and Electronics Engineers, Washington, DC), 453–462.
[10] Bansal N, Dürr C, Thang NK, Vásquez ÓC (2016) The local-global conjecture for scheduling with non-linear cost. *J. Scheduling* 20(3):239–254.
[11] Bellman R (1957) *Dynamic Programming* (Princeton University Press, Princeton, NJ).
[12] Chekuri C, Motwani R (1999) Precedence constrained scheduling to minimize sum of weighted completion times on a single machine. *Discrete Appl. Math.* 98(1):29–38.
[13] Chudak FA, Hochbaum DS (1999) A half-integral linear programming relaxation for scheduling precedence-constrained jobs on a single machine. *Oper. Res. Lett.* 25(5):199–204.
[14] Conforti M, Cornuéjols G (1984) Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Appl. Math.* 7(3):251–274.
[15] Correa JR, Schulz AS (2005) Single-machine scheduling with precedence constraints. *Math. Oper. Res.* 30(4):1005–1021.
[16] Cunningham W (1983) Decomposition of submodular functions. *Combinatorica* 3(1):53–68.
[17] Dürr C, Jeż Ł, Vásquez ÓC (2015) Scheduling under dynamic speed-scaling for minimizing weighted completion time and energy consumption. *Discrete Appl. Math.* 196(December):20–27.
[18] Edmonds J (1970) Submodular functions, matroids, and certain polyhedra. Guy R, Hanani H, Sauer N, Schönheim J, eds. *Combinatorial Structures and Their Applications* (Gordon and Breach, New York), 69–87.
[19] Fokkink R, Ramsey D, Kikuta K (2016) The search value of a set. *Ann. Oper. Res.* 256(1):1–11.
[20] Fujishige S, ed. (2005) *Submodular Functions and Optimization*, Annals of Discrete Mathematics, vol. 58 (Elsevier, Amsterdam).
[21] Gal S (1980) *Search Games* (Academic Press, New York).
[22] Gal S (2011) Search games. Cochran JJ, Cox LA Jr, Keskinocak P, Kharoufeh JP, Smith JC, eds. *Wiley Encyclopedia of Operations Research and Management Science* (John Wiley & Sons, Hoboken, NJ).
[23] Garey MR, Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco).
[24] Gittins JC, Glazebrook K, Weber RR (2011) *Multi-Armed Bandit Allocation Indices*, 2nd ed. (Wiley, New York).
[25] Gluss B (1959) An optimum policy for detecting a fault in a complex system. *Oper. Res.* 7(4):468–477.
[26] Hall LA, Schulz AS, Shmoys DB, Wein J (1997) Scheduling to minimize average completion time: Off-line and on-line algorithms. *Math. Oper. Res.* 22(3):513–544.
[27] Held M, Karp RM (1962) A dynamic programming approach to sequencing problems. *J. Soc. Indust. Appl. Math.* 10(1):196–210.
[28] Höhn W, Jacobs T (2015) On the performance of Smith's rule in single-machine scheduling with nonlinear cost. *ACM Trans. Algorithms* 11(4):25.
[29] Iwata S, Murota K, Shigeno M (1997) A fast parametric submodular intersection algorithm for strong map sequences. *Math. Oper. Res.* 22(4):803–813.
[30] Iwata S, Tetali P, Tripathi P (2012). Approximating minimum linear ordering problems. Gupta A, Jansen K, Rolim J, Servedio R, eds. *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, Lecture Notes in Computer Science, vol. 7408 (Springer, Berlin), 206–217.
[31] Lawler EL (1978) Sequencing jobs to minimize total weighted completion time. *Ann. Discrete Math.* 2:75–90.
[32] Lawler EL, Lenstra JK, Rinnooy Kan AHG, Shmoys DB (1993) Sequencing and scheduling: Algorithms and complexity. Graves SC, Rinnooy Kan AHG, Zipkin PH, eds. *Handbooks in Operations Research and Management Science*, vol. 4 (Elsevier, Amsterdam), 445–522.
[33] Lidbetter T (2013) Search games with multiple hidden objects. *SIAM J. Control Optim.* 51(4):3056–3074.
[34] Margot F, Queyranne M, Wang Y (2003) Decompositions, network flows and a precedence constrained single machine scheduling problem. *Oper. Res.* 51(6):981–992.
[35] Matula D (1964) A periodic optimal search. *Amer. Math. Monthly* 71(1):15–21.
[36] Megiddo N, Hakimi SL, Garey MR, Johnson DS, Papadimitriou CH (1988) The complexity of searching a graph. *J. ACM* 35(1):18–44.
[37] Megow N, Verschae J (2018) Dual techniques for scheduling on a machine with varying speed. *SIAM J. Discrete Math.* 32(3):1541–1571.
[38] Mitten LG (1960) An analytic solution to the least cost testing sequence problem. *J. Indust. Engrg.* 11(1):17–33.
[39] Monma CL, Sidney JB (1979) Sequencing with series-parallel precedence constraints. *Math. Oper. Res.* 4(3):215–224.
[40] Pisaruk NN (1992) The boundaries of submodular functions. *Comput. Math. Math. Phys.* 32(12):1769–1783.
[41] Pisaruk NN (2003) A fully combinatorial 2-approximation algorithm for precedence-constrained scheduling a single machine to minimize average weighted completion time. *Discrete Appl. Math.* 131(3):655–663.
[42] Queyranne M (1998) Minimizing symmetric submodular functions. *Math. Programming* 82(1–2):3–12.
[43] Rothkopf MH (1966) Scheduling independent tasks on parallel processors. *Management Sci.* 12(5):437–447.
[44] Schrijver A (2003) *Combinatorial Optimization, Polyhedra and Efficiency*, *Algorithms and Combinatorics*, vol. 24 (Springer, Berlin).
[45] Schulz AS (1996) Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds. Cunningham WH, McCormick ST, Queyranne M, eds. *Proc. 5th Internat. Conf. Integer Programming Combin. Optimization* (Springer-Verlag, Heidelberg), 301–315.

[46] Schulz AS, Verschae J (2016) Min-sum scheduling under precedence constraints. Sankowski P, Zaroliagis C, eds. *Proc. 24th Annual Eur. Sympos. Algorithms*, Leibniz International Proceedings in Informatics, vol. 57 (Schloss Dagstuh–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany), 74:1–74:13.
[47] Sidney JB (1975) Decomposition algorithms for single-machine sequencing with precedence relations and deferral costs. *Oper. Res.* 23(2): 283–298.
[48] Sidney JB, Steiner G (1986) Optimal sequencing by modular decomposition: Polynomial algorithms. *Oper. Res* 34(4):606–612.
[49] Smith WE (1956) Various optimizers for single-stage production. *Naval Res. Logist. Quart.* 3(1–2):59–66.
[50] Stiller S, Wiese A (2010) Increasing speed scheduling and flow scheduling. Cheong O, Chwa KY, Park K, eds. *Algorithms and Computation*, Lecture Notes in Computer Science, vol. 6507 (Springer, Berlin), 279–290.
[51] Vöcking B (2007) Selfish load balancing. Nisam N, Roughgarden T, Tardos É, Vazirani VV, eds. *Algorithmic Game Theory* (Cambridge University Press, Cambridge, UK), 517–542.
[52] von Neumann J, Morgenstern O (2007) *Theory of Games and Economic Behavior* (Princeton University Press, Princeton, NJ).
[53] von Stengel B, Werchner R (1997) Complexity of searching an immobile hider in a graph. *Discrete Appl. Math.* 78(1):235–249.
[54] Vondrák J (2010) Submodularity and curvature: The optimal algorithm. *RIMS Kôkyûroku Bessatsu* B23:253–266.