Algorithmica (2001) 30: 101–139
DOI: 10.1007/s00453-001-0003-0

Algorithmica

Optimal Search and One-Way Trading Online Algorithms

R. El-Yaniv,[1] A. Fiat,[2] R. M. Karp,[3] and G. Turpin[4]

Abstract. This paper is concerned with the *time series search* and *one-way trading* problems. In the (*time series*) *search* problem a player is searching for the maximum (or minimum) price in a sequence that unfolds sequentially, one price at a time. Once during this game the player can decide to accept the current price p in which case the game ends and the player's payoff is p. In the *one-way trading* problem a trader is given the task of trading dollars to yen. Each day, a new exchange rate is announced and the trader must decide how many dollars to convert to yen according to the current rate. The game ends when the trader trades his entire dollar wealth to yen and his payoff is the number of yen acquired.

The search and one-way trading are intimately related. Any (deterministic or randomized) one-way trading algorithm can be viewed as a randomized search algorithm. Using the *competitive ratio* as a performance measure we determine the optimal competitive performance for several variants of these problems. In particular, we show that a simple *threat-based* strategy is optimal and we determine its competitive ratio which yields, for realistic values of the problem parameters, surprisingly low competitive ratios.

We also consider and analyze a one-way trading game played against an adversary called *Nature* where the online player knows the probability distribution of the maximum exchange rate and that distribution has been chosen by Nature. Finally, we consider some applications for a special case of portfolio selection called *two-way trading* in which the trader may trade back and forth between cash and one asset.

Key Words. Time series search, One-way trading, Two-way trading, Portfolio selection, Online algorithms, Competitive analysis.

1. Introduction

1.1. *Time Series Search.* Consider the following *elementary search problem*. A player is searching for the maximum (resp. minimum) price of some asset. At each time period $i = 1, 2, \ldots, n$ the player obtains a price quotation p_i and must decide whether or not to accept this price. Once the player decides to accept some price p_j the game ends and the player's payoff (resp. cost) is p_j. The horizon n may or may not be known to the player and if the player has not decided to accept a price during the first $n - 1$ periods he must accept some minimum (resp. maximum) price m. In essence, in this elementary search

[1] Department of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel. rani@cs.techion.ac.il. R. El-Yaniv is a Marcella S. Geltman Academic Lecturer.
[2] Department of Computer Science, Tel Aviv University, Ramat Aviv 69978, Israel. fiat@math.tau.ac.il.
[3] International Computer Science Institute, 1947 Center St., Berkeley, CA 94704, USA, and Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA.
[4] Department of Computer Science, University of Toronto, Toronto, Ontario, Canada M5S 1A4. gt@cs.toronto.edu.

Received October 19, 1998; revised August 12, 1999. Communicated by Ming-Yang Kao.
Online publication February 1, 2001.

problem the player is confronting the decision problem of when he has acquired sufficient price sampling so that the accepted price is satisfying given what he already knows.

The elementary search problem has many variations and extensions. If the horizon n is known (resp. unknown) to the player, then the problem is termed *with known duration* (resp. *with unknown duration*). In the case of unknown duration the player is informed just before the last period so that he can accept the last offered price. Throughout this paper we assume a finite (known or unknown) duration. A search problem is *with recall* if some number of the most recent price offers are retained and at any period the player can choose to accept one of the retained offers. In another natural extension of the elementary search problem the player has to pay a *sampling cost* c_i to obtain the ith price quotation. In this case the total payoff (or return) of a player that accepts the price p_j is

$$p_j - \sum_{1 \le i \le j} c_i .$$

Applications of the time series search decision task are required and performed by virtually all economic agents and institutions. Therefore, search is a most fundamental feature of economic markets. Consider the following straightforward applications.

Job and employee search. In the job search the player is seeking employment. In each period the job seeker obtains one job offer which may be viewed as the lifetime earning from the job. A sampling cost may be associated with the generation of each offer and may include advertising costs, transportation costs, and perhaps the loss incurred from being currently unemployed.

Now consider an employer searching for an employee. We assume that the employer can test and quantify the suitability of candidates to the job offered. Such quantities correspond to price quotations. The sampling cost in this case may be attributed to advertisement, qualification tests, etc.[5]

Search for the lowest price of goods. Here the player needs to buy some goods that are sold at different stores at different prices. The player can obtain price quotations after identifying the relevant sellers. Here sampling fees may account for traveling or phone costs and for the time wasted. Like job and employee search this application is of fundamental importance because optimal search strategies determine the demand function which in turn determines the nature of the market itself.

1.2. *One-Way Trading.* Consider a trader who needs to exchange some initial wealth w_0, given in some currency (say, dollars), to some other asset or currency (say, yen). At the start of each trading period the trader obtains the current price quotation and must decide whether to accept it or wait for a better price. Typically, the cost of sampling a price quotation is negligible as prices are widely available in quotation services. Nevertheless, the trader may be required to pay some transaction fees for the exchange (e.g., to a financial institution).[6] Thus stated, the one-way trading problem is a direct application

[5] This employee search is closely related to the well known *secretary problem* [16], [1] where typically the objective is to accept one or more "secretaries" of best ordinal value among an ordered set of all secretaries.

[6] These transaction fees can be a function of the amount traded (i.e., dollars spent) or a function of the price (which is equivalent to the amount of yen received). Alternatively (or in addition), the fees can include a fixed cost.

of the time series search problem. Nevertheless, the trader can partition the initial wealth w_0 and exchange it sequentially in parts, each part in a different exchange rate. Clearly, this more general trading scheme may result in higher returns. As we shall later see, trading in parts is equivalent to a randomized search.

One-way trading algorithms can be applied in various economic situations. For instance, consider a fund manager who decides to change the position of some portfolio and enter (or exit) some market. In this case w_0 is the part of his wealth allocated to the new position. Another natural application arises when an individual, for the purpose of emigrating to a foreign country, sells local property in order to exchange the local currency received to the foreign currency.

1.3. *Previous Work Related to Time Series Search.* The search problem has received considerable attention in mathematical economics and operations research. Traditionally, a Bayesian approach has been employed: An optimal search strategy is sought under the assumption that the prior distribution of prices is given and stationary. In order to allow for analytical results it is usually further assumed that price quotations are independent observations of this distribution. Although the stationarity and independence assumptions are somewhat simplistic, this basic paradigm has given rise to a rich theory that relies on tools from the theory of optimal stopping [11]. It is beyond the scope of this paper to survey the Bayesian work related to search.[7]

Not surprisingly, the Bayesian approach derives search algorithms that are heavily dependent on the price prior distribution. Nevertheless, one striking feature of Bayesian optimal search algorithms (applicable to many problem variants) is that they have the following structure. Based upon the problem parameters, and, in particular, the assumed prior distribution, there is a single fixed critical number called the *reservation price* such that the optimal policy is to reject all prices below the reservation price and to accept any offer above it.[8]

The least acceptable assumption of classical Bayesian search models is that the probability distribution of prices is fully known to the player. Several models attempt to relax this assumption. For example, Rosenfield and Shapiro [28] studied cases where the price distribution is itself a random variable distributed according to probability law with some known moments (e.g., the price distribution is known to be normal with mean distributed according to some other probability law).

In this paper we attempt to circumvent almost all distributional assumptions by resorting to competitive analysis (defined in Section 1.4). Our only assumption is that the price generating process has finite (known or unknown) support.

1.4. *Competitive Analysis of Online Problems.* Let $\mathcal{P} = (\mathcal{I}, F, U)$ be a profit maximization problem where $\mathcal{I}$ is a set of possible inputs; for each $I \in \mathcal{I}$, $F(I)$ is the set of feasible outputs; U is a utility function such that for all I and $O \in F(I)$, $U(I, O) \in \Re$. Consider any algorithm ALG for the problem $\mathcal{P}$. Given any input I, ALG computes a feasible output $O \in F(I)$. We denote the profit (or return) of ALG on the input instance

[7] The reader is referred to the excellent surveys by Lippman and McCall that discuss Bayesian solutions for a variety of search problem variants [23], [24].

[8] In the case of search with known duration the reservation price can change dynamically.

I by $\text{ALG}(I) = U(I, O)$. Typically, each input can be represented as a finite sequence $I = i_1, i_2, \ldots, i_n$ and a feasible output O can also be represented as finite sequence $O = o_1, o_2, \ldots, o_n$.

An algorithm ALG computes *online* if for each $j = 1, \ldots, n-1$, ALG must compute o_j before i_{j+1} is given. An algorithm is *offline* if it can produce a feasible output given the entire input sequence. We denote an optimal offline algorithm by OPT. By definition, for each input sequence I the return of OPT is $\text{OPT}(I) = \sup_{O \in F(I)} U(I, O)$.

An online algorithm is c-competitive if for any $I \in \mathcal{I}$,

$$\text{ALG}(I) \geq \frac{1}{c} \cdot \text{OPT}(I). \tag{1}$$

In this case we also say that ALG attains a competitive ratio c. The least competitive ratio that ALG attains is called *the* competitive ratio of ALG.

The competitive ratio is thus a worst-case performance measure. Any c-competitive online algorithm is guaranteed to return at least a fraction $1/c$ of the optimal offline profit no matter how unfortunate or erratic the future will be.

The use of the competitive ratio to measure performance of online algorithms is called *competitive analysis*.[9] Competitive analysis has been used in the computer science literature for over 20 years. At first it was used implicitly for the online approximation of NP-hard problems (see, e.g., [17], [19], [20], and [30]). Somewhat later, the seminal work of Sleator and Tarjan [29] on virtual memory and dictionary management put forth the use of the competitive ratio as a general performance measure for online decision making.

Sometimes it is convenient to view the competitive analysis of an (online) problem as a two-person game between the online player and an adversary. The online player chooses an online algorithm ALG and informs the adversary of his choice. The adversary then chooses an input sequence I. The payoff to the adversary is the resulting performance ratio $\text{OPT}(I)/\text{ALG}(I)$ and the payoff to the online player is minus this quantity (i.e., the game is zero-sum).

As is generally the case with two-person zero-sum games, a randomized strategy is required to obtain optimal (expected) competitive performance. Extending the definition of the competitive ratio to randomized algorithms is straightforward. We simply substitute in (1) $E[\text{ALG}(I)]$ for $\text{ALG}(I)$ where the expectation is taken with respect to the distributions used by ALG. Since in the game corresponding to this definition the adversary is ignorant of the outcomes of the random choices made by the online algorithm this adversary is called *oblivious*.[10] Indeed, it is often the case that randomization (against an oblivious adversary) dramatically improves the competitive performance (see the classical results of [8] and [15] regarding metrical task systems and virtual memory management). As we shall later see randomization empowers the online player also in the search problem.

1.5. *Competitive Analysis: A Discussion.* The main attraction in using the competitive ratio for analyzing online algorithms is that there is no need to rely on statistical modeling of input sequences. Indeed, it is often extremely difficult to devise realistic statistical

[9] The term *competitive ratio* was termed by Karlin et al. [21].

[10] Other kinds of adversaries that generate input sequences adaptively (based on the outcomes of the online random choices) have been considered (see [5]).

models for possible inputs (which are always highly dependent on the particular application). This difficulty is often more extreme in complex dynamical environments such as economical systems. Strategic financial decision making is therefore a very attractive domain for competitive analyses. Moreover, in financial decision making it is often desirable to secure some minimal sure profit rather than expecting higher average profits while being exposed to severe risks. Essentially, this is what competitive analysis offers. Nevertheless, this risk aversion property of the competitive ratio is quite often a drawback since this performance measure can lead to overly defensive algorithms. Indeed, whenever decision makers do have some side information or partial (statistical) knowledge on the evolution of input sequences it would be a terrible waste to ignore it, which is precisely what the competitive ratio does. Recently, al-Binali [2] generalized the pure competitive analysis so that it can utilize predictions (in the form of partial knowledge on future input sequences) while retaining the natural risk aversion of the competitive ratio. In fact, this generalized competitive analysis framework allows for trading-off the associated risk with the potential reward.[11] This (generalized) competitive analysis offers a robust yet flexible complementary framework to the analysis of (financial) decision making under uncertainty. An additional advantage of the competitive ratio is that it offers a unified measure of performance under which any two strategies are comparable in some fundamental sense.

1.6. *Paper Organization*. This paper is organized as follows. In Section 2 we study the relationship between randomized search and one-way trading. In Section 3 we present a simple deterministic search algorithm of a reservation price policy type, which is optimal for (deterministic) search. We then show how randomization over the reservation price policies can dramatically improve the competitive ratio. In Section 4 we present a general "threat-based" policy for one-way trading (or search). This general policy yields optimal algorithms for a number of variants of the one-way trading problem. We provide in this section detailed analyses of four problem variants. In Section 5 we give a few numerical examples of the competitive ratios obtained in the previous sections, showing that for realistic values of the problem parameters we can obtain surprisingly low ratios. In Section 6 we study a problem variant called a "game against Nature." We define *Nature* as an adversary that chooses the probability distribution of the maximum exchange rate (the maximum rate is the optimal offline return). Although at the outset it seems that Nature is weaker than our ordinary adversary we prove that (with respect to one-way trading) Nature and the ordinary adversary are equivalent (in a sense to be made precise later). In Section 7 we apply our results for the one-way trading game to a *two-way trading* game in which the player can trade currencies back and forth. Lastly, in Section 8 we summarize our conclusions and indicate some directions for future work.

2. Randomized Search and One-way Trading. Notice that in the search problem the online player must accept one price and in the one-way trading problem the trader

[11] To demonstrate the utility of this new technique al-Binali has used some of the results in this paper (based on a preliminary version of our paper) showing how the competitive performance of a one-way trader can be boosted when he has the knowledge of useful predictions.

can partition his initial wealth and trade the parts sequentially, each part at a different exchange rate. Nevertheless, the search and one-way trading problems are closely related. Any deterministic (or randomized) one-way trading algorithm that trades the initial wealth in parts can be interpreted as a randomized search algorithm and vice versa. This follows from the fact that any (randomized) one-way trading algorithm is equivalent to a randomized trading algorithm that trades the entire wealth at once (at some randomly chosen period). Further, any randomized trading algorithm that trades the entire wealth at once is equivalent to a deterministic algorithm that trades the initial wealth in parts. Formally we have

THEOREM 1. (i) *Let* ALG_1 *be any randomized one-way trading algorithm. Then there exists a deterministic one-way trading algorithm* ALG_2 *such that for any exchange rate sequence* σ, $\mathbf{E}[\text{ALG}_1(\sigma)] = \text{ALG}_2(\sigma)$. *The reverse statement is also true*: (ii) *Let* ALG_2 *be any deterministic one-way trading algorithm. Then there exists a randomized algorithm* ALG_1 *that makes only a single trade such that* $\mathbf{E}[\text{ALG}_1(\sigma)] = \text{ALG}_2(\sigma)$ *holds for all* σ.

PROOF. Let ALG_1 be any randomized one-way trading algorithm. In particular (and using game-theoretic terminology), ALG_1 may be a *mixed* strategy (a distribution over deterministic algorithms) or a *behavioral* strategy (whose daily transactions are chosen randomly). Since in this online game (between the online player and the adversary) the online player has perfect recall (she has no memory restrictions), we know that the set of behavioral strategies is a subset of the set of mixed strategies (see [3]). Thus, in any case we can assume without loss of generality that ALG_1 is a mixed algorithm, which is a probability distribution $\{w(a)\}$ over $\mathcal{A}$, the set of all deterministic algorithms. For any sequence of prices $\sigma = p_1, p_2, \ldots, p_n$, the expected return of ALG_1 is $\mathbf{E}_w[\text{ALG}_1(\sigma)] = \int_{\mathcal{A}} a(\sigma)\, dw(a)$. With respect to σ consider a deterministic algorithm ALG_2 that on period i spends a fraction $\int_{\mathcal{A}} s(i,a) dw(a)$ of its initial wealth where $s(i,a)$ is the amount spent by the deterministic algorithm a on period i. Thus the return of ALG_2 is

$$\begin{aligned}\text{ALG}_2(\sigma) &= \sum_{i=1}^{n} p_i \int_{\mathcal{A}} s(i,a)\, dw(a) \\ &= \int_{\mathcal{A}} \sum_{i=1}^{n} s(i,a)\, p_i\, dw(a) \\ &= \int_{\mathcal{A}} a(\sigma)\, dw(s) = \mathbf{E}_w[\text{ALG}_1(\sigma)].\end{aligned}$$

To prove part (ii) consider a deterministic algorithm that trades a fraction s_i of its initial wealth at the ith period, $\sum_i s_i = 1$. Now consider a randomized algorithm ALG_1 that with probability s_i trades its entire wealth at the ith period. Clearly, the expected return of ALG_1 equals the return of the deterministic algorithm ALG_2. □

COROLLARY 2. *It follows that a* (*competitive*) *optimal deterministic one-way trading algorithm has the same return as an optimal randomized search algorithm. This implies that randomization cannot improve the competitive performance in one-way trading.*

Although randomization cannot help in one-way trading we will see that randomization is advantageous for search.

As noted earlier, both in the search and one-way trading problem the online player may be required to pay a sampling cost (and/or transaction cost) for each price quotation (and/or dollar traded). In this paper we consider simpler problem variants where there are no such costs. Also we make the assumptions that arbitrary fractions of money units can be traded.

REMARK 1. In general, the nature of the sampling/transaction fees will affect the competitive ratio. We state without proofs the following observations. (i) The competitive ratio of any one-way trading algorithm is independent of transaction costs determined by a fixed percentage applied to the amount spent. In this case the equivalence of Theorem 1 obviously holds. (ii) When we introduce transaction fees which are a fixed percentage applied to the prices, the competitive ratio will improve but Theorem 1 still holds. (iii) When fixed transaction costs are introduced the deterministic competitive ratio increases and there is no longer an equivalence between deterministic one-way trading algorithms and randomized search algorithms.

3. Competitive Search Algorithms. Throughout this paper we assume that prices (exchange rates) are chosen (by an adversary) from the real interval $[m, M]$ where $0 < m \leq M$. We define the maximum *fluctuation ratio* of possible prices to be $\varphi = M/m$. Competitive ratios of algorithms will be determined in terms of φ. The parameters m, M, or φ may or may not be known to the online player.

For a start, suppose that both m and M are known to the online player. In this case the optimal deterministic search strategy is the following *reservation price policy* (RPP): Accept the first price greater than or equal to $p^* = \sqrt{Mm}$. We call p^* the *reservation price*. Clearly, the optimal reservation price should balance the return ratios (offline/online) resulting by the following two events: (i) the maximum price encountered, $p_{\max}$, is $\geq p^*$ in which case the worst-case return ratio is M/p^*; (ii) $p_{\max} < p^*$ in which case the worst-case return ratio is $p_{\max}/m$. Therefore, the optimal reservation price p^* is the solution of $M/p = p/m$.

REMARK 2. It is possible to show that if only the fluctuation ratio φ is known (but not m or M), then no better ratio than the trivial one of φ is achievable.

One can dramatically improve the competitive ratio by using randomized algorithms. We now introduce simple randomized algorithms of "exponential threshold" type that achieve exponentially better competitive ratios. The basic idea of these algorithms is due to Levin [22].

Assume, for simplicity, that $\varphi = 2^k$ for some integer k. For $i = 0, 1, \ldots, k-1$, let RPP(i) be the (deterministic) reservation price policy with reservation price $m2^i$. Define EXPO to be a uniform probability mixture over the set $\{\text{RPP}(i)\}_{i=0}^{k-1}$. That is, before the start of the game EXPO chooses one of the RPP(i) strategies, each with probability $1/k$.

THEOREM 3 (Levin). *Let $\varphi = 2^k$ for some integer k. Algorithm* EXPO *is $(c(\varphi)\log\varphi)$-competitive with $c(\varphi)$ approaching 1 when $\varphi \to \infty$.*

PROOF. Let $p_{\max}$ be the posterior maximum price obtained. Let j be an integer satisfying $m2^j \le p_{\max} < m2^{j+1}$. The particular choice of $p_{\max}$ (and, therefore, of j) is controlled by the adversary. Note that $j \le k$ and since all prices are in $[m, M]$, $j = k$ if and only if $p_{\max} = M$. It will later be made clear that the choice $p_{\max} = M$ (and, thus, of $j = k$) is not optimal for the adversary so we can assume that $j \le k-1$. For any choice of j algorithm EXPO will return on average

$$\frac{m}{k}\left(k - j + \sum_{1 \le i \le j} 2^i\right) = \frac{m}{k}(2^{j+1} + k - j - 2).$$

Denote by $R(j)$ the return ratio, offline to online, obtained for a particular j. Since for any choice of j it is optimal for the adversary to choose $p_{\max}$ arbitrarily close to $m2^{j+1}$, we have

$$R(j) = k \cdot \frac{2^{j+1}}{2^{j+1} + k - j - 2}.$$

It is not hard to see that the real-valued function $R(j)$ obtains its maximum at $j^* = k - 2 + 1/\ln 2$ and it follows that the coefficient of k in $R(j)$ is almost 1 resulting in a competitive ratio that is greater than but approaching $k = \log\varphi$ (as φ and k grow). □

REMARK 3. Theorem 3 can be extended in two ways. First, similar result can be obtained when φ is not a power of 2. Second, exactly the same bound holds if the player does not know m and M but only knows φ

Algorithm EXPO can be modified to work even without knowledge of φ. Let $\mu = \{q(i)\}_{i=0}^{\infty}$ be any probability distribution over the natural numbers. Consider algorithm EXPO'_μ that acts as follows. After p_1, the first price, is revealed to the online player algorithm EXPO'_μ chooses the reservation price $p_1 2^i$ with probability $q(i)$.

LEMMA 1. *Algorithm EXPO'_μ is $2/q(\lfloor\log\varphi\rfloor)$-competitive against an oblivious adversary, where φ is the posterior global fluctuation ratio.*

PROOF. Let $p_{\max}$ be the maximum price obtained and assume that $p_1 2^j \le p_{\max} < p_1 2^{j+1}$ for some integer j. In the worst case OPT's return is less than, but arbitrarily close to, $p_1 2^{j+1}$ and algorithm EXPO'_μ returns at least $q(j)p_1 2^j$ on average. It follows that the expected competitive ratio of EXPO'_μ is not smaller than $2/q(j)$. By definition, $\varphi = p_{\max}/p_1$ and therefore $j \le \log\varphi < j+1$, so $\lfloor\log\varphi\rfloor = j$ and the proof is complete. □

In order to apply Lemma 1 and derive small competitive ratios for algorithm EXPO'_μ we need to construct an appropriate probability distribution μ. Small competitive ratios are obtained by taking appropriate converging infinite sums. For example, we can use

the Riemann zeta function $\zeta(x) = \sum_{i=1}^{\infty}(1/i^x)$. Specifically, for every positive ε the infinite sum $\sum_{i=1}^{\infty} 1/i^{1+\varepsilon}$ converges to the constant $\zeta(1+\varepsilon)$. It follows that $\mu_\varepsilon = \{1/\zeta(1+\varepsilon)(i+1)^{1+\varepsilon}\}_{i=0}^{\infty}$ is a probability distribution over the natural numbers. Hence by Lemma 1, $\text{EXPO}'_{\mu_\varepsilon}$ attains a competitive ratio of $2\zeta(1+\varepsilon)(j+1)^{1+\varepsilon} = O(\log^{1+\varepsilon}(\varphi))$ with φ being the posterior global fluctuation ratio. However, we can do even better. Consider the infinite sum $\sum_i 1/(i\log^{1+\varepsilon}(i))$. Lemma 1 yields the competitive ratio of

$$O\left(\log(\varphi)\cdot\log^{1+\varepsilon}(\log\varphi)\right). \tag{2}$$

Notice however that as ε decreases the constant in the "big-O" increases. Hence, the particular choice of the distribution μ_ε can only be optimized if some bounds on φ are known.

REMARK 4. It is possible to generalize the upper bound given by (2) and achieve an upper bound of

$$O\left(\log(\varphi)\cdot\log^{1+\varepsilon}(\underbrace{\log\log\cdots\log}_{k}\varphi)\right)$$

for every integer k.

4. Optimal "Threat-Based" Policy for One-Way Trading. As can be seen in the previous section the competitive ratio $O(\log\varphi)$ is attainable by the simple EXPO algorithm under minor assumptions on possible prices (i.e., the global fluctuation ratio is known). It turns out that the ratio $O(\log\varphi)$ is within a constant factor of the best possible. Nevertheless, to obtain an optimal competitive ratio, somewhat more involved algorithms and analyses are required. The optimal algorithms are best described as (deterministic) one-way trading algorithms and we focus on the one-way trading problem for the remainder of the paper.

The optimal performance is obtained by algorithms that obey the following *threat-based policy*. Let c be any competitive ratio that can be attained by some one-way trading algorithm. For a start, assume that c is known to the trader. For each such c the corresponding threat-based policy consists of the following two rules.

RULE 1. Consider trading dollars to yen only when the current rate is the highest seen so far.

RULE 2. Whenever you convert dollars, convert *just enough* to ensure that a competitive ratio c would be obtained if an adversary dropped the exchange rate to the minimum possible rate[12] and kept it there throughout the game.

Note that these two rules apply to all but the last trading day when, by the rules of the game, the trader must trade all remaining dollars to yen.

[12] The "minimum possible rate" is defined with respect to the information known to the trader. That is, it is m if m is known, and it is p/φ if only φ is known and p is highest price seen so far.

At the outset it is not clear how to follow such a policy, in particular, how to follow Rule 2 that requires trading a quantity that equals the amount of "just enough" dollars in order to ensure a competitive ratio of c. For now assume that it is possible to compute the quantities prescribed by Rule 2 and assume an algorithm that follows this policy. Such an algorithm converts dollars to yen based on the threat that the exchange rate will drop permanently to the minimum possible rate. For each attainable competitive ratio c the corresponding threat-based algorithm can be shown to be c-competitive. This can be intuitively justified as follows. Consider the first trade (exchange rate is p_1). Since the current exchange rate is the highest seen so far the algorithm considers a trade. Since the competitive ratio c is attainable by some deterministic trading algorithm, there exists some $s \geq 0$ such that the ratio c will still be attainable if s dollars are traded to yen. Further, the chosen amount of dollars s is such that the ratio c is so far guaranteed even if there will be a permanent drop of the exchange rate and no further trades will be conducted (except for one last trade converting the remaining dollars with the minimum possible exchange rate). In particular, there is no need to consider any exchange rate which is smaller than p_1. Similar arguments can be used to justify the choice of the amounts for the rest of the trades and thus intuitively this policy induces a c-competitive algorithm. A formal analysis follows.

REMARK 5. Denote the minimum possible exchange rate by m. Notice that as long as the exchange rates are not larger than cm, the threat-based trader should not trade any dollars to yen (except of course for the last day, when the trader must trade all remaining dollars to yen). This follows from Rule 2 because a competitive ratio of c is always attainable when the maximum rate is cm even if all dollars are exchanged at rate m.

We now develop some basic properties of threat-based trading. These properties will facilitate analyses of the threat-based policy for the following variants of the one-way trading game.

Variant 1: Known duration (i.e., n is known) with m and M known.
Variant 2: Unknown duration with m and M known.
Variant 3: Known duration and known φ.
Variant 4: Unknown duration and known φ.

For each of the above variants, we identify the corresponding optimal threat-based online algorithm and determine its competitive ratio.

Fix any two positive reals m and M with $m < M$, and any integer $n > 1$. An *exchange rate sequence*, σ, is an element of $[m, M]^n$. Thus, $\sigma = p_1, p_2, \ldots, p_n$, and for each $1 \leq i \leq n$, $p_i \in [m, M]$ is called the *exchange rate of the* ith *day* giving the number of yen that can be traded for one dollar on that day. Let $\Sigma^{(n)}$ denote the set of all such exchange rate sequences. A deterministic *one-way trading algorithm* is a function, $D\colon \Sigma^{(n)} \times \{1, 2, \ldots, n\} \to [0, 1]$, satisfying the following properties:

- D is nonincreasing;
- for each $\sigma \in \Sigma^{(n)}$, $D(\sigma, 0) = 1$, and $D(\sigma, n) = 0$.

$D(\sigma, i)$ is defined to be the number of remaining dollars just after the ith day when the algorithm trades its dollars in accordance with σ. To measure the performance of D we

make use of some more definitions. For each $\sigma \in \Sigma^{(n)}$, let

$$s_i \stackrel{\text{def}}{=} D(\sigma, i-1) - D(\sigma, i), \qquad i = 1, \ldots, n. \tag{3}$$

s_i is called *the* ith *transaction* and it thus gives the amount of dollars traded to yen on the ith day by the algorithm D. Given $D(\cdot)$ and the s_i, we define Y_D as follows:

$$\begin{aligned} Y_D(\sigma, 0) &\stackrel{\text{def}}{=} 0, \\ Y_D(\sigma, i) &\stackrel{\text{def}}{=} \sum_{j=1}^{i} s_j p_j, \qquad i = 1, \ldots, n. \end{aligned}$$

Thus, $Y_D(\sigma, i)$ gives the number of accumulated yen just *after* the ith transaction has been performed. Clearly, Y_D is nondecreasing and the monotonicity of D (and Y_D) corresponds to the *one-way* requirement. The *return* of D on the sequence σ, denoted by $P_D(\sigma)$, is defined to be the quantity $Y_D(\sigma, n)$. An *online* one-way trading algorithm is a one-way trading algorithm D such that the ith transaction is solely based on past and present exchange rates, and the parameters known to the online player.

Assume that the online player knows that there are at most n trading days. This corresponds to Variant 1 or 3.[13] Consider the threat-based strategy. Rule 1 requires that once a transaction has been made at some exchange rate, further transactions will be made only at higher exchange rates; rates that are the same or lower will be ignored. Hence, both OPT and the threat-based algorithm conduct transactions only when the exchange rate sequence reaches a new high. Therefore, in a worst-case analysis of the performance of the threat-based algorithm, we may assume that the exchange rate sequence consists of an initial segment of successive maxima of length $k \le n$. In addition, we can assume that the first rate p_1 is larger than the minimum possible rate times r where r is the target competitive ratio of the threat-based algorithm (see Remark 5). That is,

$$\text{(minimum possible rate)}\ r \le p_1 < p_2 < \cdots < p_k \le \text{(maximum possible rate)}.$$

However, in order to realize a threat, the adversary may choose $k < n$ and then take

$$p_{k+1} = p_{k+2} = \cdots = p_n = \text{(minimum possible rate)}.$$

Given a problem variant (1 or 3), let ALG be the optimal threat-based algorithm for that variant. The algorithm ALG will be associated with the function $D(\cdot)$, as defined above, and throughout this section we abbreviate $D(\sigma, i)$ (resp. $Y(\sigma, i)$) to D_i (resp. Y_i). Recall that the player starts with $D_0 = 1$ dollars and $Y_0 = 0$ yen. Also, recall that s_i (as defined in (3)) denotes the ith transaction. Specifically, $s_i = D_{i-1} - D_i$. It is easy to see that $p_i s_i = Y_i - Y_{i-1}$. Let r be the target competitive ratio that ALG is trying to achieve.

LEMMA 2. *If* ALG *is an r-competitive threat-based algorithm then for every $i \ge 1$,*

$$s_i = \frac{p_i - r \cdot (Y_{i-1} + D_{i-1} \cdot (\textit{minimum possible rate}))}{r \cdot (p_i - (\textit{minimum possible rate}))} \tag{4}$$

[13] The results for Variants 2 and 4 will be then derived from the results for Variants 1 and 3, respectively, by taking the limit $n \to \infty$.

and

$$Y_i + D_i \cdot (\textit{minimum possible rate}) = \frac{p_i}{r}. \tag{5}$$

PROOF. Since ALG is r-competitive, by Rule 2 it must be that

$$\frac{p_i}{Y_i + D_i \cdot (\text{minimum possible rate})} \le r. \tag{6}$$

Here the denominator represents the return of ALG if an adversary dropped the exchange rate to the minimum possible rate, and the numerator is the return of OPT for such an exchange rate sequence. That is,

$$\frac{p_i}{(Y_{i-1} + s_i p_i) + (D_{i-1} - s_i) \cdot (\text{minimum possible rate})} \le r. \tag{7}$$

Since (by Rule 2) ALG must spend the minimal s_i that satisfies inequality (7) and since the left-hand side in (7) is decreasing with s_i we must replace the inequality with equality:

$$\frac{p_i}{(Y_{i-1} + s_i p_i) + (D_{i-1} - s_i) \cdot (\text{minimum possible rate})} = r.$$

Equation (4) is obtained by solving this equality for s_i. Equation (5) is obtained by replacing the inequality in (6) with equality and rearranging. □

4.1. *Analysis of Variant* 1: *Known Duration with m, M Known.* To begin our analysis for Variant 1, we make use of (4) and (5) specializing them to the case in which the "minimum possible rate" is m as assumed for Variant 1. This yields the following expressions for the daily transactions:

$$s_1 = \frac{1}{r} \frac{p_1 - rm}{p_1 - m}, \tag{8}$$

and for $i > 1$,

$$s_i = \frac{1}{r} \cdot \frac{p_i - r \cdot (Y_{i-1} + D_{i-1} m)}{p_i - m}. \tag{9}$$

Then, by (5) at $i - 1$ instead of i we have $Y_{i-1} + D_{i-1} m = p_{i-1}/r$, so we obtain, for $i > 1$,

$$s_i = \frac{1}{r} \cdot \frac{p_i - p_{i-1}}{p_i - m}. \tag{10}$$

In the above formulas for the daily transactions, r is the target competitive ratio that the algorithm is attempting to achieve. Clearly the algorithm cannot attain an arbitrarily small r. For example, if $r = 1$ we see that $s_1 = 1$ (formula (8)) and the algorithm will spend its entire wealth on the first rate thus failing to achieve a competitive ratio of 1 (with any continuation of the exchange rate sequence that increases above the first rate).

Hence, we can obtain from these formulas r-competitive (threat-based) algorithms only by using sufficiently large values r. Our goal now is to identify the smallest achievable competitive ratio.

Let σ be an exchange rate sequence and let $r > 1$ be any real. We say that the threat-based algorithm A_r, as defined by formulas (8) and (10) applied with r, is *r-proper* with respect to σ if (i) the sum of daily transactions computed by A_r, when the exchange rate sequence is σ, is not larger than 1 (the initial wealth); and (ii) the resulting ratio of optimal offline return over online return $A_r(\sigma)$ with respect to σ is not larger than r.

LEMMA 3. *Let σ be any exchange rate sequence. If A_r is r-proper with respect to σ, then for any $r' \geq r$, $A_{r'}$ is r'-proper.*

PROOF. As noted, and without loss of generality, suppose that $\sigma = p_1, \ldots, p_k, m, \ldots, m$ with $m < p_1 < p_2 < \cdots < p_k$. For each day i, the daily dollar transactions s'_i as calculated by $A_{r'}$ are not larger than the respective amounts s_i as calculated by A_r. Specifically, for $i = 1$ we have, using (8),

$$s_1 - s'_1 = \frac{p_1}{p_1 - m}\left(\frac{1}{r} - \frac{1}{r'}\right) \geq 0.$$

Similarly, from (10) we have for $i > 1$,

$$s_i - s'_i = \frac{p_i - p_{i-1}}{p_i - m}\left(\frac{1}{r} - \frac{1}{r'}\right) \geq 0.$$

Therefore, $\sum_i s'_i \leq \sum_i s_i$, and since A_r is r-proper, $\sum_i s_i \leq 1$ and, therefore, $\sum_i s'_i \leq 1$. By the definition of the threat-based algorithm and since the competitive ratio r' is attainable, for every day $i \leq k$, $A_{r'}$ chooses a transaction that guarantees a competitive ratio of r' even in a case of a permanent drop to m. Therefore, A'_r is r'-proper with respect to σ. □

Let $\sigma = p_1, \ldots, p_k, m, \ldots, m$ be an input sequence. For any k, we want the daily transactions to satisfy $\sum_{i=1}^{k} s_i \leq 1$. Suppose for the moment that k is known to the online player. In this case, the *optimal* competitive ratio for a threat-based algorithm must be determined such that there will be no dollars remaining after the last purchase (on day k). In other words, the optimal competitive ratio of the threat-based strategy (with k known) has the property that

$$\sum_{i=1}^{k} s_i = 1.$$

Substituting for the s_i from (8) and (10) we obtain

$$1 = \frac{1}{r}\frac{p_1 - rm}{p_1 - m} + \frac{1}{r}\sum_{i=2}^{k}\frac{p_i - p_{i-1}}{p_i - m}. \tag{11}$$

After solving (11) for r we write

$$\begin{aligned} r &= r^{(k)}(p_1, p_2, \ldots, p_k) \\ &\stackrel{\text{def}}{=} 1 + \frac{p_1 - m}{p_1} \cdot \sum_{i=2}^{k}\frac{p_i - p_{i-1}}{p_i - m}. \end{aligned} \tag{12}$$

As shown in the following lemma, we can determine an attainable competitive ratio of threat-based algorithm in an n-day game by maximizing $r^{(k)}(p_1, \ldots, p_k)$ over all choices of $k \le n$ and $m \le p_1 < p_2 < \cdots < p_k \le M$. Define

$$r_n(m, M) = \sup_{\substack{k \le n, \\ m \le p_1 < \ldots < p_k \le M}} r^{(k)}(p_1, p_2, \ldots, p_k). \tag{13}$$

LEMMA 4. *Let σ be any exchange rate sequence. Then the threat-based algorithm $A_{r_n(m,M)}$ is $r_n(m, M)$-proper with respect to σ.*

PROOF. As usual, suppose that $\sigma = p_1, \ldots, p_k, m, \ldots, m$. Let $r = r^{(k)}(p_1, p_2, \ldots, p_k)$. By construction, the threat-based algorithm A_r is r-proper for σ. Since $r_n(m, M) \ge r$, by Lemma 3 $A_{r_n(m,M)}$ is $r_n(m, M)$-proper with respect to σ. □

By Lemma 4, $r_n(m, M)$ is an achievable competitive ratio for the problem. The rest of this section is devoted to calculating $r_n(m, M)$. Our analysis proceeds as follows: we first fix k, p_1, and p_k, and maximize over $\{p_i\}_{i=2}^{k-1}$. Then we maximize over p_k, next over p_1, and, lastly, over k. Without loss of generality we assume that $k > 1$ since for the choice $k = 1$, a competitive ratio of 1 is trivially achieved. The sequence of following lemmas lead to the evaluation of (13).

LEMMA 5. *For fixed $k > 1$, p_1 and p_k,*

$$\max_{p_1 < p_2 < \cdots < p_{k-1} < p_k} \sum_{i=2}^{k} \frac{p_i - p_{i-1}}{p_i - m} = (k-1)\left(1 - \left(\frac{p_1 - m}{p_k - m}\right)^{1/(k-1)}\right)$$

and the maximum is obtained when for every $2 \le i \le k$,

$$\frac{p_i - p_{i-1}}{p_i - m} = 1 - \left(\frac{p_1 - m}{p_k - m}\right)^{1/(k-1)}. \tag{14}$$

PROOF. For each i, set $x_i = p_i - m$. Hence,

$$\begin{aligned} \sum_{i=2}^{k} \frac{p_i - p_{i-1}}{p_i - m} &= \sum_{i=2}^{k} \frac{x_i - x_{i-1}}{x_i} \\ &= k - 1 - \sum_{i=2}^{k} \frac{x_{i-1}}{x_i}. \end{aligned}$$

However, by the geometric-arithmetic mean inequality,

$$\frac{(x_1/x_2 + x_2/x_3 + \cdots + x_{k-1}/x_k)}{k-1} \ge \left(\frac{x_1}{x_2} \cdot \frac{x_2}{x_3} \cdot \cdots \cdot \frac{x_{k-1}}{x_k}\right)^{1/(k-1)} = \left(\frac{x_1}{x_k}\right)^{1/(k-1)},$$

and equality is obtained if and only if all the terms in the left-hand side are equal. Hence,

$$\max_{\substack{p_i, \\ 2 \le i \le k-1}} \sum_{i=2}^{k} \frac{p_i - p_{i-1}}{p_i - m} = (k-1) - \min_{\substack{x_i, \\ 2 \le i \le k-1}} \left(\sum_{i=1}^{k} \frac{x_{i-1}}{x_i}\right)$$

$$= (k-1)\left(1-\left(\frac{x_1}{x_k}\right)^{1/(k-1)}\right)$$
$$= (k-1)\left(1-\left(\frac{p_1-m}{p_k-m}\right)^{1/(k-1)}\right). \qquad \square$$

From Lemma 5 we immediately obtain

$$\sup_{\substack{k\le n,\\ m\le p_1<\cdots<p_k\le M}} r^{(k)}(p_1, p_2, \ldots, p_k) = \sup_{\substack{k\le n,\\ m\le p_1<p_k\le M}} r^{(k)}(p_1, p_k),$$

where

$$r^{(k)}(p_1, p_k) \stackrel{\text{def}}{=} 1 + \frac{p_1-m}{p_1}\cdot(k-1)\left(1-\left(\frac{p_1-m}{p_k-m}\right)^{1/(k-1)}\right).$$

It is readily seen that $r^{(k)}(p_1, p_k)$ is increasing with p_k. Therefore, it is maximized when p_k takes its maximum possible value, M, and $\sup_{p_1,p_k} r(k)(p_1, p_k)$ reduces to $\sup_{p_1} r^{(k)}(p_1)$ where

$$r^{(k)}(p_1) \stackrel{\text{def}}{=} 1 + \frac{p_1-m}{p_1}\cdot(k-1)\left(1-\left(\frac{p_1-m}{M-m}\right)^{1/(k-1)}\right). \tag{15}$$

Abbreviate $\max_{p_1} r^{(k)}(p_1)$ by $r^{(k)}$.

LEMMA 6. *$\max_{p_1} r^{(k)}(p_1)$ exists; let p^* be a number in $[m, M]$ such that $r^{(k)}(p^*) = \max_{p_1} r^{(k)}(p_1)$. Then p^* is unique and*

$$r^{(k)}(p^*) = \frac{kp^*}{km + (p^*-m)}. \tag{16}$$

PROOF. We use the following substitutions:

$$u = (p_1-m)^{1/(k-1)},$$
$$v = (M-m)^{1/(k-1)}.$$

After simplification, we can write the derivative of $r^{(k)}(p_1)$ as follows:

$$\frac{dr^{(k)}(p_1)}{dp_1} = -\frac{u^k + mku - m(k-1)v}{p_1^2 v}. \tag{17}$$

Consider the numerator of (17). For every positive v and $k > 1$, the equation

$$u^k + mku - m(k-1)v = 0 \tag{18}$$

has a unique positive root, u^*. In other words, there exists $p^* = (u^*)^{k-1} + m$, that is, a stationary point of $r^{(k)}(p_1)$. It is straightforward to check that the quantity $(d^2 r^{(k)}(p_1)/dp_1^2)(p^*)$ is negative and thus $r^{(k)}(p^*)$ is a maximum.

We can rewrite (18) in the form

$$\frac{u^*}{v} = \frac{m(k-1)}{(u^*)^{k-1} + mk},$$

and if we substitute back for u^* and v we obtain

$$\left(\frac{p^* - m}{M - m}\right)^{1/(k-1)} = \frac{m(k-1)}{p^* - m + mk} = \frac{m(k-1)}{p^* + m(k-1)}. \tag{19}$$

The proof of (16) is then complete by substituting (19) in (15). That is,

$$\begin{aligned} r^{(k)}(p^*) &= 1 + \frac{p^* - m}{p^*}(k-1)\left(1 - \left(\frac{p^* - m}{M - m}\right)^{1/(k-1)}\right) \\ &= 1 + \frac{p^* - m}{p^*}(k-1)\left(1 - \frac{m(k-1)}{p^* + m(k-1)}\right) \\ &= 1 + \frac{(p^* - m)(k-1)}{p^* + m(k-1)} \\ &= \frac{kp^*}{p^* + m(k-1)}. \end{aligned}$$

□

We can now uniquely characterize the worst-case k-day sequence of exchange rates against the threat-based algorithm. Let $\hat{\sigma}_k = \hat{p}_1, \hat{p}_2, \ldots, \hat{p}_k$ denote this sequence. By Lemma 6, $\hat{p}_1 = p^*$. We also know, from an earlier discussion, that $\hat{p}_k = M$. In addition, by Lemma 5, for all $2 \le i \le k-1$,

$$\frac{p_i - p_{i-1}}{p_i - m} = 1 - \left(\frac{p_1 - m}{p_k - m}\right)^{1/(k-1)}, \tag{20}$$

so

$$\hat{p}_{i-1} = \hat{p}_i \left(\frac{\hat{p}_1 - m}{M - m}\right)^{1/(k-1)} + m\left(1 - \left(\frac{\hat{p}_1 - m}{M - m}\right)^{1/(k-1)}\right).$$

The behavior of the threat-based algorithm against $\hat{\sigma}_k$ can also be made clear now. For $1 \le i \le k$, denote by $\hat{s}_i$ the daily amounts that the threat-based algorithm spends when the exchange rate sequence is $\hat{\sigma}_k$.

LEMMA 7. *For all* $1 \le i \le k$, $\hat{s}_i = 1/k$.

PROOF. It is readily seen, by Lemma 5, that

$$\hat{s}_2 = \hat{s}_3 = \cdots = \hat{s}_k.$$

The proof is completed by showing that $\hat{s}_1 = 1/k$ as follows. We substitute the expression of $r^{(k)}$ from (16) for r in (8). That is,

$$\hat{s}_1 = \frac{1}{r^{(k)}} \cdot \frac{p^* - r^{(k)}m}{p^* - m}$$

$$
\begin{aligned}
&= \frac{p^* + m(k-1)}{kp^*} \cdot \frac{p^* - kmp^*/(p^* + m(k-1))}{p^* - m} \\
&= \frac{p^* + m(k-1)}{kp^*} \cdot \frac{p^*(p^* - m)}{(p^* - m)(p^* + m(k-1))} \\
&= \frac{1}{k}.
\end{aligned}
$$

□

Thus, against the (worst-case) exchange rate sequence $\hat{\sigma}_k$, the threat-based algorithm obeys the conventional wisdom of investment advisers by employing a *dollar-cost averaging* strategy, in which an equal number of dollars is invested each day.

The next lemma yields a more informative characterization of $r^{(k)}$.

LEMMA 8. *$r^{(k)}$ is the unique solution, r, of*

$$r = k \cdot \left(1 - \left(\frac{m(r-1)}{M-m}\right)^{1/k}\right). \tag{21}$$

PROOF. First we show that

$$r^{(k)} = k\left(1 - \left(\frac{\hat{p}_1 - m}{M-m}\right)^{1/(k-1)}\right). \tag{22}$$

Consider formula (10) for s_i, $i > 1$. We already know that $\hat{s}_i = 1/k$. Therefore, using (20) we obtain

$$
\begin{aligned}
\frac{1}{k} &= \frac{1}{r^{(k)}} \cdot \frac{\hat{p}_i - \hat{p}_{i-1}}{\hat{p}_i - m} \\
&= \frac{1}{r^{(k)}} \cdot \left(1 - \left(\frac{\hat{p}_1 - m}{M-m}\right)^{1/(k-1)}\right).
\end{aligned}
$$

This proves (22). Using (8) we derive the following expression for $\hat{p}_1$:

$$\hat{p}_1 = \frac{mr^{(k)}(k-1)}{k - r^{(k)}} \tag{23}$$

which is obtained when we solve the equation $1/k = (1/r^{(k)}) \cdot ((\hat{p}_1 - mr^{(k)})/(\hat{p}_1 - m))$ for $\hat{p}_1$. We now substitute the expression for $\hat{p}_1$, (23), in (22) and learn that $r^{(k)}$ is the solution of the following equation:

$$r^{(k)} = k\left(1 - \left(\frac{mk(r^{(k)} - 1)}{(k - r^{(k)})(M-m)}\right)^{1/(k-1)}\right). \tag{24}$$

Starting with (24), the following sequence of equivalent equalities completes the proof of the lemma:

$$r^{(k)} = k - k\left(\frac{mk(r^{(k)} - 1)}{(k - r^{(k)})(M-m)}\right)^{1/(k-1)}, \tag{25}$$

$$\frac{k-r^{(k)}}{k} = \left(\frac{mk(r^{(k)}-1)}{(k-r^{(k)})(M-m)}\right)^{1/(k-1)},$$

$$\frac{k-r^{(k)}}{k}\left(\frac{mk(r^{(k)}-1)}{(k-r^{(k)})(M-m)}\right) = \left(\frac{mk(r^{(k)}-1)}{(k-r^{(k)})(M-m)}\right)^{k/(k-1)},$$

$$\left(\frac{m(r^{(k)}-1)}{M-m}\right)^{1/k} = \left(\frac{mk(r^{(k)}-1)}{(k-r^{(k)})(M-m)}\right)^{1/(k-1)}.$$

Notice that the right-hand side of (25) is identical to the exponentiated term of (24). The lemma is complete by using (24) while substituting the left-hand side of (25) for this term. □

Consider the representation of $r^{(k)}$, (21). It is clear that $r^{(k)} < M/m$ since the competitive ratio M/m is attained by the trivial strategy that trades all dollars in the minimum possible rate, m, and the threat-based algorithm certainly performs strictly better. Hence, $m(r-1)/(M-m) = (r-1)/(M/m-1) < 1$, and then it is not hard to see that $r^{(k)}$ is strictly increasing with k. Therefore, we must take the pessimistic assumption that $k = n$. This yields the following corollary that gives the best attainable competitive ratio, $r_n(m, M)$, that the threat-based algorithm can attain for an n-day conversion game.

COROLLARY 4. *$r_n(m, M)$ is the root, r, of the equation*

$$r = n \cdot \left(1 - \left(\frac{m(r-1)}{M-m}\right)^{1/n}\right). \tag{26}$$

To summarize, we have two methods of calculating $r_n(m, M)$:

- Solve $dr^{(n)}(p_1)/dp_1 = 0$ for its root e^* (as in (17)) and then substitute it into (15).
- Solve (26) for r.

The next theorem states that $r_n(m, M)$ is the best competitive ratio that a one-way trading algorithm can achieve for the known duration case with n trading days.

THEOREM 5. *Let m, M, and n be given. Then $r_n(m, M)$ is the lowest possible competitive ratio for a known duration one-way trading game (with known m and M).*

PROOF. Let ALG be any deterministic algorithm different from the threat-based algorithm. Using an adversary argument we show that ALG cannot achieve a ratio smaller than r_n (as defined in (26)). Let $\hat{\sigma}_n = \hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n$ be the exchange rate sequence that maximizes $r(p_1, p_2, \ldots, p_n)$ for an n-day game (see the discussion after Lemma 6). On the first day we present $\hat{p}_1$ to ALG. If ALG spends less than $1/n$ dollars on this rate, then we end the game. Therefore, ALG must convert the remaining dollars with the minimum possible rate, m. If this is the case, ALG cannot achieve a ratio smaller than r_n; simply because $\hat{p}_1$ is chosen such that $1/n$ is the minimal amount that should be spent to guarantee the ratio r_n. Therefore, we assume that ALG spends on the first day an amount $s_1' \geq 1/n$. In this case we continue the game and present ALG with the next rate, $\hat{p}_2$. In general, if

at the end of the ith day the total amount in dollars that ALG spent is less than i/n we immediately end the game. Otherwise, we continue and present ALG with the next rate, $\hat{p}_{i+1}$, etc. Let j be the minimum i such that at the end of the ith day, the total amount spent by ALG so far is less than i/n. Denote by s'_i the amount spent by ALG on the ith day, $1 \le i \le j$. Since the game proceeded to the jth day we know that

$$\begin{aligned} \frac{1}{n} &\le s'_1, \\ \frac{2}{n} &\le s'_1 + s'_2, \\ &\vdots \\ \frac{j-1}{n} &\le \sum_{i=1}^{j-1} s'_i. \end{aligned}$$

However, by the choice of j,

$$\frac{j}{n} > \sum_{i=1}^{j} s'_i.$$

Therefore ALG could have gained more by spending exactly $1/n$ on each of the first $j-1$ days and by spending $\bar{s}_j = s'_j + (\sum_{i=1}^{j-1} s'_i - (j-1)/n)$ at the higher rate, $\hat{p}_j$. Even if this is the case, since $\bar{s}_j < 1/n$, ALG could not guarantee a competitive ratio of r_n since $\hat{p}_j$ is chosen such that exactly $1/n$ dollars should be spent on the jth day to attain a ratio of r_n. It follows then that ALG must coincide with the threat-based algorithm, achieving a ratio of r_n, or otherwise ALG incurs a higher ratio on this exchange rate sequence. □

REMARK 6. With respect to the proof of Theorem 5, notice that the adversary may end the game after any day, i, provided that the total amount spent by ALG is less than or equal to i/n. For example, if ALG spends exactly $1/n$ dollars on the first day, then by dropping the rest of the sequence to m, the adversary forces a competitive ratio of r_n on ALG. Thus, there are exactly n types of worst-case sequences against the threat-based algorithm. Namely, the sequences

$$\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_i, \underbrace{m, m, \ldots, m}_{n-i}, \qquad i = 1, 2, \ldots, n.$$

(Of course, the adversary may use other rates smaller than $\hat{p}_i$ instead of all the m's on the last days, except for the very last day.)

REMARK 7. It is easy to extend the proof of Theorem 5 to the case of randomized algorithms against oblivious adversaries. All that is needed is to note that an oblivious adversary can calculate the expected amounts that the algorithm will spend on each trading day. Then the proof is analogous to the proof of Theorem 5. Nevertheless, we already know from Corollary 2 that randomization cannot help in one-way trading.

4.2. *A Game Against a Lenient Adversary.* As can be seen in the proof of Theorem 5, the adversary can always force a competitive ratio of $r_n(m, M)$ on any algorithm. Nevertheless, for any practical purpose, it is most likely the case that we will confront a more lenient adversary—one which deviates from the worst-case sequence of exchange rates. In this section we describe an algorithm that always perform as well as the previous algorithm. However, on some exchange rate sequences, those which are not worst-case, the new algorithm strictly improves the offline to online ratio.

At the start of each trading day, the online player knows the number of remaining days, $n' \le n$ and is presented with an exchange rate, x. In addition, the player already accumulated $Y \ge 0$ yen and has $D \le 1$ dollars remaining. At this stage, the online player can calculate the best attainable competitive ratio for the remaining days *given* x, and use it to determine the amount of dollars to be spent. The idea is simply to assume that the current day is the first trading day of an n'-day trading period and that the rest of the rates will be chosen by an adversary (i.e., to maximize the competitive ratio). Since we consider x to be a "first" day rate we denote it by p'_1 and similarly, we denote the rest of the *worst-case* exchange rates by p'_i, $i = 2, 3, \ldots, n'$. Also, the (worst-case) daily transactions will be denoted by s'_i. Given D, Y, n', and p'_1 we now derive a formula for r', the best attainable ratio from this stage onward, as well as formulas for the s'_i. Note that for usage the online player need only know the quantity s'_1.

An application of (4) with $i = 1$ yields

$$s'_1 = \frac{p'_1 - r' \cdot (Y + Dm)}{r' \cdot (p'_1 - m)}. \tag{27}$$

For $i > 1$ we obtain from (10)

$$s'_i = \frac{1}{r'} \cdot \frac{p'_i - p'_{i-1}}{p'_i - m}. \tag{28}$$

Since the amounts to be spent sum up to D we have

$$\begin{aligned} D &= s'_1 + \sum_{i=2}^{n'} s'_i \\ &= \frac{p'_1 - r' \cdot (Y + Dm)}{r' \cdot (p'_1 - m)} + \frac{1}{r'} \cdot \sum_{i=2}^{n'} \frac{p'_i - p'_{i-1}}{p'_i - m}. \end{aligned}$$

Solving for r',

$$\begin{aligned} r' &= \frac{p'_1}{Dp'_1 + Y} \cdot \left[1 + \frac{p'_1 - m}{p'_1} \sum_{i=2}^{n'} \frac{p'_i - p'_{i-1}}{p'_i - m} \right] \\ &= \frac{p'_1}{Dp'_1 + Y} \cdot r^{(n')}(p'_1, \ldots, p'_{n'}), \end{aligned}$$

where $r^{(n')}(\cdot)$ is defined by (12) (with $k = n'$). Thus, the best ratio, r', is simply written as a "normalization" of the best ratio for a regular n'-day game ($D = 1$, $Y = 0$) with future rates $p'_2, p'_3, \ldots, p'_{n'}$. To determine the optimal ratio at this stage, we must optimize over

all possible future exchange rates. Notice however that we need not maximize over p_1' since it is already given. Hence, we apply Lemma 5 to obtain

$$\begin{aligned} r' &= r'(p_1', n', D, Y, m, M) \\ &\stackrel{\text{def}}{=} \frac{p_1'}{Dp_1' + Y} \cdot r^{(n')}(p_1'), \end{aligned}$$

where $r^{(n')}(p_1')$ is given by (15).

The improved, adaptive algorithm may be summarized as follows: Given an exchange rate x when there are ℓ trading days remaining ($\ell > 0$), the online player with D dollars and Y yen calculates $r' = r'(x, \ell, D, Y, m, M)$. If $x \leq mr'$, then the online player makes no transaction (see Remark 5). Otherwise, the online player trades s_1' (equation (27)) dollars to yen.

We now exemplify how this adaptive algorithm takes advantage of opportunities encountered in the trading period (i.e., deviations from the worst-case exchange rate sequence). In this example, let n' be an arbitrary number of days (≥ 2), and suppose that the first exchange rate we encounter is the maximum possible rate, M. Suppose also that at this stage the algorithm holds D dollars and Y yen. In contrast to the "worst-case" algorithm described in the previous section, we shall see that the adaptive algorithm identifies this opportunity and trades all available D dollars on this (fortunate) exchange rate.

First, by (15), $r^{(n')}(M) = 1$, so $r' = M/(DM + Y)$. Hence,

$$\begin{aligned} s_1' &= \left(M - \frac{M(Y + Dm)}{DM + Y}\right) \Bigg/ \left(\frac{M(M - m)}{DM + Y}\right) \\ &= \frac{MD(M - m)}{M(M - m)} = D. \end{aligned}$$

REMARK 8. In general, it can be shown that when using this improved algorithm, the sequence of ratios r' that is calculated by the online player in this manner is nonincreasing and if the adversary always deviates from the worst possible sequence, then the sequence of calculated ratios, r', is strictly decreasing.

4.3. *Analysis of Variant* 2: *Unknown Duration with m, M Known.* Since in this variant the number of trading days is not given to the online player, he must consider an arbitrarily large number of days. Define

$$r_\infty(m, M) \stackrel{\text{def}}{=} \lim_{n \to \infty} r_n(m, M).$$

Notice that r_n is monotone increasing with n. Therefore, r_∞ is larger than r_n for any n and therefore, by Lemma 3 the threat-based algorithm calculated with r_∞ (A_{r_∞}) is r_∞-proper for any input sequence σ and therefore r_∞ is an *attainable* competitive ratio for any finite trading period. On the other hand, as r_n is the lower bound for each n-day trading game, the lower bound for Variant 2 approaches (from below) $r_\infty(m, M)$ since the adversary may choose arbitrarily large n.

Using elementary calculus we calculate r_∞ as follows. Using the abbreviation $P = m(r-1)/(M-m)$ we calculate the limit $\lim_{n\to\infty} n \cdot (1 - P^{1/n}) = \lim_{n\to\infty} r_n(m, M)$. Notice that $n \cdot (1 - P^{1/n}) = (1 - P^{1/n})/(1/n)$, so by L'Hôpital's rule

$$\begin{aligned}\lim_{n\to\infty} n \cdot (1 - P^{1/n}) &= \lim_{n\to\infty} \frac{P^{1/n} \cdot \ln P / n^2}{-1/n^2} \\ &= \lim_{n\to\infty} -P^{1/n} \cdot \ln P \\ &= -\ln P.\end{aligned}$$

Hence, $r_\infty(m, M)$ is the unique solution, r, of

$$r = \ln \frac{M-m}{m(r-1)}. \tag{29}$$

It is not hard to see that $r_\infty = \Theta(\ln \varphi)$ where $\varphi = M/m$.

4.4. *Analysis of Variant* 3: *Known Duration with φ Known*. Here, the online player knows only the quantity $\varphi = M/m$ but not the actual bounds on possible exchange rates, m and M. Notice that the information about the minimum possible rate available to the online player varies online. A simple observation is that at the ith day the minimum possible rate (at this stage) is p_i/φ; here we assume that p_i is an element of the initial segment of exchange rate maxima. Therefore, as in the analysis of the second variant, we now make use of (4) and (5) specializing them to the case in which the *"minimum possible rate"* is p_i/φ as inferred from the above observation. First, from (5) we have

$$Y_i + D_i \frac{p_i}{\varphi} = \frac{p_i}{r}.$$

Solving for D_i we obtain

$$D_i = \varphi \cdot \left(\frac{1}{r} - \frac{Y_i}{p_i}\right). \tag{30}$$

Then, from (4) we obtain

$$s_i = \frac{p_i - r(Y_{i-1} + D_{i-1} \cdot p_i/\varphi)}{r \cdot (p_i - p_i/\varphi)}. \tag{31}$$

Since $Y_0 = 0$ and $D_0 = 1$, we have for the case $i = 1$,

$$s_1 = \frac{\varphi - r}{r \cdot (\varphi - 1)}. \tag{32}$$

Consider (30) for the case $i-1$ and substitute it for D_{i-1} in (31). After some simplification the resulting equation can, for $i > 1$, be written as

$$s_i = \frac{Y_{i-1}\varphi}{\varphi - 1} \cdot \left(\frac{1}{p_{i-1}} - \frac{1}{p_i}\right). \tag{33}$$

By definition, $Y_i = Y_{i-1} + p_i s_i$. If we use (33) to substitute for s_i we obtain the following recurrence relation for Y_k, $k > 1$:

$$\begin{aligned} Y_k &= Y_{k-1} + p_k \cdot s_k \\ &= Y_{k-1} + \frac{p_k Y_{k-1}\varphi}{\varphi - 1} \cdot \left(\frac{1}{p_{k-1}} - \frac{1}{p_k}\right) \\ &= Y_{k-1} \cdot \frac{1}{\varphi - 1} \cdot \left(\varphi \cdot \frac{p_k}{p_{k-1}} - 1\right). \end{aligned} \tag{34}$$

Recall that the base case for this recurrence relation is given by $Y_1 = s_1 \cdot p_1$. Using (32) we obtain

$$\begin{aligned} Y_k &= Y_1 \cdot \left(\frac{1}{\varphi - 1}\right)^{k-1} \cdot \prod_{i=2}^{k} \left(\varphi \cdot \frac{p_i}{p_{i-1}} - 1\right) \\ &= \frac{p_1(\varphi - r)}{r(\varphi - 1)} \cdot \left(\frac{1}{\varphi - 1}\right)^{k-1} \cdot \prod_{i=2}^{k} \left(\varphi \cdot \frac{p_i}{p_{i-1}} - 1\right). \end{aligned} \tag{35}$$

When k is known to the online player, the optimal online algorithm must convert all dollars by the end of the kth day. Therefore, D_k should equal 0. Thus, using (30),

$$\varphi \cdot \left(\frac{1}{r} - \frac{Y_k}{p_k}\right) = 0.$$

Hence, $r = p_k / Y_k$, and if we substitute (35) for Y_k we obtain

$$\begin{aligned} r &= \frac{p_k}{Y_k} \\ &= \frac{p_k / p_1}{((\varphi - r)/r(\varphi - 1)) \cdot (1/(\varphi - 1))^{k-1} \cdot \prod_{i=2}^{k} (\varphi \cdot (p_i / p_{i-1}) - 1)}. \end{aligned}$$

After solving for r and simplifying we obtain

$$\begin{aligned} r &= r^{(k)}(p_1, p_2, \ldots, p_k) \\ &\stackrel{\text{def}}{=} \varphi - \frac{(p_k / p_1) \cdot (\varphi - 1)^k}{\prod_{i=2}^{k} (\varphi \cdot (p_i / p_{i-1}) - 1)}. \end{aligned} \tag{36}$$

Hence, it remains to calculate

$$r_n(\varphi) \stackrel{\text{def}}{=} \max_{\substack{k \le n, \\ p_1 < p_2 < \cdots < p_k}} r^{(k)}(p_1, \ldots, p_k).$$

LEMMA 9. *For fixed p_1 and p_k,*

$$\max_{p_1 < p_2 < \cdots < p_k} r^{(k)}(p_1, p_2, \ldots, p_k)$$

occurs when, for every $2 \le j \le k$,

$$\frac{p_j}{p_{j-1}} = \left(\frac{p_k}{p_1}\right)^{1/(k-1)}.$$

PROOF. If we fix p_1 and p_k in $r^{(k)}(p_1, p_2, \ldots, p_k)$ it only remains to maximize $P = \prod_{i=2}^{k}(\varphi \cdot (p_i/p_{i-1}) - 1)$. First notice that since $\varphi \geq 1$ and $0 < p_1 < p_2 < \cdots < p_k$, every term in P is positive. Now, for any $2 \leq j \leq k-1$, p_j contributes to the product P only in the multiplication of two terms which we abbreviate by A_j:

$$A_j = \left(\varphi \cdot \frac{p_j}{p_{j-1}} - 1\right) \cdot \left(\varphi \cdot \frac{p_{j+1}}{p_j} - 1\right).$$

Consider

$$\frac{dA_j}{dp_j} = -\frac{\varphi \cdot (p_j^2 - p_{j+1} \cdot p_{j-1})}{p_{j-1} \cdot p_j^2}.$$

Clearly, $p_j' = (p_{j-1} \cdot p_{j+1})^{1/2}$ is a root of $dA_j/dp_j = 0$. In addition, $(d^2A_j/dp_j^2)(p_j')$ is negative. Therefore, for fixed p_{j-1} and p_{j+1}, A_j is maximized when

$$\frac{p_j}{p_{j-1}} = \frac{p_{j+1}}{p_j} = \left(\frac{p_{j+1}}{p_{j-1}}\right)^{1/2}.$$

It follows then that P is maximized when for every $2 \leq j < k$, $p_j/p_{j-1} = p_{j+1}/p_j$. We denote this ratio by ρ. Thus,

$$\rho^{k-1} = \frac{p_2}{p_1} \cdot \frac{p_3}{p_2} \cdot \cdots \cdot \frac{p_k}{p_{k-1}} = \frac{p_k}{p_1}$$

and $\rho = (p_k/p_1)^{1/(k-1)}$. □

As a direct conclusion of Lemma 9 $r^{(k)}(p_1, p_2, \ldots, p_k)$ can be rewritten as $r^{(k)}(p_k)$ where

$$\begin{aligned} r^{(k)}(p_k) &\stackrel{\text{def}}{=} \varphi - \frac{(p_k/p_1) \cdot (\varphi - 1)^k}{\prod_{i=2}^{n}(\varphi \cdot (p_k/p_1)^{1/(k-1)} - 1)} \\ &= \varphi - \frac{p_k/p_1 \cdot (\varphi - 1)^k}{\left(\varphi \cdot (p_k/p_1)^{1/(k-1)} - 1\right)^{k-1}}. \end{aligned} \tag{37}$$

By taking the derivative of $r^{(k)}(p_k)$ with respect to p_k, it can be seen that for every $\varphi > 1$, $r^{(k)}(p_k)$ increases with p_k, so let p_k take its maximum value $p_1\varphi$ to obtain

$$\begin{aligned} r^{(k)} &\stackrel{\text{def}}{=} \varphi - \frac{(p_1\varphi/p_1) \cdot (\varphi - 1)^k}{(\varphi \cdot (p_1\varphi/p_1)^{1/(k-1)} - 1)^{k-1}} \\ &= \varphi \cdot \left(1 - \frac{(\varphi - 1)^k}{\left(\varphi^{k/(k-1)} - 1\right)^{k-1}}\right). \end{aligned} \tag{38}$$

The final step to establish the best attainable competitive ratio for this variant is to optimize $r^{(k)}$ with respect to k.

LEMMA 10. *$r^{(k)}$ is monotone increasing with k.*

PROOF. Set

$$f(k) \stackrel{\text{def}}{=} \frac{(\varphi-1)^k}{\left(\varphi^{k/(k-1)}-1\right)^{k-1}}$$

and

$$g(k) \stackrel{\text{def}}{=} \frac{\left(\varphi^{k+1/k}-1\right)^k}{(\varphi-1)^k}.$$

By showing that for all $k \geq 2$ and $\varphi > 1$, $f(k) - f(k+1)$ is positive, we shall prove that $f(k)$ is monotone decreasing. By considering (38), this will readily prove that $r^{(k)}$ is monotone increasing with k.

$$\begin{aligned} f(k) - f(k+1) &= \frac{(\varphi-1)^k}{\left(\varphi^{k/(k-1)}-1\right)^{k-1}} - \frac{(\varphi-1)^{k+1}}{\left(\varphi^{(k+1)/k}-1\right)^k} \\ &= (\varphi-1)\cdot\left(\frac{1}{g(k-1)} - \frac{1}{g(k)}\right). \end{aligned} \tag{39}$$

Showing that f is decreasing is therefore equivalent to showing that g is an increasing function. Since log is an increasing function, it is sufficient to show that $\log(g(k))$ is increasing. Set $y \stackrel{\text{def}}{=} 1/k$ and $h(t) \stackrel{\text{def}}{=} \log(\varphi^{1+t}-1)$. Then

$$\begin{aligned} \log(g(k)) &= k\log\left(\frac{\varphi^{1+1/k}-1}{\varphi-1}\right) \\ &= k(\log(\varphi^{1+y}-1) - \log(\varphi-1)) \\ &= \frac{h(y)-h(0)}{y}. \end{aligned}$$

Since k is decreasing in y, we need to show that $(h(y)-h(0))/y$ is a monotone decreasing function of y. This is established in two steps: (i) $h(y)$ is a strictly concave function; (ii) $(h(y)-h(0))/y$ is a decreasing function.

We first show that (ii) follows from (i). If h is strictly concave, then

$$h(ty + (1-t)x) > t\cdot h(y) + (1-t)\cdot h(x)$$

for all t in $(0, 1)$ and $x \neq y$ such that $h(x)$ and $h(y)$ are defined. Take $x = 0$ to obtain

$$h(ty) > t\cdot h(y) + (1-t)\cdot h(0)$$

for $y > 0$. This yields

$$\frac{h(ty)-h(0)}{ty} > \frac{h(y)-h(0)}{y}, \qquad t \in (0,1),$$

which proves (ii). To prove (i) we show that $h'(y)$ is a decreasing function. We differentiate h:

$$h'(y) = \log(\varphi)\frac{\varphi^{1+y}}{\varphi^{1+y}-1}.$$

We write $z = \varphi^{1+y}$ and then $h'(y) = \log(\varphi)z/(z-1)$. Since $z/(z-1)$ is a decreasing function of z in the range $z > 1$, and since z is an increasing function of y, it follows that $h'(y)$ is strictly decreasing. It is well known that this implies that h is concave. This establishes (i). □

A direct conclusion of Lemma 10 is that the adversary will choose k to be n and we obtain the following:

THEOREM 6. $r_n(\varphi) = \varphi(1 - (\varphi-1)^n/(\varphi^{n/(n-1)} - 1)^{n-1})$.

4.5. *Analysis of Variant* 4: *Unknown Duration with* φ *Known*. Here, analogously to Variant 2, the best attainable competitive ratio is

$$r_\infty(\varphi) \stackrel{\text{def}}{=} \lim_{n\to\infty} r_n(\varphi).$$

Using elementary calculus it can be shown that

$$\lim_{n\to\infty} \frac{(\varphi-1)^n}{\left(\varphi^{n/(n-1)} - 1\right)^{n-1}} = (\varphi-1)\exp\left(-\frac{\varphi\ln\varphi}{\varphi-1}\right).$$

Therefore,

$$\begin{aligned} r_\infty(\varphi) &= \varphi\left(1 - (\varphi-1)\cdot\exp\left(-\frac{\varphi\ln\varphi}{\varphi-1}\right)\right) \\ &= \varphi - \frac{\varphi-1}{\varphi^{1/(\varphi-1)}}. \end{aligned}$$

LEMMA 11. $r_\infty(\varphi) = \Theta(\ln\varphi)$.

PROOF. We first prove the upper bound, $r_\infty(\varphi) = O(\ln\varphi)$. Let $f(x) \stackrel{\text{def}}{=} x - x/(x+1)^{1/x}$. We will show that for all $x > 1$, $f(x) \le \ln(x+1)$. This shall prove that $r_\infty(\varphi) = O(\ln\varphi)$ (for $\varphi > 2$), since $f(\varphi-1)+1 = r_\infty(\varphi)$. To prove the claim it is sufficient to prove that

$$\exp(f(x)) \le x+1,$$

or, in other words, that

$$\frac{e^x}{\exp(x/(x+1)^{1/x})} \le x+1.$$

This is equivalent to showing that

$$\frac{1}{x+1}\cdot e^x \le \left(\exp\left(\frac{1}{(x+1)^{1/x}}\right)\right)^x.$$

So it is sufficient to prove that

$$\frac{1}{(x+1)^{1/x}}\cdot e \le \left(\exp\left(\frac{1}{(x+1)^{1/x}}\right)\right).$$

Set $a = 1/(x+1)^{1/x}$. We need, then, to show that $g(x) \stackrel{\text{def}}{=} e^a - ea$ is nonnegative for all $x > 1$. This can be established by differentiation. We shall show that g' is negative, so g is decreasing. Then it can be shown that $\lim_{x\to\infty} g(x) = 0$ (and also it is easy to verify that $g(1) = \sqrt{e} - e/2 \simeq 0.2896$). This will complete the proof. It remains to show that $g' < 0$.

$$g'(x) = \frac{a(x\ln(x+1) + \ln(x+1) - x)\cdot(e^a - e)}{x^2(x+1)}.$$

It is not hard to verify that $x\ln(x+1) + \ln(x+1) - x$ is positive (for example, at $x = 1$ the value of this expression is $> \frac{1}{3}$). Set $h(x) = e^a - e$. We will show that h is negative in the interval $(1, \infty)$ and this will prove that $g' < 0$. We differentiate h:

$$h'(x) = \frac{e^a a(x\ln(x+1) + \ln(x+1) - x)}{x^2+1} > 0.$$

Therefore, h is increasing and it can be shown that $\lim_{x\to\infty} h(x) = 0$. Hence, h is negative in the interval $(0, \infty)$. This proves the upper bound.

For the lower bound, recall that $\varphi = M/m$. It is clear that for each $n \geq 2$, $r_n(m, M) \leq r_n(\varphi)$. Hence,

$$r_\infty(\varphi) \geq r_\infty(m, M) = \Theta(\ln\varphi). \qquad \square$$

5. Numerical Examples of Competitive Ratios of Search and One-way Trading Algorithms. In this section we provide some numerical examples of competitive ratios attained by some of the algorithms discussed so far. Consider Table 1. Clearly, the optimal threat-based algorithm for the unknown duration case with m and M known is always significantly better than all other algorithms. Notice that the deterministic reservation price policy RPP is better than algorithm EXPO for small values of φ, but the growth rate of EXPO's competitive ratio is approximately the logarithm of the growth rate of the competitive ratio of RPP. In general it is not hard to show that the limit of the ratio of EXPO's competitive ratio to the threat-based competitive ratio is $1/\ln 2 \approx 1.44$.

With respect to the known duration case, it is interesting to consider the rate of increase of the optimal competitive ratio as a function of the number of trading days n. It is not hard to see that the optimal competitive ratio grows very quickly to its asymptote. Nevertheless, there is still an advantage in playing short games. For instance, already at the $n = 20$th period $r_n(1, 2)$ almost reaches its asymptote, $r_\infty(1, 2) \approx 1.278$ (which is equivalent to guaranteeing 78.2% of the optimal offline return), at $n = 10$ the ratio achieved is 1.26 (79.3%), and at $n = 5$ the ratio is 1.24 (80.6%).

6. One-Way Trading Against Nature. Let μ be the maximum value that the exchange rate will assume. In this section we define and study a one-way trading game in which the online player knows the probability distribution of μ, and this distribution is chosen by an adversary. This model is typically referred to as a game against Nature (see [4], [26], and [12]). The result in this section reveals an interesting relationship between this game against Nature and the results of previous sections. At the outset it may appear

Table 1. Numerical examples of competitive ratios for some search and one-way trading algorithms (unknown duration).

Algorithm	Value of φ					
	1.5	2	4	8	16	32
RPP (m, M known)	1.22	1.41	2	2.82	4	5.65
EXPO (only φ known)*	1.5	2	2.66	3.42	4.26	5.16
THREAT (only φ known)	1.27	1.50	2.11	2.80	3.53	4.28
THREAT (m, M known)	1.15	1.28	1.60	1.97	2.38	2.83

*Or m and M known.

that Nature is weaker than the oblivious adversary since Nature chooses its policy based solely on the problem parameters whereas the oblivious adversary knows and makes use also of the online strategy. Moreover, Nature has to declare its strategy before the start of the game whereas the oblivious adversary keeps its strategy secret and reveals it only piecemeal online. Nevertheless, we shall see that the online player does not gain more power against Nature and in fact the competitive ratios of the strategy against the oblivious adversary and Nature are exactly the same.

We now define the new game more precisely. In contrast to the previous analyses we consider here a continuous time model. Fix m and M and let F be a cumulative distribution function of μ; that is, $F(x) = \Pr[\mu < x]$. Assume that the support of F is in $[m, M]$. Let $\mathcal{F}$ denote the set of all such cumulative distribution functions.

REMARK 9. Notice that our definition of F is slightly different from the conventional one. Usually, a cumulative distribution function F with support $[a, b]$ is any real function that satisfies: (i) for any x in $[a, b]$, $F(x)$ is nonnegative; (ii) $F(a) = 0$ and $F(b) = 1$; (iii) F is nondecreasing over $[a, b]$; and (iv) F is right-hand continuous in the open interval (a, b). Thus, for each $x, y \in [a, b]$, with $a < x < y$, $F(y) - F(x)$ is the probability that a number is chosen in $(x, y]$, and $F(y) - F(a)$ is the probability that a number is in $[a, y]$.

In our formulation, for each $x, y \in [m, M]$ with $x < y < M$, $F(y) - F(x)$ is the probability that a number is in $[x, y)$ and $F(M) - F(y)$ is the probability that a number is in $[y, M]$. Thus, our cumulative distribution functions are left-hand continuous in the open interval (m, M). The usefulness of this definition in our context is twofold. First it will simplify some of the calculations. More importantly, it will be essential later to guarantee the existence of Stieltjes integral when the integrand is a probability distribution function of this type (left-hand continuous) and the integrator is an ordinary (right-hand continuous) probability distribution function.

For each $\mu \in [m, M]$ consider the following exchange rate function, E_μ: $[m, M] \to \mathbb{R}[m, M]$,

$$E_\mu(t) \stackrel{\text{def}}{=} \begin{cases} t, & \text{if} \quad m \le t \le \mu, \\ m, & \text{if} \quad t > \mu. \end{cases}$$

Thus, E_μ is increasing to a global maximum μ and then drops to m. For any trading algorithm ALG, and $F \in \mathcal{F}$, let $P_{\text{ALG}}(F)$ denote the expected return of ALG with respect

to F when the algorithm starts with one dollar and trades its dollars in accordance with an exchange rate function E_μ where μ is a random number chosen with probability distribution F. Later, in Remark 11 we justify this choice of E_μ.

Given any F and any online algorithm ALG, we measure the performance of ALG by its *return ratio against* F, $r_{\text{ALG}}(F, m, M)$, which is defined as

$$r_A(F, m, M) \stackrel{\text{def}}{=} \frac{P_{\text{OPT}}(F)}{P_{\text{ALG}}(F)}.$$

Thus, the online player and the adversary's goals are, respectively, to minimize and to maximize the online algorithm's return ratio against F. With respect to a distribution function, F, we say that an online algorithm is *optimal* if no other online algorithm attains a smaller return ratio against F.

The new game is summarized as follows:

- Nature chooses a probability distribution $F \in \mathcal{F}$, so as to maximize the return ratio of the best online algorithm against F. The function F is then made known to the online player.
- Based on F, the online player chooses his best online trading algorithm, ALG.
- Nature chooses a random number, μ (with cumulative distribution F), which remains unknown to the online player.
- The game is played: the online algorithm ALG against the exchange rate function E_μ.

The main theorem we prove states that the smallest competitive ratio that the online player can achieve, for Variant 2, r_∞ (see Section 4.3), is equal to the largest return ratio that the adversary can force by choosing a probability distribution for the maximum exchange rate.

THEOREM 7. $r_\infty(m, M) = \max_F \min_A r_A(F, m, M)$.

Before attempting to prove Theorem 7, the question that we ask is: What is the online algorithm that maximizes the return ratio against a given F?

In order to answer this question we now reduce the class of candidates from all possible (continuous) algorithms to the small class of reservation price policies (see Section 1.3). Namely, for each $x \in [m, M]$, the reservation price policy RPP(x) is an online trading algorithm that trades all dollars at exchange rate x if the exchange rate function ever rises to the rate x. It follows (by definition) that if the exchange rate function never reaches x, RPP(x) trades all dollars at rate m.

It is not hard to see that for any $F \in \mathcal{F}$, the expected return of RPP(x) is

$$P_{\text{RPP}(x)}(F) = xF^c(x) + mF(x),$$

where $F^c(x) \stackrel{\text{def}}{=} 1 - F(x)$.

Similar to the discrete time setup we denote by D a continuous time trading function (algorithm). Consider $D^c(x) \stackrel{\text{def}}{=} 1 - D(x)$. By the properties of D we have: (i) D^c: $[m, M] \to [m, M]$; (ii) $D^c(m) = 0$, $D^c(M) = 1$; (iii) D^c is nondecreasing; and (iv) D^c is right-hand continuous. Hence, D^c is a cumulative distribution function with support $[m, M]$.

Let $\mathcal{D}$ be the set of all such functions D and let $\mathcal{D}^c$ be the set of all D^c with $D \in \mathcal{D}$. For each $G \in \mathcal{D}^c$ we can use G as a trading algorithm in the two following ways:

- Use G^c as an (ordinary) deterministic trading algorithm.
- Use G as a randomized algorithm, viewed as a probability distribution over threshold algorithms. Specifically, $G(x)$ is the cumulative probability of choosing one of the algorithms, RPP(s), $s \in [m, x]$.

It is straightforward to show that these two algorithms are equivalent in the sense that when played against a particular distribution function, they exhibit the same expected returns. Let $G \in \mathcal{D}^c$. For any $F \in \mathcal{F}$, when G is used as a randomized algorithm, its expected return with respect to F is

$$R_1(F, G) = \int_m^M \left(xF^c(x) + mF(x)\right) dG(x).$$

On the other hand, as a deterministic algorithm (G^c) its expected return is

$$R_2(F, G) = \int_m^M \left(\int_m^x u\, dG(u) + m\,[1 - G(x)]\right) dF(x).$$

We note that since G is right-hand continuous and F is left-hand continuous, and since both G and F are of bounded variation, then both R_1 and R_2 exist.

LEMMA 12. *For each $F \in \mathcal{F}$ and $G \in \mathcal{D}^c$, $R_1(F, G) = R_2(F, G)$.*

PROOF.

$$\begin{aligned}
R_1(G, F) &= \int_m^M x(1 - F(x))\, dG(x) + m \int_m^M F(x)\, dG(x), \\
R_2(G, F) &= \int_m^M \left(\int_m^x u\, dG(u)\right) dF(x) + m \int_m^M dF(x) - m \int_m^M G(x)\, dF(x) \\
&= \int_m^M \left(\int_m^x u\, dG(u)\right) dF(x) + m - m\left(1 - \int_m^M F(x)\, dG(x)\right) \\
&= \int_m^M \left(\int_m^x u\, dG(u)\right) dF(x) + m \int_m^M F(x)\, dG(x).
\end{aligned}$$

Hence, it is sufficient to prove that

$$\int_m^M \left(\int_m^x u\, dG(u)\right) dF(x) = \int_m^M x(1 - F(x))\, dG(x). \tag{40}$$

For any positive integer n, let π_n be the following subdivision of the interval $[m, M]$:

$$m = x_0 < x_1 < \cdots < x_n = M,$$

where $x_i = m + i(M - m)/n$, $i = 0, 1, \ldots, n$. With respect to π_n we write particular Stieltjes sums for both integrals of (40). These sums turn out to be identical. For each

$1 \le i \le n$, set $g_i = G(x_i) - G(x_{i-1})$. The sum

$$S_n = \sum_{i=1}^{n} \left(\sum_{j=1}^{i-1} x_j g_j \right) [F(x_i) - F(x_{i-1})]$$

is a Stieltjes sum of the left-hand integral of (40), whereas the sum

$$T_n = \sum_{i=1}^{n} [x_i - x_i F(x_i)]\, g_i$$

is a Stieltjes sum of the right-hand integral of (40).

$$\begin{aligned} S_n \;=\; & [F(x_2) - F(x_1)]\, x_1 g_1 \\ & + [F(x_3) - F(x_2)]\, (x_1 g_1 + x_2 g_2) \\ & + [F(x_4) - F(x_3)]\, (x_1 g_1 + x_2 g_2 + x_3 g_3) \\ & + \\ & \vdots \\ & + [F(x_{n-1}) - F(x_{n-2})]\, (x_1 g_1 + x_2 g_2 + \cdots + x_{n-2} g_{n-2}) \\ & + [F(x_n) - F(x_{n-1})]\, (x_1 g_1 + x_2 g_2 + \cdots + x_{n-2} g_{n-2} + x_{n-1} g_{n-1}) \\ =\; & \sum_{i=1}^{n-1} x_i g_i - \sum_{i=1}^{n-1} x_i g_i F(x_i) \\ =\; & \sum_{i=1}^{n-1} (x_i - x_i F(x_i))\, g_i . \end{aligned}$$

However, $F(x_n) = F(M) = 1$, so $x_n g_n = F(x_n) x_n g_n$, and $S_n = T_n$.

By definition, as $n \to \infty$, S_n and T_n approach $R_2(F, G)$ and $R_1(F, G)$, respectively. □

One conclusion of Lemma 12 is that the online player may only consider mixtures of reservation price policies as candidates for optimal algorithms against F. The next lemma states that the online player may restrict his attention only to (deterministic) reservation price policies.

LEMMA 13. *Let $G \in \mathcal{D}^c$ be any distribution reservation price policy* RPP(s). *Then, for every $\varepsilon > 0$, and for every $F \in \mathcal{F}$, there exists a number $s^* \in [m, M]$ such that $P_{\mathrm{RPP}(s^*)}(F) \ge P_G(F) - \varepsilon$.*

PROOF. Let $\varepsilon > 0$ be given.

$$P_G(F) = \int_m^M P_{\mathrm{RPP}(x)}(F)\, dG(x) = \int_m^M \left(x F^c(x) + m F(x) \right) dG(x).$$

Let s^* be a number in $[m, M]$ such that

$$P_{\mathrm{RPP}(s^*)}(F) = \sup_x P_{\mathrm{RPP}(x)}(F) - \varepsilon.$$

Therefore, for all $x \in [m, M]$, $P_{\text{RPP}(s^*)}(F)+\varepsilon \geq P_{\text{RPP}(x)}(F)$. For any distribution function, G,

$$\begin{aligned} P_{\text{RPP}(s^*)}(F) + \varepsilon &= \left(P_{\text{RPP}(s^*)}(F) + \varepsilon\right) \int_m^M 1 \cdot dG(x) \\ &= \int_m^M \left(P_{\text{RPP}(s^*)}(F) + \varepsilon\right) dG(x) \\ &\geq \int_m^M P_{\text{RPP}(x)}(F)\, dG(x). \end{aligned}$$

Hence,

$$P_{\text{RPP}(s^*)}(F) + \varepsilon \geq \sup_{G \in \mathcal{D}^c} \int_m^M P_{\text{RPP}(x)}(F)\, dG(x).$$

Let $I_{s^*} \in \mathcal{F}$ be the cumulative distribution function

$$I_{s^*}(x) = \begin{cases} 0, & \text{if} \quad m \leq x < s^*, \\ 1, & \text{if} \quad s^* \leq x \leq M; \end{cases}$$

$$\sup_{G \in \mathcal{D}^c} \int_m^M P_{\text{RPP}(x)}(F)\, dG(x) \geq \int_m^M P_{\text{RPP}(x)}(F)\, dI_{s^*}(x) = P_{\text{RPP}(s^*)}(F).$$

Therefore,

$$\sup_{G \in \mathcal{D}^c} \int_m^M P_{\text{RPP}(x)}(F)\, dG(x) - \varepsilon \leq P_{\text{RPP}(s^*)}(F) \leq \sup_{G \in \mathcal{D}^c} \int_m^M P_{\text{RPP}(x)}(F)\, dG(x).$$

Thus, the proof is complete. □

REMARK 10. In the case where $\max_x P_{\text{RPP}(x)}(F)$ exists we obtain the stronger result

$$P_{\text{RPP}(s^*)}(F) = \max_{G \in \mathcal{D}^c} \int_m^M P_{\text{RPP}(x)}(F)\, dG(x),$$

where s^* is a number in $[m, M]$ such that

$$P_{\text{RPP}(s^*)}(F) = \max_{m \leq x \leq M} P_{\text{RPP}(x)}(F).$$

For convenience, we assume for the rest of the section that $\max_x P_{\text{RPP}(x)}(F)$ exists. Thus, the results we obtain through consideration of the optimal performance of deterministic threshold algorithms are limits of the actual bounds.

PROOF OF THEOREM 7. By Lemma 13, it is sufficient to consider only reservation price policies as candidates for the optimal online algorithm against F. Let x^* be the value of x at which the quantity $mF(x) + xF^c(x)$ is maximized. Thus, given F, $\text{RPP}(x^*)$ is an optimal online algorithm against F.

We want to calculate the quantity

$$\max_F \frac{P_{\text{OPT}}(F)}{P_{\text{RPP}(x^*)}(F)} = \max_F \frac{\mathbf{E}[\mu]}{mF(x^*) + x^* \cdot F^c(x^*)}. \tag{41}$$

Consider the problem of maximizing $\mathbf{E}[\mu]$ subject to the constraint $mF(x^*) + x^* \cdot F^c(x^*) \le z$, where z is a parameter to be determined later. The constraint is equivalent to the following condition: for all $x \in [m, M]$,

$$mF(x) + xF^c(x) \le z, \tag{42}$$

or, equivalently,

$$F^c(x) \le \frac{z-m}{x-m}. \tag{43}$$

Notice that $F^c(M) = 0$ (and $F(M) = 1$). Hence, by substituting M for x in (42) we learn that $z \ge m$.

Then

$$\begin{aligned}
\mathbf{E}[\mu] &= \int_m^M x\, dF(x) \\
&= M \cdot F(M) - mF(m) - \int_m^M F(x)\, dx \\
&= M - \int_m^M F(x)\, dx \\
&= m + \int_m^M (1 - F(x))\, dx \\
&= m + \int_m^M F^c\, dx \\
&= m + \int_m^z F^c\, dx + \int_z^M F^c\, dx \\
&\le m + \int_m^z 1 \cdot dx + \int_z^M F^c\, dx \\
&= z + \int_z^M F^c\, dx \\
&\le z + \int_z^M \frac{z-m}{x-m} dx \\
&= z + (z-m) \ln \frac{M-m}{z-m}.
\end{aligned}$$

By the above derivation, it is evident that $\mathbf{E}[\mu]$ can be maximized, while still satisfying the constraint (43), by taking

$$F^c(x) = \begin{cases} 0, & \text{if} \quad x = M, \\ \dfrac{z-m}{x-m}, & \text{if} \quad z < x < M, \\ 1, & \text{if} \quad m \le x \le z. \end{cases}$$

For such $F^c(x)$,

$$\mathbf{E}[\mu] = z + (z - m)\ln\frac{M-m}{z-m}.$$

Set

$$g(z) \stackrel{\text{def}}{=} \frac{z + (z-m)\ln((M-m)/(z-m))}{z}.$$

Hence, by the choice of $F^c(x)$ and by the definition of z,

$$g(z) \le \frac{\mathbf{E}[\mu]}{mF(x^*) + x^* \cdot F^c(x^*)},$$

so we can calculate (41) by maximizing $g(z)$. Using elementary calculus we show that $g(z)$ is maximized at $z^* = m\ln((M-m)/(z^*-m))$. By differentiation,

$$g'(z) = \frac{-z + m\ln((M-m)/(z-m))}{z^2},$$

and

$$g''(z) = -\frac{1}{z(z-m)} + 2z^{-2}\left(g(z) - \ln\frac{M-m}{z-m}\right).$$

Therefore, $g'(z^*) = 0$, and it can be verified that $g''(z^*) = -1/z^*(z^*-m)$. Hence, z^* maximizes g.

Let r denote $\max_F \min_{\text{RPP}(x)}(F, m, M)$. Then r is determined by evaluating $g(z^*)$. A brief calculation shows that $r = \ln((M-m)/(z^*-m)) = \ln((M-m)/(mr-m))$ ($z^* = mr$ follows from the identity $z^* = m\ln((M-m)/(z^*-m))$. The proof is completed since r is identical to $r_\infty(m, M)$ (see (29)). □

Notice that the proof of Theorem 7 is constructive; it explicitly yields the probability distribution, F, that maximizes the return ratio; namely, for $mr \le x < M$, $F^c = (rm-m)/(x-m)$ and F is discontinuous at M, where the mass $(rm-m)/(M-m)$ is concentrated. It turns out that with respect to such F, all reservation price policies yield the maximum possible expected return, mr.

As we defined this trading game, the online player chooses his best strategy before the start of the game and is not allowed to change it thereafter. However, as the exchange rate varies, the online player's estimate of μ also varies. Assume that if the maximum rate observed so far is s, then the conditional distribution of μ given the history of E is obtained simply by excluding all values of μ less than s, and normalizing the relative probabilities of values greater than or equal to s. That is, this conditional distribution has the cumulative distribution function, F_s, given by

$$F_s(x) \stackrel{\text{def}}{=} \begin{cases} \dfrac{F(x) - F(s)}{1 - F(s)}, & x \ge s, \\ 0, & x < s. \end{cases} \tag{44}$$

It is not hard to see that if the online player chose the optimal threshold algorithm $\text{RPP}(x^*)$ before the start of the game, then the optimal threshold, x^*, remains fixed throughout the

game independent of the ever-changing estimate of μ. To see this, we write the expected return using some threshold x given that the maximum rate observed so far is $s < M$. Clearly, this expected return is

$$mF_s(x) + xF_s^c(x) = \frac{mF(x) + xF^c(x)}{1 - F(s)} - \frac{mF(s)}{1 - F(s)}. \tag{45}$$

It is evident that the same x maximizes (45) and $P_{\mathrm{RPP}(x)}(F)$ so the threshold remains fixed.

REMARK 11. We note that it is possible to remove the assumption that the exchange rate function is of a particular nature, except, of course, the requirement that the function reaches the (random) global maximum μ. Simply, if the adversary chooses any other kind of exchange rate function (e.g., with discontinuities over $[0, \mu]$), the expected return of the online player may only increase if he uses a reservation price policy. The optimal offline (expected) return remains the same. On the other hand, the online player cannot hope for higher returns since the adversary (knowing the algorithm chosen by the online player) may choose the function E_μ against which it is shown that reservation price policies can attain optimal returns.

7. Two-Way Trading. In this section we apply some of our results for one-way trading to the *two-way trading* problem where the player is allowed to convert the money back and forth between the two currencies. Two-way trading is a special case of the more general *portfolio selection* problem. In this problem a trader reallocates online his wealth between a number N of assets (or other investment instruments) in order to take advantage of the relative fluctuations between the various assets. The special case of $N = 2$ where one of the assets is cash corresponds to the two-way trading problem.

There is a considerable body of work related to two-way trading and portfolio selection. From the perspective of competitive analysis, there are a number of results which are somewhat related to the results presented in this section. For example, Raghavan [27] and Chou et al. [9], [10] study an online two-way trading problem against statistical adversaries, which must produce exchange rate sequences that conform to some statistical constraints. Cover and Ordentlich [13], [25], Helmbold et al. [18], and Blum and Kalai [6] study the general portfolio selection and give algorithms which are competitive relative to a constrained optimal offline algorithm. In particular, they compare their online algorithm to the best (offline) *constant rebalanced* algorithm that must keep a fixed proportion invested in each of the N assets. In this section we study the two-way trading problem with respect to the omnipotent optimal offline algorithm (still however we keep the constraint that prices are drawn from a bounded support).

We first state the problem more formally and then give preliminary lower and upper bounds on the competitive ratio of online algorithms for this problem. Both the lower and upper bounds are obtained simply by a decomposition of the exchange rate sequence into monotone increasing/decreasing segments on which we can apply our one-way trading results.

The online player starts with some D_0 dollars (here again, without loss of generality $D_0 = 1$) and converts back and forth between dollars and yen in accordance with

a sequence of exchange rates $\sigma = p_1, p_2, \ldots$ which is revealed online. These rates (yen/dollar) must remain in the interval $[m, M]$, but otherwise may rise or fall arbitrarily. When the game ends, the money is all converted to one of the currencies, say, dollars, at the present rate. Note that the ratio of the amount accumulated by the optimal offline algorithm on this sequence of exchange rates to the amount accumulated by the online player is the same, no matter which currency they convert to when the game ends. As in the one-way trading problem, the online player's goal is to minimize the competitive ratio.

If the number of local maxima and minima in the exchange rate sequence is unbounded, there is no strategy with a finite competitive ratio (see the lower bound argument in Section 7.2). Assume there are k such extrema. We show a lower bound of $r_\infty^{k/2}$ and an upper bound of r_∞^k where $r_\infty = r_\infty(m, M)$ is the optimal competitive ratio for a one-way game with an unbounded number of days (see Section 4.3).

7.1. *Upper Bound.* Assume that k is even. The sequence σ of exchange rates consists of $k/2$ upward runs and $k/2$ downward runs. The optimal algorithm for the offline player is to convert all his dollars to yen at the end of each upward run, and all his yen to dollars at the end of each downward run. We describe how the online player can achieve a competitive ratio of r_∞^k. Suppose the first upward run consists of $p_1 \le p_2 \le \cdots \le p_i$, with $p_{i+1} < p_i$. During the first i days, the online player plays according to our optimal one-way algorithm (with competitive ratio r_∞). By the end of the ith day he acquires D_i dollars and Y_i yen, where $mD_i + Y_i = p_i/r_\infty$. On day $i+1$ he converts all his dollars to yen. Since $p_{i+1} \ge m$, he then has at least p_i/r_∞ yen at the beginning of the first downward run. He then proceeds similarly during the downward run beginning at day $i+1$, converting yen to dollars during the decreasing run and exchanging any remaining yen on the first day of the next increasing run. Thus, two transactions occur on the first day of any decreasing run: first, the exchange of all dollars to yen, and second, the first transaction of the one-way algorithm on the downward run (conversion of yen to dollars). These conceptually distinct transactions can, of course, be combined into a single transaction. Similarly, on the first day of any subsequent increasing run, these two transactions may occur. In each run the ratio between the offline player's wealth and the online player's wealth increases by at most the factor r_∞, and thus the online player achieves the competitive ratio r_∞^k.

Notice that the player does not need to know k. In addition, some improvements may be employed. For example, if the player has a bound, l, on the maximum length of the upward/downward runs, then he can use the competitive ratio $r_l(m, M)$ in each one-way game (see Section 4.1), instead of $r_\infty(m, M)$. For most real-life price sequences, such a bound l can be identified and is rather small. Moreover, further improvement in performance may be obtained by using the improved algorithm of Section 4.2 in every one-way game.

7.2. *Lower Bound.* Assume that the online player knows only m and M. For any n, it is possible for an adversary to force a competitive ratio of $(r_n)^{k/2}$ against any online strategy.

Our argument here follows from the lower bound proof of the one-way case. Suppose each player starts with one dollar. Then, regardless of what strategy the online player

is employing, the adversary can construct a sequence of at most $n + 1$ exchange rates consisting of an upward run (of length n) followed by an immediate drop of the exchange rate to m, such that, at the end of the sequence, his total holdings in yen and dollars (evaluated at the exchange rate m) will exceed that of the online player by at least the factor r_n. Moreover, when the exchange rate drops to m, the adversary will convert all his yen to dollars, and the online player can do no better than to follow suit, since he will never have a more favorable conversion rate. Thus, in one upward and one downward run, the ratio of adversary currency to online currency can be made to increase by a factor of r_n. This yields a factor $(r_n)^{k/2}$ for the entire game (at most $k(n + 1)/2$ days). As n increases, this lower bound approaches $(r_\infty)^{k/2}$.

We point out that the fact that the above lower and upper bounds are exponential in k is somewhat misleading. It may appear worse than it really is. The reason is that optimal offline returns on sequences with k local minima and maxima are exponentially large so that these competitive ratios do not exclude the possibility of very large (exponential) returns for the online player.

8. Concluding Remarks. In our example of one-way trading, a striking feature of the problem is the conceptual simplicity of the optimal strategy. To attain a certain competitive ratio, the online player simply defends himself against the threat of dropping the exchange rate permanently to the minimum possible rate. It would be interesting to identify other problems in which a threat-based strategy is optimal. We note that we have identified another interesting problem that exhibits this kind of solution. Consider the following problem called *trading on option*:

A money trader starts with D_0 dollars and receives a finite sequence of option offers. Each option offer consists of a pair (e, p), where e is an *exchange rate* and $p > 0$ is an *option price*. For any $s \geq 0$ this offer enables the trader to pay ps dollars for the right to later trade s dollars for yen at the exchange rate e. The trader knows in advance that, even without purchasing options, he will always be able to exchange dollars for yen at the rate m, and that the maximum exchange rate he will ever be offered is M. As each offer (e, p) is presented, the trader chooses a corresponding value s and his stock of dollars is depleted by ps. At the end of the sequence of offers the trader converts his remaining dollars to yen by exercising some of the options he has purchased and by exchanging any remaining dollars at the rate m. The optimal offline strategy is clear; the offline player will choose that option (e, p) for which $e/(p + 1)$ is greatest and (provided that $e/(p + 1) > m$) will spend $pD_0/p + 1$ dollars for the right to exchange his remaining $D_0/(p + 1)$ dollars for yen at the exchange rate e. Under this optimal strategy the offline player receives $eD_0/(p + 1)$ yen. Intuitively, if the trader can achieve a competitive ratio of r for this problem, then he can do so using a threat-based strategy of the following form: whenever an offer (e, p) is presented, choose the minimum value of s that will ensure that the trader can obtain $(1/r)(eD_0/(p + 1))$ even under the threat that no further option offers will be made. (Full analysis of this problem will soon be published in a separate work.)

This paper leaves a number of questions open for future research. One intriguing question is that of the true lower/upper bounds on the competitive ratio of the two-way game. We suspect that a good starting point to investigate this question is to im-

prove the upper bound which is conjectured to be suboptimal. An intuitive reason for this conjecture is the following. Notice that on any upward (downward) run the online player can take advantage of the knowledge that in order to force a ratio r on the following downward (upward) run, the adversary must begin that run with a sufficiently small (large) exchange rate (i.e., rm in the case where the following run is an upward run).

It would be of interest to examine the robustness of our results under some real-life considerations such as transaction costs and errors in the choice of the parameters (m, M, n, and/or Φ).

An important issue that requires further study is the sensitivity and the dependence of the trading strategies on the constraining parameters. It is clear that a greedy choice of these parameters may result in an unfortunate outcome. However, an ability to estimate this dependence quantitatively allows for some degree of risk management. For example, consider the case in which the online player underestimates the upper bound on possible exchange rates. Say, $\overline{M} = cM$ is the true upper bound where $c > 1$. Clearly, in the worst possible rate sequence $p_{n-1} = M$ and $p_n = \overline{M}$. Therefore, our trading strategy will spend all the available dollars at the second last day and will gain at least $M/r_n(m, M)$ yen. On the same sequence the optimal offline player acquires $\overline{M}$ yen and therefore the attainable competitive ratio using the incorrect upper bound may be almost as poor as $(\overline{M}/M)R(M, n) = c \cdot r_n(m, M)$.

Acknowledgments. We warmly thank Allan Borodin, Brenda Brown, Faith Fich, Vassos Hadzilacos, Alex Lee, Charlie Rackoff, Oded Schramm, and Hisao Tamaki for many useful discussions and comments. We are also grateful to the anonymous referees for their comments.

References

[1] M. Ajtai, N. Megiddo, and O. Waarts. Improved algorithms and analysis for secretary problems and generalizations. In *Proceedings of the* 36*th Annual Symposium on Foundations of Computer Science*, pages 473–482, 1995.

[2] S. al-Binali. The competitive analysis of risk raking with application to online trading. In *Prodeedings of the* 38*th Annual Symposium on Foundations of Computer Science*, pages 336–344, 1997.

[3] R.J. Aumann. Mixed and behavior strategies in infinite extensive games. In M. Dresher, L.S. Shapley, and A.W. Tucker, editors, *Advances in Game Theory*, volume 1, pages 627–650. Princeton University Press, Princeton, New Jersey, 1964.

[4] R.E. Bellman and S.E. Dreyfus. *Applied Dynamic Programming*. Cambridge University Press, Princeton, New Jersey, 1962.

[5] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Widgerson. On the power of randomization in on-line algorithms. In *Proceedings of the* 22*nd Symposium on Theory of Algorithms*, pages 379–386, 1990.

[6] A. Blum and A. Kalai. Universal portfolios with and without transaction costs. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 309–313, 1997.

[7] A. Borodin, N. Linial, and M. Saks. An optimal online algorithm for metrical task systems. In *Proceedings of the* 19*th Annual ACM Symposium on Theory of Computing*, pages 373–382, 1987. For the journal version see [8].

[8] A. Borodin, N. Linial, and M. Saks. An optimal online algorithm for metrical task systems. *Journal of the ACM*, 39:745–763, 1992. For the conference version see [7].

[9] A. Chou, J. Cooperstock, R. El-Yaniv, M. Klugerman, and T. Leighton. The statistical adversary allows optimal money-making trading strategies. In *Proceedings of the* 6*th Annual ACM–SIAM Symposium on Discrete Algorithms*, 1995.

[10] A. Chou, A. Shrivastava, and R. Sidney. On the power of magnitude. Unpublished manuscript, May 1995.

[11] Y.S. Chow, H. Robbins, and D. Siegmund. *The Theory of Optimal Stopping*. Dover, New York, 1971.

[12] A. Condon. *Computational Models of Games*. The MIT Press, Cambridge, Massachusetts, 1988.

[13] T.M. Cover and E. Ordentlich. Universal portfolios with side information. *IEEE Transactions on Information Theory*, 42(2), March 1996.

[14] A. Fiat, R.M. Karp, M. Luby, L.A. McGeoch, D.D. Sleator, and N.E. Young. Competitive Paging Algorithms. Technical Report CMU-CS-88-196, Carnegie Mellon University, Pittsburg, Pennsylvania, 1988. For the journal version see [15].

[15] A. Fiat, R.M. Karp, M. Luby, L.A. McGeoch, D.D. Sleator, and N.E. Young. Competitive paging algorithms. *Journal of Algorithms*, 12:685–699, 1991. For the technical report version see [14].

[16] P.R. Freeman. The secretary problem and its extensions. *International Statistical Review*, 51:189–206, 1983.

[17] R.L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[18] D.P. Helmbold, R.E. Schapire, Y. Singer, and M.K. Warmuth. On-line portfolio selection using multiplicative updates. Unpublished manuscript, April 1996.

[19] D.S. Johnson. Near-Optimal Bin Packing Algorithms. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1973.

[20] D.S. Johnson, A. Demers, J.D. Ullman, M.R. Garey, and R.L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. 3:299–324, 1974.

[21] A.R. Karlin, L. Rudolph, and D.D. Sleator. Competitive snoopy caching. 3(1):70–119, 1988.

[22] L. Levin. Personal communication, July 1994.

[23] S.A. Lippman and J.J. McCall. The economics of job search: a survey. *Economic Inqury*, XIV:155–189, June 1976.

[24] S.A. Lippman and J.J. McCall. The economics of uncertainty: selected topics and probabilistic methods. In K.J. Arrow and M.D. Intriligator, editors, *Handbook of Mathematical Economics*, volume 1, chapter 6, pages 211–284. North-Holland, Amsterdam, 1981.

[25] E. Ordentlich and T.M. Cover. The cost of achieving the best porfolio in hindsight. *Mathematics of Operations Research*, 23(4):960–982, November 1998.

[26] C.H. Papadimitriou. Games against nature. In *Proceedings of the* 24*th Annual Symposium on Foundations of Computer Science*, pages 446–450, 1983.

[27] P. Raghavan. A statistical adversary for on-line algorithms. In L.A. McGeoch and D.D. Sleator, editors, *On-Line Algorithms*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 7, pages 79–83. American Mathematical Society, Providence, RI, 1992.

[28] D.B. Rosenfield and R.D. Shapiro. Optimal adaptive price search. *Journal of Economic Theory*, 25(1):1–20, 1981.

[29] D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28:202–208, 1985.

[30] A.C. Yao. New algorithms for bin packing. *Journal of the ACM*, 27:207–227, 1980.