

# Scheduling Search Procedures: The Wheel of Fortune\*

Peter Damaschke

School of Computer Science and Engineering

Chalmers University, 41296 Göteborg, Sweden

`ptr@cs.chalmers.se`

## Abstract

Suppose that a player can make progress on  $n$  jobs, and her goal is to complete a target job among them, as soon as possible. Unfortunately she does not know what the target job is, perhaps not even if the target exists. This is a typical situation in searching and testing. Depending on the player's prior knowledge and optimization goals, this gives rise to various optimization problems in the framework of game theory and, sometimes, competitive analysis. Continuing earlier work on this topic we study another two versions. In the first game, the player knows only the job lengths and wants to minimize the completion time. A simple strategy that we call wheel-of-fortune (WOF) is optimal for this objective. A slight and natural modification however makes this game considerably more difficult: If the player can be sure that the target is present, WOF fails. However we can still construct in polynomial time an optimal strategy based on WOF. We also prove tight absolute bounds on the expected search time. In a final part we study two competitive-ratio minimization problems where either the job lengths or the target probabilities are known. We show their equivalence, describe the structure of optimal strategies, and we give a heuristic solution.

**Keywords:** searching, game theory, nonclairvoyant scheduling

---

\*This work has been supported by a grant from the Swedish Research Council (Vetenskapsrådet), file no. 621-2002-4574.

# 1 Introduction

## 1.1 Search-Schedule Games

We study a type of games against nature which we call *search-schedule games*. A *player* is searching for an object called the *target*, amongst several *candidate* objects. In order to verify or falsify that an object is the target, she has to perform a certain *job* on that object. The goal is find the target, i.e. to complete the target job soon, but unfortunately, the player does not know what the target is (otherwise there is nothing to investigate). The general question is which search strategy one should apply.

Such situations naturally arise in chemical or biological laboratories: We look for a substance, a gene, etc., which has a certain property. We know, or we have strong evidence, that this target exists, because its effect has been observed, and several candidates for the target have been isolated. Concrete applications may be: searching for trait genes with help of linkage analysis or knock-out experiments, drug testing, etc.

In the analysis of such a search task we can abstract from many details, like the nature of objects we consider, and the kind of job one has to perform to check the property in question. We just model those characteristics which are relevant to the performance of a search strategy. The player, i.e. the lab, has limited working capacities and can only go on with a certain amount of work per time. Hence, basically we have the problem of *scheduling* the candidate jobs so as to promote early success. This gives rise to interesting theoretical questions, but hopefully the considerations are also of immediate practical interest, as it is important to use work time and other resources in labs economically.

We focus on simple model assumptions which are often reasonable but can, of course, also be extended in various directions, in order to capture more complicated experimental situations. We will assume that all candidates are checked *independently*. This means, no piece of work is shared by the jobs of several candidates, and what we learn about one candidate does not imply anything new for other candidates. Moreover we allow *preemption*. The player can switch between the jobs and resume an interrupted job later, without penalties. Therefore we can (approximately, on a coarsened time axis) even say that we *mix* several jobs, that is, we can continuously assign fractions of time to them. We also use the term *round-robin* schedule for a mix of jobs. Mixing is even more natural if the lab can distribute several jobs among equally fast workers, such that jobs are processed in parallel (with linear speed-up).

Clearly, under the above assumptions, the only relevant property of a job is its processing

time. Instead of physical time we may be interested in the costs of doing the several pieces of work involved in a job. In either case, we call the total cost of a job (time or other costs) the *length* of the job.

We may demand that the target job must be finished in any case, even if all other candidates have been already identified as false. That is, the target must be explicitly verified, in order to exclude the possibility that we missed the target in our current set of candidates. However, if the player is sure that the target is in the candidate set, she need not finish the last job in case that all other candidates are already falsified.

## 1.2 Discussion of models and outline of our technical contributions

First let us consider  $n$  jobs with known lengths  $c_i$ . That means, the player knows in advance for each object how long it will take to check it. We may scale any instance such that  $\sum_{i=1}^n c_i = 1$ . An adversary chooses the target, this reflects the player's ignorance. Suppose that the player wants to minimize her search time, i.e. the time to finish the target job. Trivially, the worst-case time is 1. However, the player may randomize. Intuitively it seems clear that the (expected) search time can be roughly halved by randomization, but exact results are less trivial, mainly because the jobs have different lengths. Amazingly, the two problem versions mentioned above behave quite differently: If the player is not sure that the target is in the candidate set, an optimal randomized strategy is easy to obtain (**Section 2**). For reasons that become obvious soon, we call it the WHEEL-OF-FORTUNE (WOF). But if the player is sure about presence of the target, the problem becomes much much more tricky. (Note that nothing is paradoxical with this statement.) However, we show in **Section 3** that an optimal mixed strategy with WOF as a building block can still be computed in polynomial time and uses only  $O(n)$  different permutations of jobs. The non-straightforward step is to design the structure of the mixed strategy. Also, an open problem in Section 3 indicates that this is not trivial: We don't know whether our proposed strategy has already the minimum number of permutations. We are also able to prove tight absolute bounds on the expected search times of randomized strategies:  $1/2$  and  $9/16$ .

As a motivation of the problem above, think of a lab that has to solve many instances of this type. Then an optimal randomized strategy based on the known job lengths would, in the long run, give the best guarantee (subject to unlikely deviations) for the total amount of work, whatever the actual sequence of targets will be. Already some percents of saving could be attractive. We believe that provably good strategies for search problems can have long-term economical implications.

A remarkable feature of WOF and its modification (Section 3) is that they would start on longer jobs with higher probability than on shorter jobs, although the player only wants to identify one target job in each instance of the game. This might appear counterintuitive, however the reason is quite clear: These strategies minimize the expected search time under the worst circumstances, and in unfortunate cases targets might appear more frequently among longer jobs.

A rationale for this pessimistic assumption is to model other competing labs that might employ other strategies in the search for the same target. Targets with short decision time are easier to verify, but if no such discovery for any of our candidates has been published so far, this suggests that the target might be among the longer jobs. On the other hand, we do not know how many competitors are working on the same questions, and how *they* think strategically.

The case of complete ignorance of probabilities provides an upper bound for the performance of strategies against weaker adversaries. In other scenarios however it may be justified to assign prior probabilities  $p_i$  to the candidates. Moreover, it can be more sensible to minimize the ratio of search time and target length, rather than the absolute search time. This suggests several other models that we define next.

As above, lengths  $c_1, \dots, c_n$  are given, and a malicious adversary selects the target. But now we aim at minimizing the competitive ratio  $t/c_j$ , where  $t$  is the time needed by the player's strategy, and  $c_j$  is the target length. More precisely, we wish to minimize  $\max_j t/c_j$ . In the case of randomized strategies we replace  $t$  with the expected time. We refer to this problem as CRKL (Competitive Ratio for Known Lengths). It is sensible, e.g., if the job lengths differ a lot. Then the "regret" is high if the actual target job was short. Intuition tells that competitive strategies prefer short jobs.

We also introduce a "dual" scenario. This time, the target probabilities  $p_1, \dots, p_n$  are given, i.e., known to the player, but not the lengths  $c_i$ . In other words, the adversary has no control of the choice of the target, but she determines the job lengths and does not reveal them. Due to the unknown  $c_i$ , the absolute time is meaningless as a performance measure for a strategy, therefore we consider the competitive ratio.

It is crucial for the model to say when the adversary fixes the job lengths. If she can do this after the target has been chosen, she will make the non-target jobs very long compared to the target (infinite, for simplicity). Clearly, this creates the worst situation for the player. It is not hard to prove [8] that the deterministic round-robin strategy assigning time fractions  $x_i$  proportional to  $\sqrt{p_i}$  to the jobs gives the optimal expected competitive ratio

$\sum p_i/x_i = (\sum \sqrt{p_i})^2$ . Randomization won't help, because the game is Bayesian. Permanent preemption of jobs is necessary, of course, since otherwise the player could become obsessed with a long non-target job. This model reflects a situation where the player can verify the target, but not explicitly falsify the other candidates. The problem becomes more subtle if the adversary has to fix the lengths  $c_1, \dots, c_n$  before the target is determined. This adversary is weaker, as the player could be able to exclude non-targets, namely if she can finish those jobs. Hence there might exist a strategy with expected competitive ratio better than  $(\sum \sqrt{p_i})^2$ . We leave this as an open question. At least we can say that such a strategy would have to be randomized: Any deterministic strategy has to begin in a round-robin fashion with specified fractions  $x_i$ . Now, if the adversary takes  $c_i = x_i$  for all  $i$ , all jobs are finished at the same time, and the expected competitive ratio (expectation with respect to the random target choice) becomes  $\sum p_i/x_i$ . But we know already that this is at least  $(\sum \sqrt{p_i})^2$ .

Interestingly, the picture changes completely if we want to minimize the ratio of expected costs in the second model. We call this problem REKP (Ratio of Expectations for Known Probabilities). As a motivation, think of a lab that has to solve many instances of similar order of magnitude permanently. In the long run, it makes sense to compare the total amount of working hours to the total length of targets. This amortized comparison is more "mild" than the expected ratio of costs for every instance. In view of one instance, we divide our expected search time by the *expected* target length (rather than the actual target length).

We will show in **Section 4** that REKP and CRKL are equivalent as optimization problems. In **Section 5** we compute the value of the CRKL game, prove some block-structure of certain optimal strategies, and propose an approximation heuristic. However the complexity status of CRKL remains an intriguing open question.

### 1.3 Related literature

We refer to [20] and [4] for general introductions to game theory and competitive analysis, respectively.

We got results for competitive ratios of several other search-schedule games earlier [8], starting from a concrete motivation from genetics. The scenarios in [8] are different from CRKL, relationships are already discussed above. Games where the absolute time shall be minimized, as in Section 2-3 of the present paper, are not studied in [8].

Our field is loosely related to "fair" variants of nonclairvoyant scheduling [19, 2, 6, 18] as

well as geometric search problems [17]. Online construction of hybrid algorithms for search tasks is stated as another motivation of this line of research in [17], following [13].

The cow-path problem is to find an object hidden at a point on one of  $n \geq 2$  rays emanating from a central point. The search time is proportional to the distance walked. Optimal deterministic and randomized competitive ratios have been determined for plenty of variants of the problem [1, 12], among others the variant with bounded (but not fixed) distances, e.g. in [16]. Our first problem can be considered as the cow-path problem on  $n$  rays where the putative position of the object is already known for each ray. Due to the known distances it also makes sense to minimize the absolute search time, alternatively to the competitive ratio. Apparently, this natural variant is not addressed in the “cow-path literature”. A set of possible probability distributions of the unknown location is assumed in [11], and algorithms for minimizing the search time and the competitive ratio are given for the 2-ray case. Reduction to a matrix game is explicitly utilized there.

This game of searching a hidden object by walking along paths can be considered on arbitrary graphs. The work in [10] contains results for the expected search time in different types of graphs, depending on Eulerian properties. The complexity of the problem with bounded resources (i.e., search length) is studied in [21]. These papers which also work with the interpretation as matrix games are closest to the present work, but the type of results is different. Remarkably, none of the above mentioned and other investigated articles quote any earlier results on exactly the problems we consider here. To our best knowledge our results are new, although the problems look very natural and fundamental.

A rich theory of searching for lost items (or persons) in the context of search-and-rescue operations, with many natural definitions of parameters and optimization goals, dates back to, e.g., [14, 5], published in 1946 and 1958, and has been further developed over decades. However the focus is on complicated models that capture many aspects of this application domain, different from our simplistic scenarios. A recent review, discussion, and bibliography can be found in [7], earlier surveys are [3, 9].

#### 1.4 Remarks about the notation

Throughout the paper, the terms *increasing* and *decreasing* are meant in the non-strict sense, with equality allowed. To avoid double sums and lengthy subscripts, the range of a summation is sometimes described just by a condition on the summation indices. For instance, if it is clear from context that  $k$  is a fixed integer,  $\sum_{i \leq j \leq k}$  means that  $i, j$  are the summation indices, and the sum is taken over all ordered pairs  $(i, j)$  that fulfill the condition.

## 2 The Wheel-of-Fortune Strategy

We study the game with  $n$  jobs of known lengths  $c_i$ , such that  $\sum_{i=1}^n c_i = 1$ , and a player who wants to minimize the expected search time in the worst case, i.e. maximized over all adversarial strategies to select a target.

Since the player does not learn anything new during a game until the target is identified, she can take all her random decisions already in the beginning, and then proceed with the jobs according to her choice. Thus any strategy of the player is equivalent to a probability distribution on the (deterministic) schedules. We say that a schedule *appears* in a strategy if it has positive probability there.

The following lemma says that preemption is useless, not only in the game considered here but also under more general assumptions.

**Lemma 2.1** *Consider any search-schedule game where all jobs end after finite time and the performance measure depends monotonically on the job completion times<sup>1</sup>, and an optimal strategy exists<sup>2</sup>. Then there is also a nonpreemptive optimal strategy, that is, a random distribution on the permutations of jobs.*

**Proof.** Consider a strategy and any of the schedules appearing there. If, in this schedule, a mix of jobs is processed during some time interval  $[t, u]$ , we denote by  $v \leq u$  the first moment after  $t$  where some of the jobs involved, say job  $i$ , is finished. Let be  $v := u$  if there is no such job, in that case let  $i$  be any of the jobs in this mix. If we do all pieces of job  $i$  allocated to  $[t, v]$  just before  $v$  instead, and “compact” the other jobs to the remaining interval after  $t$ , we will not aggravate the completion time of any job. In this way we can “demix” the whole schedule and replace it with a sequence of intervals where only one job is done at a time, without making the strategy worse.

Now consider a schedule without round-robin parts, as obtained above. It may still have preemptions. If the player interrupts a job  $i$  to switch to job  $j$  at some moment  $v$ , job  $i$  is not finished yet at  $v$ . If we exchange the two intervals, job  $j$  cannot end later than before, and the completion time of job  $i$  is not affected. Now it is clear how to get rid of all preemptions.  $\square$

---

<sup>1</sup>such as, in our case, the search time

<sup>2</sup>We have to demand this explicitly. In general there could be a sequence of strategies whose performances converge to some number that cannot be achieved exactly.

**Remark 2.2** Due to Lemma 2.1 we end up in a *finite* matrix game. The player's pure strategies are exactly the permutations of jobs, the adversary's pure strategies are the targets. Using game-theory terminology, we may speak of mixed strategies. The famous von Neumann theorem guarantees the existence of optimal mixed strategies for both player and adversary.

Given a mixed strategy for the player, let  $s(j)$  denote the expected search time if  $j$  is the target. The player has to adjust the probabilities of permutations so as to minimize  $\max_j s(j)$ .

We introduce the following strategy called WHEEL-OF-FORTUNE (WOF) which is barely random and computationally almost trivial: Arrange the jobs in an arbitrary cyclic order on a circle of circumference 1, assigning to job  $i$  a circular arc of length  $c_i$ . Choose a random point  $r$  (uniformly distributed) on the circle. Execute the job whose segment is hit by  $r$  completely, then continue in the cyclic order until the target job is finished. The naming WHEEL-OF-FORTUNE has obvious reasons.

**Theorem 2.3** *WOF has expected search time  $s(j) = \frac{1}{2}(1 + \sum_{i=1}^n c_i^2)$ , regardless of the target  $j$  and the cyclic order. Moreover, WOF is an optimal strategy.*

**Proof.** By linearity of expectation, the expected search time is the expected time  $1/2$  from  $r$  to completion of the target job, plus the expected time to work on the first job before point  $r$  is reached. Since job  $i$  is hit by  $r$  with probability  $c_i$ , and in that case the latter term is  $c_i/2$ , we get the result.

Let the adversary follow the mixed strategy to choose  $i$  with probability  $p_i = c_i$  as the target. She may even reveal her  $p_i$ . Then for any fixed permutation schedule, the expected search time (where expectation refers to the randomness on the adversary's side) is the same expression as before, by essentially the same calculation. Using linearity of expectation again, this remains true for mixed strategies. Hence the player cannot improve her result.  $\square$

**Remark 2.4** An obvious upper bound on the expected search time of WOF is  $\frac{1}{2}(1 + c)$ , where  $c$  denotes the maximum job length.

### 3 If the Target is Certainly in the Candidate Set

If the player knows in advance that the target is among the candidates, then, in order to identify the target, there is no need to run the target job in case that this is the only

remaining job (unless the player is supposed to explicitly verify the target in any case). This seemingly slight modification of the game changes its structure fundamentally, as it breaks the nice circular symmetry in the previous problem. As a consequence, WOF is no longer optimal, already for  $n = 2$ : Obviously it is always the best to run the shorter job. This observation can be immediately generalized: If  $c \geq 1/2$ , then any strategy that schedules the long job last is optimal, since the expected search time of the long job is  $1 - c \leq 1/2$ , and any strategy that schedules the long job earlier would make things worse, as the adversary may always select the long job as the target. The case  $c < 1/2$  turns out to be more difficult. It is treated in the remainder of this section.

First of all, by Lemma 2.1 preemption is useless also in this setting, hence the problem reduces to a finite game as above (Remark 2.2). An adversary strategy in this game is given by the probabilities  $p_j$  that job  $j$  is the target job. By fundamental results from game theory we have: If the player knew that the adversary applies her optimal strategy, she could even apply a pure strategy, that is, a certain permutation  $P$  of jobs, to achieve the optimal expected search time: Due to an obvious exchange argument, ordering the jobs by ascending  $c_j/p_j$  is optimal for the player. Now we study the structure of a pair of optimal strategies for both the player and adversary.

W.l.o.g. let the jobs be numbered in the order they have in  $P$ . Then the expected search time can be written as  $\sum_{i \leq j < n} c_i p_j + \sum_{i < n} c_i p_n$ . The first  $n - 1$  jobs are sorted by increasing ratios  $c_i/p_i$ , otherwise  $P$  would not be optimal, by an obvious exchange argument. Similarly, job  $n$  cannot be shorter than job  $n - 1$  in  $P$ .

The adversary would choose the  $p_i$  so as to maximize the expected search time the player can guarantee herself. Assume  $c_j/p_j < c_{j+1}/p_{j+1}$  for some  $j \leq n - 2$ . We transfer some amount of probability from  $j$  to  $j + 1$ , without changing the order of the  $c_j/p_j$ . It is easy to see that the above sum increases, hence the adversary's strategy was not optimal. This contradiction shows that all  $c_j/p_j$  for  $j < n$  must be equal. This implies for all  $j < n$  that

$$p_j = \frac{1 - p_n}{1 - c_n} c_j.$$

Moreover, any order of the first  $n - 1$  jobs gives the same expected search time

$$\frac{1 - p_n}{1 - c_n} \sum_{i \leq j < n} c_i c_j + p_n \sum_{i < n} c_i,$$

since the first sum does no longer depend on the numbering. However there still remain some degrees of freedom: We have to clarify which job is number  $n$ , and to fix its probability  $p_n$ . This would completely determine an optimal adversary strategy.

We have seen above that job  $n$  must not be shorter than job  $n - 1$ . Since any of the first  $n - 1$  jobs could be at position  $n - 1$ , we conclude that job  $n$  must be one with maximum length  $c$ . With  $p := p_n$ , the expected search time  $E$  becomes

$$E = \frac{1-p}{1-c} \sum_{i \leq j < n} c_i c_j + p(1-c).$$

Define

$$w := \frac{1}{1-c} \sum_{i \leq j < n} c_i c_j.$$

Since  $\sum_{j < n} c_j = 1 - c$ , we easily get

$$w = \frac{1}{1-c} \cdot \frac{1}{2} \left( (1-c)^2 + \sum_{j < n} c_j^2 \right) = \frac{1-c}{2} \left( 1 + \sum_{j < n} \left( \frac{c_j}{1-c} \right)^2 \right).$$

Thus,  $w$  is just the expected search time WOF would have for the previous problem, on the instance consisting of the first  $n - 1$  jobs. (Compare to Theorem 2.3, and note the scaling factor  $1 - c$ .) We write the expected search time concisely as

$$E = (1-p)w + p(1-c). \tag{1}$$

It remains to determine  $p$ . By the simple upper bound for WOF (Remark 2.4) we have

$$w \leq \frac{1-c}{2} \left( 1 + \frac{c}{1-c} \right) = \frac{1}{2} \leq 1-c,$$

thus expression (1) increases with  $p$ . However, if the adversary sets  $p$  too large, a permutation where the longest job is the last is no longer the player's best pure strategy. More specifically: As long as  $p \leq c$ , fraction  $c/p$  is not smaller than all the other (equal)  $c_j/p_j$ , so that the player would put the longest job at the end. This also shows  $p \geq c$ . Now, either the longest job is at the last position or not, and in the latter case it will be scheduled first, because  $c/p$  is now the minimum ratio. To express the expected search time in the latter case, define  $d \leq c$  as the length of the second largest job. Since still all  $c_j/p_j$  ( $j < n$ ) are equal, the second largest job is now scheduled last. The expected search time becomes

$$E = c + (1-p) \left( w - \frac{d^2}{1-c} \right). \tag{2}$$

To prove this claim, note the following: The first job (of length  $c$ ) must be executed first, regardless of the target. With probability  $1-p$ , one of the other jobs is the target. If we had to finish the target job, we would need to add expected search time  $w$ . However, the last job is never executed. It is the target with conditional probability  $\frac{d}{1-c}$  (as target probabilities

are already proportional to the lengths), and in that case we can deduct  $d$  from the search time. Expression (2) obviously decreases with  $p$ . Therefore the adversary maximizes the minimum of (1) and (2) by choosing  $p$  that makes them equal. Algebraic manipulation gives

$$p = \frac{c(1-c) - d^2}{(1-c)^2 - d^2} \quad (3)$$

and

$$1 - p = \frac{(1-c)(1-2c)}{(1-c)^2 - d^2}. \quad (4)$$

Together with (1), this yields the optimal expected search time

$$E = \frac{(1-c)(1-2c)w + c(1-c)^2 - (1-c)d^2}{(1-c)^2 - d^2}. \quad (5)$$

This is not a particularly lovely expression. We renumber the jobs so that  $c_1 \leq \dots \leq c_n$ . In particular,  $c = c_n$  and  $d = c_{n-1}$ . We also define  $s := \sum_{j \leq n-2} c_j^2$ , hence  $w = ((1-c)^2 + s + d^2)/2(1-c)$ . Now a few steps transform (5) into

$$E = \frac{1}{2} \left( 1 + \frac{(1-2c)s}{(1-c)^2 - d^2} \right). \quad (6)$$

We have shown:

**Theorem 3.1** *The optimal expected search time  $E$ , i.e. the value of the game, is given by (6). Consequently, this value can be computed in polynomial time from the job lengths, and  $1/2$  is a general lower bound on the expected search time if  $c < 1/2$ .  $\square$*

Next we give a mixed strategy for the player that guarantees this value  $E$  against any adversary and is therefore optimal. The lower bound proof suggests a strategy which is basically WOF except that the two longest jobs play a special role. It remains to adjust the probabilities. For the moment we abuse notation and denote the two longest jobs by  $c$  and  $d$ , and a WOF schedule of the other  $n-2$  jobs by “WOF”. Our strategy uses three types of schedules:  $(d, \text{WOF}, c)$ ,  $(c, \text{WOF}, d)$ , and  $(\text{WOF}, d, c)$ . The last option is selected with probability

$$q = \frac{s}{(1-c)^2 - d^2},$$

while any of the first two options is chosen with probability  $\frac{1}{2}(1-q)$ . Independently, the  $n-2$  shortest jobs are scheduled according to WOF.

Anyway, we have to show that, for each possible target job, only the expected time in (6) is needed to identify this target. We further simplify (6) by replacing  $s$  with  $q((1-c)^2 - d^2)$ . We get  $E = \frac{1}{2}(1 + q - 2qc)$ .

*Case 1:* The longest job (of length  $c$ ) is the target. Clearly, our expected search time is

$$\frac{1}{2}(1+q)(1-c) + \frac{1}{2}(1-q)c = \frac{1}{2}(1+q-2qc).$$

*Case 2:* The second longest job (of length  $d$ ) is the target. Similarly, our expected search time is

$$\frac{1}{2}(1-q)(d+1-d) + \frac{1}{2}q(2-2c) = \frac{1}{2}(1+q-2qc).$$

*Case 3:* One of the other jobs is the target. Let  $v$  be the WOF time for the first  $n-2$  jobs, that is,

$$\begin{aligned} v &= \frac{1-c-d}{2} \left( 1 + \sum_{j \leq n-2} \left( \frac{c_j}{1-c-d} \right)^2 \right) = \frac{1}{2} \left( 1-c-d + \frac{s}{(1-c)-d} \right) \\ &= \frac{1}{2} \left( 1-c-d + q \frac{(1-c)^2 - d^2}{(1-c)-d} \right) = \frac{1}{2}(1-c-d+q(1-c+d)). \end{aligned}$$

The expected search time for each of these jobs is

$$v + \frac{1}{2}(1-q)(c+d) = \frac{1}{2}(1+q-2qc).$$

Hence all jobs have the optimal expected search time  $E$ , and we can formulate the main result for our problem:

**Theorem 3.2** *There is an optimal mixed strategy (given above) where no more than  $3(n-2)$  different permutations appear, whose probabilities can be computed in polynomial time from the job lengths.  $\square$*

It remains open whether  $3(n-2)$  is the best “permutation complexity” of a polynomially computable optimal strategy. We also mention that the plain WOF strategy approximates the optimum very well if  $c$  is small. This might be good enough in practice, since the job lengths in real applications can be predicted only approximately.

Recall that Theorem 3.1 gives a uniform lower bound of  $1/2$ . This result is tight since, for any  $\epsilon > 0$ , there are instances of many short jobs where the player can obviously reach expected search  $1/2 + \epsilon$  for all jobs. An analogous question is how long the expected search time of the optimal strategy can be at all. Using our findings, it is not hard to give the complete answer.

**Theorem 3.3** *There exists an instance with expected search time  $9/16$ , and this is the worst case.*

**Proof.** Consider four jobs of equal length, i.e.  $1/4$ . We have  $c = d = 1/4$ ,  $s = 1/8$ , hence  $q = 1/4$ , and  $\frac{1}{2}(1 + q - 2qc) = 9/16$ . For the upper bound we come back to (6). It suffices to show that  $\frac{(1-2c)s}{(1-c)^2-d^2} > \frac{1}{8}$  is impossible. For any fixed  $c$  and  $d$ , by convexity of the square function,  $s$  is maximized if all “small” job lengths are  $d$ , which implies  $s \leq (1 - c - d)d$ . Thus the above assumption enforces  $\frac{(1-2c)d}{1-c+d} > \frac{1}{8}$ , equivalently  $d(7 - 16c) > 1 - c$ . If this could be true at all for some  $c$ , it would be true for the largest possible  $d$ , which is  $d = c$ . But this yields the contradiction  $(4c - 1)^2 < 0$ .  $\square$

## 4 A Duality Theorem for Different Adversaries and Competitive Ratios

Preemption is not advantageous for the player in REKP: The denominator in the competitive ratio is independent of the target, whereas the numerator is a monotone function of completion times of all jobs, hence Lemma 2.1 applies. Preemption is useless also for CRKL, since for every target the denominator is fixed and the numerator is monotone in the completion times. Hence, for both CRKL and REKP it is again enough to look at mixed strategies that are probability distributions on permutations. Now we are ready to show:

**Theorem 4.1** *Computing optimal strategies for CRKL and REKP are equivalent problems.*

**Proof.** Let an instance of CRKL with w.l.o.g.  $\sum_i c_i = 1$  correspond to a “dual” instance of REKP with  $p_i = c_i$ . (In the following we slightly abuse notation and do not distinguish between a problem and an instance.) Moreover, let any mixed strategy for CRKL correspond to a mixed strategy for the dual REKP where we just reverse each permutation and let the probabilities of permutations unchanged.

Given any mixed strategy for CRKL, the adversary will pick the target  $j$  with maximum expected competitive ratio  $E[t_j]/c_j$  where  $t_j$  is the completion time of job  $j$ . In the dual REKP she can assign arbitrary lengths  $d_i$  (but with  $\sum_i d_i = 1$ ) to the jobs, before the target is selected at random. Thus the expected target length in REKP is  $\sum_i d_i c_i$  (since  $p_i = c_i$ ). As for the player’s expected cost in REKP, note that a job  $j$  is done iff this or a later job in the random permutation is the target. Hence, the probability that job  $j$  must be done is, in any fixed permutation, the sum of  $c_i$  of all jobs  $i$  that do not precede  $j$ . Due to the reverse permutations, the joint probability that the mixed strategy executes  $j$  equals the completion time (!) from CRKL which is  $E[t_j]$ . It follows that the expected time spent on every job  $j$  is  $d_j E[t_j]$ , and by linearity of expectation the player’s expected cost is  $\sum_i d_i E[t_i]$ .

Ratio  $\sum_i d_i E[t_i] / \sum_i d_i c_i$  is maximized under constraint  $\sum_i d_i = 1$  if  $d_j = 1$  for some  $j$  with maximum  $E[t_j]/c_j$ . What we have shown is that the adversary achieves the same value in CRKL and REKP when the player runs a dual pair of mixed strategies. Thus, the optimal strategies for both problems form a dual pair.  $\square$

By this equivalence it suffices to study one of the two problems. In the following we discuss the more intuitive CRKL.

## 5 The Expected Competitive Ratio for Known Lengths

By a similar approach as in Section 3 we can determine the value of any instance of CRKL.

**Theorem 5.1** *For given job lengths  $c_j$ , the optimal expected competitive ratio is*

$$\max_k \frac{\sum_{i \leq j \leq k} c_i c_j}{\sum_{j \leq k} c_j^2}.$$

**Proof.** Suppose that the player knows the target probability  $p_i$  for every job  $i$ . Provided that we number the jobs in the order they are processed, the expected competitive ratio is

$$\sum_{j=1}^n p_j r_j = \sum_{j=1}^n p_j \sum_{i=1}^j c_i / c_j = \sum_{j=1}^n (p_j / c_j^2) \sum_{i=1}^j c_i c_j,$$

where  $r_j = \sum_{i=1}^j c_i / c_j$  denotes the actual competitive ratio if  $j$  is the target. The player's optimal strategy is to do the jobs by decreasing  $p_j / c_j^2$ , due to an exchange argument: If  $p_j / c_j^2 < p_{j+1} / c_{j+1}^2$  and we exchange jobs  $j$  and  $j+1$ , the net change of expected competitive ratio is  $c_j c_{j+1} (p_j / c_j^2 - p_{j+1} / c_{j+1}^2) < 0$ .

The adversary may choose the  $p_j$  so as to maximize the value of the optimal strategy. First of all, she may increase the sum without destroying the decreasing order of  $p_j / c_j^2$ . As long as  $p_j / c_j^2 > p_{j+1} / c_{j+1}^2$  for some  $j$  with  $r_j < r_{j+1}$ , she can do so by raising  $p_{j+1}$  at cost of  $p_j$ . In the following, a block is a maximal consecutive sequence of jobs  $j$  with increasing  $r_j$ . By the preceding discussion, the adversary can reach equal  $p_j / c_j^2$  within each block. Now, permutations of jobs in the block do not alter the expected competitive ratio. Hence we can w.l.o.g. assume that the player executes the jobs within each block by increasing lengths  $c_j$ . Next, consider some  $j$  being the last job in its block. From  $r_j > r_{j+1}$  it follows immediately  $c_j < c_{j+1}$ . Altogether, the  $c_j$  can only increase. We have shown that the adversary can suppose  $c_1 \leq \dots \leq c_n$ . An optimal adversary strategy against this schedule is optimal at all. Note that the sequence of  $p_j / c_j^2$  has to be decreasing (otherwise the player would not choose

this schedule). Thus we can write any solution  $(p_1, \dots, p_n)$  as a convex linear combination of solutions where the first  $k$  ratios  $p_i/c_i^2$  ( $1 \leq i \leq k$ ) are equal, followed by zeros. Consider any of the basic solutions, say with  $k$  nonzeros. Its expected competitive ratio is

$$\sum_{j=1}^k (p_j/c_j^2) \sum_{i=1}^j c_i c_j = a_k \sum_{j=1}^k \sum_{i=1}^j c_i c_j$$

for some  $a_k = p_j/c_j^2$  which has to be chosen so that  $\sum_{j=1}^k p_j = 1$ . It follows  $a_k \sum_{j=1}^k c_j^2 = 1$ , hence the expected competitive ratio is  $\sum_{i \leq j \leq k} c_i c_j / \sum_{j \leq k} c_j^2$ . By linearity of expectation, the expected competitive ratio of an arbitrary strategy with decreasing  $p_j/c_j^2$  is a convex linear combination of these values for the different  $k$ . Thus, the adversary's optimal choice is the  $k$ th basic solution where the above ratio is maximized.  $\square$

As earlier, the difficulty is to establish a mixed strategy for the player that achieves this value. We give a partial result describing the structure of certain optimal mixed strategies. The optimal deterministic strategy for CRKL is trivial: It schedules the jobs one-by-one, by increasing lengths  $c_1 \leq \dots \leq c_n$ . Hence the best deterministic competitive ratio is  $\max_k \sum_{i \leq k} c_i/c_k$ . By randomization we may balance different ratios and get a better expected value. For any mixed strategy,  $r_k$  denotes from now on the expected competitive ratio if job  $k$  is the target.

Let  $Opt$  be the set of optimal mixed strategies. We distinguish a subset  $Opt_1 \subseteq Opt$  of solutions where the *number* of jobs  $k$  with the maximal  $r_k$  is minimized. Next, let  $Opt_2 \subseteq Opt_1$  be the set of solutions in  $Opt_1$  where also the second largest value of  $r_k$  is minimized and the number of jobs attaining this second largest is minimized, too. In this way we define the  $Opt_i$  inductively, until we meet some  $i$  with  $Opt_i = Opt_{i+1}$ . We denote this last set  $Opt^*$  and call the solutions in  $Opt^*$  lexicographically optimal.

**Theorem 5.2** *There exists a lexicographically optimal mixed strategy for any CRKL instance with the following properties: The sequence of jobs, sorted by increasing lengths, is partitioned into blocks, all  $r_k$  within each block are equal, the  $r_k$  strictly decrease from block to block, and the order of blocks is the same in all appearing permutations.*

**Proof.** We prove by induction on  $\mu$  the existence of a solution in  $Opt^*$  that has this block structure for the first jobs in all appearing permutations, divided in  $\mu$  blocks. The hypothesis is vacuously true for  $\mu = 0$ . Suppose that it holds for some  $\mu$ . We consider any job  $j$  with the  $(\mu + 1)$ st largest  $r_k$  value. (All longer jobs are in the first  $\mu$  blocks.)

Assume that some job  $i$  with  $r_i < r_j$  is done immediately before job  $j$  in some appearing permutation  $\pi$ . Let  $\pi'$  be permutation  $\pi$  with jobs  $i$  and  $j$  switched. ( $\pi'$  may or may not already appear in the strategy.) Transferring some probability from  $\pi$  to  $\pi'$  improves  $r_j$  at cost of raising  $r_i$ . This contradicts lexicographic optimality. We conclude that all jobs with the  $(\mu+1)$ st largest  $r_k$  value follow immediately the  $\mu$ th block in all appearing permutations. This establishes the induction step.

It remains to show that this block partition also partitions the sequence of jobs sorted by increasing length. Assume, on the contrary, that two jobs  $i, j$  exist so that  $i$  is in an earlier block than  $j$ , and  $c_i > c_j$ . But then  $r_i < r_j$ , a contradiction.  $\square$

We conjecture that some converse is also true: If a strategy has the above block structure and the first block has the highest  $r_k$ , then it is optimal. It would be sufficient to show that there is no better balanced strategy (with all  $r_k$  equal) for the first block. From this the assertion would follow, since extending the first block can only raise its competitive ratio.

Once the blocks are decided, one can minimize the maximum  $r_k$  independently in every block by linear programming, details are straightforward. (Moreover, it suffices to optimize the first block and make sure that all  $r_k$  in later blocks are not larger.) This suggests an iterative algorithm: As long as the first block is not the worst, merge a block with currently highest  $r_k$  into the previous block, and rebalance the new block. Large blocks of size  $k$ , say, may give rise to  $k!$  permutations, and thus produce  $k!$  variables (probabilities) with positive value. On the other hand, one can limit the set of permutations by rather obvious heuristics (basically this is a column generation approach). In particular, if the  $c_i$  in a block do not differ so much, WOF will almost minimize the expected competitive ratio in this block. It seems that such a process quickly leads to  $\max r_k$  close to optimum.

## 6 Discussion

It would be interesting to obtain an explicit tradeoff for approximation guarantee vs. computation time for CRKL, based on the above preliminary thoughts, and to figure out the complexity status of the exact problem. Dimension arguments suggest that some optimal strategy is the product of independent random distributions of jobs in each block, with a few permutations per block (not more than the block size). However it remains open whether it can be computed in polynomial time.

In this paper we considered malicious adversaries. Things will become more complicated

for the more general diffuse adversaries [15]. At least two different definitions of competitive ratio come in mind [8].

## References

- [1] R.A. Baeza-Yates, J.C. Culberson and G.J.E. Rawlins, “Searching in the plane”, *Information and Computation* 106, pp. 234-252, 1993.
- [2] N. Bansal, K. Dhamdhere, J. Könemann and A. Sinha, “Non-clairvoyant scheduling for minimizing mean slowdown”, *20th Symposium on Theoretical Aspects of Computer Science, STACS’2003, LNCS 2607*, pp. 260-270.
- [3] S. Benkoski, M.G. Monticino and J.R. Weisinger, “A survey of the search theory literature”, *Naval Research Logistics* 38, pp. 469-494, 1991.
- [4] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*, Cambridge University Press, 1998.
- [5] A. Charnes and W.W. Cooper, “The theory of search: optimal distribution of search effort”, *Management Science* 5, pp. 44-50, 1958.
- [6] M. Chrobak, L. Epstein, J. Noga, J. Sgall, R. van Stee, T. Tichy and N. Vakhania, “Preemptive scheduling in overloaded systems”, *29th Int. Colloquium on Automata, Languages and Programming, ICALP’2002, LNCS 2380*, pp. 800-811.
- [7] D.C. Cooper, J.R. Frost and R. Quincy Robe, “Compatibility of Land SAR procedures with search theory”, prepared for U.S. Dept. of Homeland Security, 2003, available on <http://www.uscg.mil/hq/G-0/G-0PR/nsarc/nsarc.htm>
- [8] P. Damaschke, “Scheduling search procedures”, *Journal of Scheduling* 7, pp. 349-364, 2004.
- [9] J.R. Frost, “Principles of search theory”, part I-IV, in: *Response* 17, 1999.
- [10] S. Gal, “On the optimality of a simple strategy for searching graphs”, *Int. Journal of Game Theory* 29, pp. 533-542, 2001.
- [11] M.Y. Kao and M.L. Littman, “Algorithms for informed cows”, *AAAI-97 Workshop on On-Line Search*, 1997.

- [12] M.Y. Kao, J.H. Reif and S.R. Tate, “Searching an unknown environment: an optimal randomized algorithm for the cow-path problem”, *Information and Computation* 131, pp. 63-79, 1996.
- [13] M.Y. Kao, Y. Ma, M. Sipser and Y. Yin, “Optimal constructions of hybrid algorithms”, *5th ACM-SIAM Symposium on Discrete Algorithms, SODA 1994*, pp. 372-381.
- [14] B.O. Koopman, “Search and screening”, OEG Report 56, Columbia Univ. Division of War Research, 1946.
- [15] E. Koutsoupias and C. Papadimitriou, “Beyond competitive analysis”, *SIAM Journal on Computing* 30, pp. 300-317, 2000.
- [16] A. López-Ortiz and S. Schuierer, “The ultimate strategy to search on  $m$  rays?”, *4th Conf. on Computing and Combinatorics COCOON 1998, LNCS 1449*, pp. 75-84.
- [17] A. López-Ortiz and S. Schuierer, “Online parallel heuristics and robot searching under the competitive framework”, *8th Scandinavian Workshop on Algorithm Theory, SWAT 2002, LNCS 2368*, pp. 260-269.
- [18] M. Mastrolilli, “Scheduling to minimize max flow time: Offline and online algorithms”, *14th Symposium on Fundamentals of Computation Theory, FCT 2003, LNCS 2751*, pp. 49-60.
- [19] R. Motwani, S. Phillips and E. Torng, “Nonclairvoyant scheduling”, *Theoretical Computer Science* 130, pp. 17-47, 1994.
- [20] R.B. Myerson, *Game Theory: Analysis of Conflict*, Harvard University Press, 1991.
- [21] B. v.Stengel and R. Werchner, “Complexity of searching an immobile hider in a graph”, *Discrete Applied Mathematics* 78, pp. 235-249, 1997.