On Capital Investment [*]

Yossi Azar [†] Yair Bartal [‡] Esteban Feuerstein [§]

Amos Fiat [†] Stefano Leonardi [¶] Adi Rosén [||]

Abstract

We deal with the problem of making capital investments in machines for manufacturing a product. Opportunities for investment occur over time, every such option consists of a capital cost for a new machine and a resulting productivity gain, i.e., a lower production cost for one unit of product. The goal is that of minimizing the total production costs and capital costs when future demand for the product being produced and investment opportunities are unknown. This can be viewed as a generalization of the ski-rental problem and related to the mortgage problem [3].

[*]A preliminary version of this paper appeared in the Proceedings of the 23rd International Colloquium on Automata, Languages and Programming (ICALP 96), Lecture Notes in Computer Science 1099, Springer-Verlag, 1996, pp. 429-441.

[†]Department of Computer Science, Tel Aviv University. Research supported in part by a grant from the Israel Academy of Sciences. e-mail: azar@math.tau.ac.il

[‡]International Computer Science Institute (ICSI), Berkeley. Research supported in part by the Rothchild Postdoctoral fellowship. email: yairb@icsi.berkeley.edu

[§]Depto. de Computacion, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires & Instituto de Ciencias, Universidad de General Sarmiento. Research supported in part by EEC project KIT-DYNDATA. email: efeuerst@dc.uba.ar

[¶]Dipartimento di Informatica Sistemistica, Università di Roma "La Sapienza". Research supported in part by EU Esprit Long Term Research project ALCOM IT under contract 20244, and by Italian Ministry of Scientific Research Project 40% "Algoritmi, Modelli di Calcolo e Strutture Informative". email: leon@dis.uniroma1.it

[||]Department of Computer Science, University of Toronto, Toronto, Canada. email: adiro@cs.toronto.edu

If all possible capital investments obey the rule that lower production costs require higher capital investments, then we present an algorithm with constant competitive ratio.

If new opportunities may be strictly superior to previous ones (in terms of both capital cost and production cost), then we give an algorithm which is $O(\min\{1+\log C, 1+\log\log P, 1+\log M\})$ competitive, where C is the ratio between the highest and the lowest capital costs, P is the ratio between the highest and the lowest production costs, and M is the number of investment opportunities. We also present a lower bound on the competitive ratio of any on-line algorithm for this case, which is $\Omega(\min\{\log C, \frac{\log\log P}{\log\log\log P}, \frac{\log M}{\log\log M}\})$. This shows that the competitive ratio of our algorithm is tight (up to constant factors) as a function of C, and not far from the best achievable as a function of P and M.

1 Introduction

We consider the problem of manufacturing costs versus capital investment on production resources. A factory uses machines for producing some product. The production of each unit of the product requires some fixed cost for using the machine (electricity, raw material, etc.). Over time opportunities for investment in new machines, that would replace the old ones, become available. Such opportunities could be the result of technological improvement, relocation to a cheaper market, or any other investment that would replace the facilities of the factory and would lead to lower production costs. We model all these opportunities as machines that can be bought, and then used to produce the product. The factory must decide if to invest in buying new machines to reduce production costs while neither future demand for the product nor future investment opportunities are known.

Many financial problems require to take decisions without having knowledge, or having only partial knowledge, of future opportunities. Competitive analysis of financial problems has received an increasing attention during recent years, for instance for currency exchange problems [2] or asset allocation [5].

The problem considered in this paper is a generalization of one of the basic on-line problems, the *ski-rental* problem due to L. Rudolph (see [4]), a model for the well known practical problem "rent or buy?". The ski-rental problem can be stated as follows: you don't know in advance how many times you will go skiing; renting a pair of skis costs $\$r$; to purchase your own pair costs $\$p$. When do you buy? It is not hard to see that the best deterministic competitive ratio is obtained if you buy when the total rental cost (thus far) is equal to the cost of buying your own pair. Another problem considered in this model in the past is the so-called mortgage problem [3], where a fluctuating mortgage rate and associated re-financing charges lead to the question "re-finance or not?".

While for the ski-rental problem the only possible capital expenditure is to purchase a pair of skis, and then the "production" costs drop to zero, in the capital investment problem there may be many future capital expenditure options and the resulting productivity gains are unknown. Unlike the mortgage problem, where the future demand is known (the entire debt — which is a known fixed value — must be served), and capital investments have a fixed cost (the cost of re-financing the mortgage), in the capital investment problem future demand is unknown and capital investments may have arbitrary costs.

We consider two models for our problem, and call the first one the *convex case*. Here, we assume that to get a lower production cost, one must spend more as capital expenditures. In this case we get a constant competitive ratio. This scenario is usually true in manufacturing: purchasing a better machine costs more. However, sometimes technological breakthroughs are achieved, after which both machine costs and production costs are reduced. This matches our second model, the *non-convex* case, which allows both capital and production costs to drop.

In contrast to the convex case, for the non-convex case we present a non-constant lower bound on the competitive ratio of any on-line deterministic algorithm for the problem. We show that no deterministic algorithm can achieve a competitive ratio better than $\Omega(\min\{\log C, \frac{\log\log P}{\log\log\log P}, \frac{\log M}{\log\log M}\})$, where C is the ratio between the highest and the lowest capital costs, P is the ratio between the highest and the lowest production costs, and M is the number of investment opportunities. We complement this lower bound with an algorithm for general capital investment scenarios which is $O(\min\{1 + \log C, 1 + \log\log P, 1 + \log M\})$ competitive.

2 The On-line Capital Investment Problem

Imagine a factory whose goal is to produce units of some commodity at low cost. From time to time, orders for units of the commodity arrive, and at times new machines become available in the market. Every such machine is characterized by its *production cost*, and by its *capital cost*. The production cost is the cost of producing one unit of commodity using this machine. The capital cost is the capital investment necessary to buy the machine. We assume that once a machine becomes available, then it is available forever. We also assume that one can produce an unlimited number of units with any machine. An algorithm for this problem has to decide what machines to buy and when to do so, so as to minimize the total cost (capital costs plus production costs).

More formally, an instance of the problem consists of a sequence of machines, and a sequence of orders of demand. Machine m_i is defined by the triplet (t_i, c_i, p_i), where t_i is a positive integer that indicates the discrete time at which the machine becomes available, c_i is its capital cost, and p_i is its production cost. Every order of demand is defined by its arrival time. Without loss of generality we may assume that the j'th order appears at discrete time $t_j = j$, where j is a positive integer. We also assume that any integer time t, the algorithm can buy any of the available machines (those with $t_i \leq t$), and use this machine for the production of units of commodity. When an order of demand is placed, say at time t, the on-line algorithm has to produce one unit of commodity immediately. (It can however buy a new machine m_i presented at time $t_i \leq t$ to produce the unit of commodity.)

We say that machine m_i *dominates* machine m_j if both the production cost and the capital cost of m_i are lower than those of m_j. We call an instance of the problem *convex* if no machine presented dominates another. I.e., an instance is convex if for any two machines i, j such that $p_i < p_j$ it holds that $c_i \geq c_j$. To distinguish between the two versions of the problem, the case in which convexity restrictions do not necessarily hold will be called the *non-convex* case.

We note that if all machines are available at the very beginning, then all machines that are dominated by others can be removed. Thus, whenever all machines are available in advance, we are left with the convex setting. The non-convex setting only makes sense if machines appear over time and it is possible that a better machine (in terms of both capital cost and production cost criteria) will appear later.

2.1 Performance Measures

We measure the performance of an on-line algorithm for this problem by its competitive ratio [6]. Let σ be a sequence of offers of machines and orders of demand for units of the commodity to be produced.

We denote by $\mathrm{ON}(\sigma)$ the cost of the on-line algorithm ON for the problem over the sequence σ, and with $\mathrm{OPT}(\sigma)$ the cost of an *optimal* off-line algorithm that knows the entire sequence σ in advance. We parameterize the sequences by the ratio between the cost of the most expensive and cheapest machines (denoted by C), by the ratio between the highest and the lowest production cost (denoted by P), and by the total number of machines presented during the sequence (denoted by M). Denote by $\Sigma(C, P, M)$ the set of sequences that obey the above restrictions.

The competitive ratio of an algorithm may be a function of the above parameters. An on-line algorithm ON is $\rho(C, P, M)$-competitive for a set $\Sigma(C, P, M)$ of sequences if

$$\sup_{\sigma \in \Sigma(C,P,M)} \frac{\mathrm{ON}(\sigma)}{\mathrm{OPT}(\sigma)} \leq \rho(C, P, M).$$

3 Upper Bound for the Convex Case

In this section we study the convex case in which a machine with a lower production cost cannot be cheaper than a machine with a higher production cost. We present an on-line algorithm for the convex case with competitive ratio $4 + 2\sqrt{2} \approx 6.83$.

3.1 The Algorithm

The algorithm is defined as follows: before producing the first unit the algorithm buys the machine m_i that minimizes $p_i + c_i$ amongst all machines available at the beginning of the sequence. It then produces the first unit of commodity. The initial cost $p_i + c_i$ is considered a *production* cost.

Let α and β be positive constants satisfying $2/\alpha \leq 1$ and $1/\alpha + 2\beta \leq 1$. In particular we choose $\alpha = 1 + \sqrt{2}$ and $\beta = 1/(2 + \sqrt{2})$.

Before producing any subsequent unit of commodity the algorithm considers buying a new machine. However, it is not always allowed to buy a new machine. When an amount of c is spent as capital cost to buy a machine, it is not allowed to buy another machine until the algorithm spends at least $\beta \cdot c$ on production.

When it is allowed to buy a machine, the algorithm buys the machine m_i that minimizes production cost p_i amongst all machines of capital cost at most α times the total production cost incurred since the beginning of the sequence. If no such machine is available, the algorithm does not buy a new machine.

3.2 Analysis

We prove that the competitive ratio of the above algorithm is $1 + \alpha + 1/\beta = 4 + 2\sqrt{2}$.

We use the following notation. Fix the sequence σ. Denote by $\mathrm{ON} = \mathrm{ON}^c + \mathrm{ON}^p$ the total cost of the algorithm that is equal to the sum of the total capital cost ON^c and the total production cost ON^p. Let p^t be the production cost incurred by the on-line algorithm to produce unit number t. Let ON^p_t be the production cost incurred by the algorithm to produce the first t units, i.e., $\mathrm{ON}^p_t = \sum_{i=1}^{t} p^i$. Let OPT_t be the optimal total (capital and production) cost to produce the first t units. We start by proving a bound on the total cost spent on purchasing machines, in terms of the total production cost incurred.

Lemma 3.1 *The total capital cost* ON^c *incurred by the on-line algorithm is at most* $(\alpha + 1/\beta)$ *its total production cost* ON^p*.*

Proof. The capital cost of the last machine bought is at most α times the total production cost. For every other machine, the production costs in the interval between the time this machine has been bought, and the time the next machine is bought, is at least β times the capital cost of the machine. These intervals do not overlap, and thus the total capital cost of all the machines except the last one sums to at most $1/\beta$ times the total production cost. ■

We now relate the production cost of the on-line algorithm to the total cost of the off-line algorithm.

Lemma 3.2 *At any time t the production cost ON_t^p of the on-line algorithm is at most the total cost OPT_t of the off-line algorithm.*

Proof. We prove the claim by induction on the number of units produced.

For $t = 1$ the claim holds since the on-line production cost of the first unit (defined as the sum of the capital and the production costs of the first machine bought) is the minimum possible expense to produce the first unit. Therefore $\mathrm{ON}_1^p \leq \mathrm{OPT}_1$.

Consider unit t for $t > 1$, and assume the claim holds for any unit $\hat{t} < t$. Let m be the machine used by the on-line algorithm to produce unit t. Let m' be the machine used by the optimal off-line solution to produce unit t, p' its production cost, and c' its capital cost.

If $p' \geq p^t$ then we have $\mathrm{ON}_t^p = \mathrm{ON}_{t-1}^p + p^t \leq \mathrm{OPT}_{t-1} + p' \leq \mathrm{OPT}_t$.

If $p' < p^t$ then the on-line algorithm did not buy machine m' before producing unit t. Let the capital cost of the last machine bought by the on-line algorithm (i.e. m) be $\bar{c}$, and assume it was bought just before unit $\bar{t}$ was produced. Since we consider the convex case we have that $p' < p^t = p^{\bar{t}}$ implies $c' \geq \bar{c}$.

As we assume that the on-line algorithm did not buy m' just before producing unit t, one of the following holds:

1. The capital cost of machine m' was too high, i.e., less than $\frac{1}{\alpha}c'$ was spent on production since the start of the sequence.

2. It was not allowed to buy any machine at this time: less than $\beta \cdot \bar{c}$ was spent on production since machine m was bought, and until unit number $t-1$ is produced.

We consider each of these cases:

1. We have that $\mathrm{ON}_t^p = \mathrm{ON}_{t-1}^p + p^t \leq 2 \cdot \mathrm{ON}_{t-1}^p \leq \frac{2}{\alpha}c' \leq \mathrm{OPT}_t$.

2. We have that

$$\mathrm{ON}^p_t = \mathrm{ON}^p_{\bar{t}-1} + \sum_{i=\bar{t}}^{t-1} p^i + p^t \le \mathrm{ON}^p_{\bar{t}-1} + 2\sum_{i=\bar{t}}^{t-1} p^i < \mathrm{ON}^p_{\bar{t}-1} + 2\beta \cdot \bar{c}\ .$$

   We now distinguish between two sub-cases, depending on whether machine m' is available before unit $\bar{t}$ is produced. The first sub-case is that machine m' becomes available only after unit $\bar{t}$ is produced. In this case we have

$$\mathrm{ON}^p_t < \mathrm{ON}^p_{\bar{t}-1} + 2\beta\cdot\bar{c} \le \mathrm{OPT}_{\bar{t}-1} + 2\beta\cdot\bar{c} \le \mathrm{OPT}_{\bar{t}-1} + \bar{c} \le \mathrm{OPT}_{\bar{t}-1} + c' \le \mathrm{OPT}_t\ .$$

   The second sub-case is when machine m' is available before unit number $\bar{t}$ is produced. We have that its capital cost, c', is higher than $\alpha \cdot \mathrm{ON}^p_{\bar{t}-1}$, otherwise the on-line algorithm would have bought this (or a better) machine at time $\bar{t}$, which contradicts $p^{\bar{t}} > p'$. Therefore we have

$$\mathrm{ON}^p_t < \mathrm{ON}^p_{\bar{t}-1} + 2\beta\cdot\bar{c} \le \mathrm{ON}^p_{\bar{t}-1} + 2\beta\cdot c' \le (1/\alpha)c' + 2\beta\cdot c' = (1/\alpha + 2\beta)c' \le \mathrm{OPT}_t\ .$$

■

Combining Lemma 3.1 and Lemma 3.2 we get the following theorem.

Theorem 3.1 *The algorithm presented above for the convex case of the on-line capital investment problem achieves a competitive ratio of* $1 + \alpha + 1/\beta$.

4 Lower Bound for the Non-Convex Case

In contrast to the constant upper bound proved in the previous section, in this section we prove a lower bound on the competitive ratio of any deterministic on-line algorithm for the non-convex case. The lower bound that we prove is $\Omega(\min\{\log C, \frac{\log\log P}{\log\log\log P}, \frac{\log M}{\log\log M}\})$, where C is the ratio between the highest and the lowest capital costs, P is the ratio between the highest and the lowest production costs, and M is the number of presented machines.

We now describe the instance of the problem on which the lower bound is achieved. Let C be some large power of 2, at least $2^5 = 32$. The capital costs of all the machines in the instance are powers of 2 between 2 and C, and their production costs will be of the form $1/\log^k C$, for some positive integer k. We assign a *level* between 1 and $\log C$ to each machine; machines of level i have capital cost $c_i = 2^i$.

We divide the time into phases for any level between 1 and $\log C$. If a *phase of level* i ends at time t then a new phase of level i starts at time $t+1$. We assume that at time 0 phases of all levels end. Then at time 1, a phase of each level starts.

A phase of level i ends when one of the following occurs:

1. The on-line algorithm buys a machine of level i.

2. The on-line algorithm has reached a global cost (production and capital) in the phase greater or equal to $\frac{i}{2}c_i$.

3. A phase of level higher than i ends.

If more than one phase ends at the same time in Case 1 or in Case 2, we say that the phase of highest level among them ends in Case 1 or 2 and consider the other, lower-level phases, as ending in Case 3.

The sequence is produced by the adversary as follows: at every integral time unit $t \geq 1$ there is a request for the production of one unit of commodity. The presentation of machines follows the rule that at the beginning of a phase of level i, a machine of level i is presented. To define the production costs of the machines the following rule is applied: Let $n_k(i) = \frac{i!}{k!}$ for $i = 1, \ldots, \log C$, $k = 1, \ldots, \log C$. When a phase of level i with an associated machine of production cost p ends in Case 1 or Case 2 (and thus new phases of levels $j \leq i$ start), a set of i machines are presented, one for each level $j = 1, \ldots, i$. The production cost of the appropriate machine of level j is defined to be

$$p_j = \frac{p}{(\log C)^{\sum_{k=1}^{j} n_k(j)}}.$$

Recall that if a phase ends in Case 1 at time t since the on-line algorithm buys a machine at time t, then at time $t+1$ a new machine of the same level is presented, and hence a new phase of that level starts.

At the beginning we act as if at time 0 a phase of level $i = \log C$, with a machine having production cost $p = \frac{1}{\log C}$, ended, so that phases of all levels start at time 1 when a first set of machines of all levels are presented, with capital and production costs as defined above.

The sequence will be over with the end of the phase of level $\log C$ associated with the machine of capital cost C presented at time 1. The sequence is built so that there is only one machine of capital cost C presented in the whole sequence, and that machine's production cost is at most $1/\log C$ the production cost of any other machine presented in the sequence.

We define a relation of inclusion between phases. A phase of level i contains all the phases of level $j < i$ that start simultaneously with that phase, or start during that phase. Note that no phase of level $j > i$ starts during a phase of level i. We call a phase *active* if it is not ended yet. At every point in time one phase is active at every level.

We call a phase that ends in Case 1 or Case 2 a *complete phase* and a phase that ends in Case 3 an *incomplete phase.* If a phase of level i is complete then the $i-1$ phases at lower levels that end as a consequence of the end of this level i phase are incomplete.

Lemma 4.1 *At most i machines of level $i-1$ are presented during a phase of level i for $i \geq 2$.*

Proof. For $i \geq 2$, a new machine of level $i-1$ is presented during the phase of level i only when the on-line algorithm buys the previous machine of level $i-1$, or when its cost during the phase of level $i-1$ reaches $\frac{i-1}{2}c_{i-1}$. In any case, the on-line algorithm's cost for the phase of level $i-1$ is at least c_{i-1}. Hence, the maximum number x of phases of level $i-1$ is restricted to be $xc_{i-1} \leq \frac{i}{2}c_i$, which implies $x \leq i$. ■

The production costs defined above were chosen so as to obey the property stated in the following lemma.

Lemma 4.2 *The machine of level i presented during a phase P of level i has production cost less than or equal to $1/\log C$ times the production cost of:*

1. *any machine presented during a phase that ends before the starting of phase P;*

2. any machine of level $k < i$ presented during phase P.

Proof. We first prove the second part of the claim, by induction on i. We have to prove that every machine of level $k < i$ presented during the phase of level i has production cost at least $p_i \cdot \log C$, where p_i is the production cost of the machine of level i in question.

For the basis of the induction, we note that for $i = 1$ the claim is trivially true, as there are no machines of levels $k < i$. We now prove the claim for $i > 1$, assuming that it holds for $i - 1$. Consider the machines of level $i - 1$ presented during the phase of level i. First note that by Lemma 4.1 there are at most i such machines. The first machine is presented together with the the machine of level i. Let p be the production cost of the machine associated with the phases that ends just before these machines are presented. Then, the production cost of the machine of level $i - 1$ is

$$p_{i-1} = \frac{p}{(\log C)^{\sum_{k=1}^{i-1} n_k(i-1)}} \ .$$

Now note that the next machine of level $i-1$ is presented when this phase of level $i - 1$ ends (as no phase of higher level can end while the phase of level i is active). Therefore, the production cost of the ℓ'th machine of level $i - 1$ presented during the phase of level i is

$$p_{i-1} = \frac{p}{(\log C)^{\ell(\sum_{k=1}^{i-1} n_k(i-1))}} \ .$$

That is, the production costs of the machines of level $i-1$ are decreasing, with the last machine presented during the phase of level i being of production cost

$$p'_{i-1} \geq \frac{p}{(\log C)^{i(\sum_{k=1}^{i-1} n_k(i-1))}} = \frac{p}{(\log C)^{(\sum_{k=1}^{i} n_k(i))-1}} = p_i \ \log C \ .$$

We conclude that all machines of level $i - 1$ presented during the phase of level i have cost at least $\log C$ times the cost of the machine of level i in question. Using the induction hypothesis, we also know that any machine presented *during* any of the phases of level $i - 1$ has production cost at least $\log C$ times the production cost of the machine of level $i - 1$ and hence at least $\log C$ times the production cost of the machine of level i. This concludes the proof of the second part.

We now prove the first part of the claim. We prove it by induction on time (i.e., unit of commodity produced). That is, we claim that any phase that starts at time t is associated with a machine of production cost less or equal to $1/\log C$ the production cost of any machine presented during a phase that ended before time t. For time $t = 1$ (when the sequence starts) the claim is obviously true, as there is no phase that ended previously. Now consider a phase of level i that starts at time $t > 1$. This phase starts due to the end of a phase of some level $j \geq i$; let the production cost of the machine associated with this phase be p. Then the production cost of the machine of level i presented at t is $p_i = \frac{p}{(\log C)^{\sum_{k=1}^{i} n_k(i)}}$. Now note that any phase that ended before time t is either a phase that starts during the phase of level j that just ended, or a phase that ended before the phase of level j started. For the phases of the first type, their production cost is higher than p, by the second part of the lemma. Thus, the claim holds with respect to those. As to the machines of the latter type, their production cost is higher than p, by the induction hypothesis (as the phase of level j started before time t). Thus the claim holds with respect to them as well. ■

Consider a phase of level i. Let z_i be the cost incurred by the on-line algorithm *for production* during the phase. Let y_i be the cost of the on-line algorithm incurred during the phase for capital costs to buy machines of levels j, $j \leq i$. Let $O_i = z_i + y_i$.

Definition. Consider a phase of level i. The *restricted optimal cost* of this phase is the optimal cost to produce all the units of commodity required to be produced during the phase under the restriction that one can buy only machines of level not higher than i, and under the assumption that the algorithm already possesses one machine when the phase starts.

For a given phase of level i we denote the restricted optimal cost of the phase by A_i. When considering a sequence of phases of level i, we denote the respective restricted optimal costs by a superscript, e.g., A_i^j.

By giving upper bounds on the value of A_i we will give an upper bound on the cost of the adversary to serve this portion of the sequence. Obviously, we are interested in an upper bound on $A_{\log C}$. For $A_{\log C}$, the restriction above is empty, and by adding the cost of the cheapest machine available when the sequence starts, we get an upper bound on the cost of the real adversary for the whole sequence.

First, we state two upper bounds on A_i. The first upper bound is derived from the case where the optimal algorithm chooses to buy the machine of level i presented when the phase starts. The other one is derived from the case in which the optimal solution does not buy this machine.

Lemma 4.3 *For any phase of level i, $A_i \leq 2c_i$.*

Proof. We consider the possible scenario in which the optimal solution buys the machine of level i that is presented at the beginning of the phase as soon as it is presented. The main argument of the proof is that the on-line algorithm may use this machine only for the last unit of commodity produced in the phase, since by buying it the phase ends. Therefore, any on-line algorithm produces with production cost higher than that of the optimal solution, as all other machines available until the end of the phase have, by Lemma 4.2, higher production costs by at least a logarithmic factor.

The adversary can first buy the machine of level i, incurring a cost of $c_i = 2^i$, and then produce the rest of the demand using this machine. The production cost of the machine used is, with the possible exception of the last unit produced in the phase, at most $\frac{1}{\log C}$ times the production cost of the machine used by the online algorithm. The production of the last unit costs at most $\frac{1}{\log C}$ since this is an upper bound on the production cost of all machines. Then, the optimal solution incurs a production cost of at most $\frac{z_i}{\log C} + \frac{1}{\log C}$.

But $z_i < \frac{i}{2}c_i + \frac{1}{\log C} \leq \frac{\log C}{2}c_i + \frac{1}{\log C}$ since $i \leq \log C$ and the last unit of demand is produced by the on-line algorithm incurring a cost of at most $\frac{1}{\log C}$.

For $\log C \geq 4$ we have

$$\begin{aligned} A_i &\leq c_i + \frac{z_i}{\log C} + \frac{1}{\log C} \\ &< c_i + \frac{\frac{\log C}{2}c_i + (1/\log C)}{\log C} + \frac{1}{\log C} \\ &= c_i + \frac{c_i}{2} + (1/\log^2 C) + (1/\log C) \\ &\leq 2c_i \ . \end{aligned}$$

■

The second upper bound on A_i is derived from the case in which the adversary chooses not to buy the machine of level i. We can give an upper bound on the restricted optimal cost by summing up the costs of the lower level phases from which this phase is composed. A phase of level i (complete or incomplete) is partitioned into a sequence of phases of level $i-1$, whose number we indicate by s_i. The last one of those phases may be incomplete, while the first $s_i - 1$ are complete. Thus, we get the following lemma.

Lemma 4.4 *Consider a phase of level i. Let A_{i-1}^j, $j = 1, \ldots, s_i$ be the restricted optimal cost of the j-th sub-phase of level $i-1$. Then $A_i \leq \sum_{j=1}^{s_i} A_{i-1}^j$.*

Theorem 4.1 *If an algorithm for the non-convex on-line capital investment problem is ρ-competitive then $\rho = \Omega(\log C)$.*

Proof. We first observe that the (unique) phase of level $\log C$ indeed ends, since all production costs are positive and therefore the on-line production cost eventually reaches $\frac{\log C}{2} C$ (the phase can end earlier if the on-line algorithm buys the machine of level $\log C$).

We will now show that any on-line algorithm pays a global cost (over the sequence) of at least $\frac{1}{32} \log C$ times the cost of the adversary.

To prove that, we first prove a claim concerning a phase of any level i. We focus our attention on a specific phase of level i. The phase starts one unit of time after that the previous phase of level i ends. By definition, one machine for each level $j \leq i$ is presented at the beginning of the phase. Observe that during this phase the on-line algorithm does not buy any machine of level higher that i, since otherwise the phase immediately ends.

We prove the following claim:

- $O_i \geq \frac{i-1}{8} A_i$ for a complete phase;
- $O_i \geq \frac{i-1}{8} A_i - \frac{c_i}{2}$ for an incomplete phase.

We prove the claim for each of the three cases in which a phase ends. Recall that in Case 1 and Case 2 the phase is complete and the first part of the claim must be proved, while in Case 3 the phase is incomplete and the second part of the claim must be proved. We prove the claim by induction on i. The claim obviously holds for level $i = 1$. To prove the claim for a level $i + 1$, we assume it holds for levels $k \leq i$.

1. In Case 1, the phase ends when the on-line algorithm buys the machine of level $i+1$. Thus, O_{i+1} is obtained by the sum of O_i^j for $1 \leq j \leq s_{i+1}$, for the s_{i+1} phases of level i contained in the phase of level $i+1$, plus the capital cost c_{i+1} for buying the machine of level $i+1$ that ends the phase. Therefore

$$\begin{aligned} O_{i+1} &= \sum_{j=1}^{s_{i+1}} O_i^j + c_{i+1} \geq \sum_{j=1}^{s_{i+1}} \frac{i-1}{8} A_i^j - \frac{c_i}{2} + c_{i+1} \\ &\geq \frac{i-1}{8} A_{i+1} + \frac{3}{4} c_{i+1} \geq \frac{i}{8} A_{i+1} \ . \end{aligned}$$

   The first inequality stems by applying the inductive hypothesis. The second inequality is obtained from Lemma 4.4 and the relation $c_i = 2c_{i-1}$. Finally, the last inequality follows from Lemma 4.3.

2. In Case 2 the global cost of the on-line algorithm reaches the value $\frac{i+1}{2} c_{i+1}$, for $i \geq 1$. Then, applying Lemma 4.3, it follows that,

$$O_{i+1} \geq \frac{i+1}{2} c_{i+1} \geq \frac{i}{8} A_{i+1} \ .$$

3. In Case 3 the phase ends because a phase of a higher level ends in Case 1 or Case 2. The cost O_{i+1} of the incomplete phase is obtained by the sum of O_i^j for $1 \leq j \leq s_{i+1}$, for the s_{i+1} phases of level i contained in the phase of level $i+1$. Note that the last phase of level i is also incomplete. The claim is proved as follows:

$$\begin{aligned} O_{i+1} &= \sum_{j=1}^{s_{i+1}} O_i^j \geq \sum_{j=1}^{s_{i+1}} \frac{i-1}{8} A_i^j - \frac{c_i}{2} \\ &\geq \frac{i-1}{8} A_{i+1} - \frac{c_{i+1}}{2} + \frac{c_{i+1}}{4} \geq \frac{i}{8} A_{i+1} - \frac{c_{i+1}}{2} \ . \end{aligned}$$

   The first equality indicates the on-line global cost in the phase, while the first inequality is derived by applying the inductive hypothesis.

The second inequality is obtained from Lemma 4.4 and the relation between the capital costs of machines of level $i+1$ and i, while the final inequality is derived from Lemma 4.3.

Since the unique phase of level $\log C$ is a complete phase and its completion ends the sequence, we have that $ON = O_{\log C} \geq \frac{\log C - 1}{8} A_{\log C}$. The cost of the optimal adversary is at most $A_{\log C} + 2$, where the additional cost of 2 is the cost to buy a first machine, as the costs A_i are based on the assumption that a machine is available to the algorithm when the sequence starts. We get that $ON \geq \frac{\log C}{16}(\mathrm{OPT} - 2)$. Since the phase of level $\log C$ ends either when the on-line algorithm buys the machine of level $\log C$ or when its total cost reaches $\frac{\log C}{2} C$ we have that $ON \geq C > \frac{\log C}{8}$. We get $ON \geq \frac{\log C}{16}\mathrm{OPT} - \frac{\log C}{16} \cdot 2$ and $ON \geq \frac{\log C}{32}\mathrm{OPT}$.
■

The following corollary states the lower bound as a function of the ratio P between the highest and the lowest production costs, and of the maximum number of presented machines M.

Theorem 4.2 *If an algorithm for the non-convex on-line capital investment problem is ρ-competitive then $\rho = \Omega(\frac{\log\log P}{\log\log\log P})$ and $\rho = \Omega(\frac{\log M}{\log\log M})$.*

Proof. The claim follows by observing that in the sequence for the $\Omega(\log C)$ lower bound, the ratio between the maximum and the minimum production cost is $P = (\log C)^{(\sum_{k=1}^{\log C - 1} n_k(\log C))} = (\log C)^{O(\log C)!}$. Hence, it follows that $\Omega(\log C) = \Omega(\frac{\log\log P}{\log\log\log P})$. Similarly, the number of machines presented is $M \leq \sum_{k=1}^{\log C} n_k(\log C) = O((\log C)!)$, it therefore follows that $\Omega(\log C) = \Omega(\frac{\log M}{\log\log M})$. ■

5 Upper Bound for the Non-Convex Case

In this section we present an algorithm for the general (non-convex) case of the problem. This algorithm achieves a competitive ratio of $O(\min\{1 + \log C, 1 + \log\log P, 1 + \log M\})$.

5.1 The Algorithm

Given any new machine with production cost p_i, and capital cost c_i, our algorithm first rounds these costs up to the nearest power of two, i.e., if $2^{j-1} < c_i \leq 2^j$ then it sets $c_i = 2^j$, and if $2^{k-1} < p_i \leq 2^k$ then it sets $p_i = 2^k$.

The algorithm is defined as follows. Before producing the first unit buy the machine m_i that minimizes $p_i + c_i$ amongst all machines available at the beginning. Then produce the first unit of commodity. The initial cost $p_i + c_i$ is considered a *production* cost.

Before producing any subsequent unit, order all available machines by increasing production cost and (internally) increasing capital cost. Number the machines by index i, and let p_i, c_i be the production costs and capital costs, respectively. For all i, $p_i \leq p_{i+1}$, and if $p_i = p_{i+1}$, then $c_i \leq c_{i+1}$. Buy the machine with least i that satisfies the two following conditions:

- Its production cost p_i is smaller than the production cost of the current machine.
- A production cost of at least c_i has been spent since the last time a machine with capital cost c_i has been bought (or since the beginning of the run, if no such machine has been previously bought).

5.2 Analysis

We prove that the above algorithm achieves competitive ratio of $O(\min\{1 + \log C, 1 + \log\log P, 1 + \log M\})$. In the following analysis we assume that all capital and production costs are indeed powers of 2, as rounded by the on-line algorithm. Clearly, an adversary that uses this modified sequence incurs a cost of at most twice the cost incurred by the real adversary that uses the real sequence.

Denote by $\mathrm{ON} = \mathrm{ON}^c + \mathrm{ON}^p$ the total cost of the algorithm which is equal to the sum of the total capital cost ON^c and of the total production cost ON^p (which, by our definitions, includes the capital cost of the first machine bought).

Lemma 5.1 *The total capital cost* ON^c *is at most* $O(\log C)$ *times the total production cost* ON^p*.*

Proof. For a given j, consider all the machines of cost 2^j that are bought. A machine of cost 2^j can be bought only after an amount of 2^j has been spent on production since the last time a machine of the same cost has been bought (or since the beginning of the sequence, if not such machine was previously bought). It follows that for any j, the total cost of the algorithm for buying machines of cost 2^j is at most ON^p. Since there are at most $1 + \lceil \log C \rceil$ different costs for the machines, $\mathrm{ON}^c = O(\mathrm{ON}^p \cdot (1 + \log C))$. ■

Lemma 5.2 *The total capital cost* ON^c *is at most* $O(1 + \log M')$ *times the total production cost* ON^p*, where* M' *is the total number of machines bought.*

Proof. Let 2^l be the cost of the cheapest machine, and let 2^k be the cost of the most expensive machine such that $2^k \leq \mathrm{ON}^p$. All machines bought by the algorithm have costs between 2^l and 2^k. For any j, $l \leq j \leq k$, let b_j be the number of machines of cost 2^j bought by the algorithm. An upper bound on the capital cost spent by the algorithm is the maximum of $Z = \sum_{j=l}^{k} b_j 2^j$ as a function of the variables $b_j, j = l, \ldots, k$ subject to constraints $b_j 2^j \leq \mathrm{ON}^p$, and $\sum_{j=l}^{k} b_j = M'$.

We relax the above problem by allowing the variables b_j to assume non-integer values. Clearly the solution to this relaxed problem is also an upper bound on ON^c. Denote by $b_j^r, j = l, \ldots, k$, the variables of the relaxed problem. For the optimal solution of the relaxed problem, there are no h and h' such that $l \leq h < h' \leq k$, $b_h^r > 0$ and $b_{h'}^r < \frac{\mathrm{ON}^p}{2^{h'}}$. Otherwise, there would have been a solution with higher value of the objective function Z of the relaxed problem, achieved by reducing b_h^r and increasing $b_{h'}^r$ by the same amount, until either $b_h^r = 0$, or $b_{h'}^r = \frac{\mathrm{ON}^p}{2^{h'}}$.

From the above observation we derive an upper bound on the maximum of the objective function (and thus an upper bound on ON^c). If $\frac{\mathrm{ON}^p}{2^k} \geq M'$ then the maximum is achieved by setting $b_k^r = M'$ and $b_j^r = 0$ for $l \leq j \leq k-1$. In this case $\sum_{j=l}^{k} b_j 2^j \leq \mathrm{ON}^p$, and the lemma clearly holds.

If $\frac{\mathrm{ON}^p}{2^k} < M'$, let h^* be the maximum integer such that $\sum_{j=h^*}^{k} \frac{\mathrm{ON}^p}{2^j} \geq M'$. An upper bound on the maximum of the objective function is obtained by assigning $b_j^r = \frac{\mathrm{ON}^p}{2^j}$, $j = h^* + 1, \ldots, k$, $b_{h^*}^r = M' - \sum_{j=h^*+1}^{k} b_j^r \leq \frac{\mathrm{ON}^p}{2^{h^*}}$, and $b_j^r = 0$, for $j = l, \ldots, h^* - 1$. The upper bound on the value of the objective function is

$$\sum_{j=l}^{k} b_j^r 2^j \leq \sum_{j=h^*}^{k} b_k^r 2^{k-j} 2^j = \sum_{j=h^*}^{k} b_k^r 2^k \leq \mathrm{ON}^p \cdot (k - h^* + 1) \ .$$

It remains to show that $k-h^*+1 = O(1+\log M')$. By the definition of h^*, $\sum_{j=h^*+1}^{k} 2^{k-j} b_k^r = \sum_{j=h^*+1}^{k} b_j^r = \sum_{j=h^*+1}^{k} \frac{\mathrm{ON}^p}{2^j} < M'$. Therefore, we get $\sum_{j=h^*}^{k} 2^{k-j} b_k^r \leq 3M'$ and thus $(2^{k-h^*+1} - 1) \leq \frac{3M'}{b_k^r}$. Since $\mathrm{ON}^p \geq 2^k$, it follows that $b_k^r \geq 1$, and we obtain $2^{k-h^*+1} - 1 \leq 3M'$. Since $M' \geq 1$, we obtain $k - h^* + 1 = O(1 + \log M')$. ■

Corollary 5.1 *The total capital cost* ON^c *is at most* $O(1+\log M)$ *times the total production cost* ON^p.

Corollary 5.2 *The total capital cost* ON^c *is at most* $O(1 + \log\log P)$ *times the total production cost* ON^p.

Proof. The algorithm buys a machine only if the production cost decreases. Since all production costs are powers of 2, the algorithm buys at most $O(1 + \log P)$ machines. ■

Lemma 5.3 *At any time the total production cost* ON^p *of the on-line algorithm is at most twice the total cost of the off-line algorithm.*

Proof. Let p^t be the production cost incurred by the on-line algorithm to produce unit number t. Let ON_t^p be the production cost incurred by the algorithm to produce the first t units, i.e., $\mathrm{ON}_t^p = \sum_{i=1}^{t} p^i$. Let OPT_t be the lowest (optimal) cost to produce the first t units.

We prove by induction on t that $\mathrm{ON}_t^p \leq 2 \cdot \mathrm{OPT}_t$.

To produce the first unit the on-line algorithm buys the machine that minimizes the sum of production and capital costs. This is the minimum possible cost to produce the first unit. Thus, $\mathrm{ON}_1^p \leq \mathrm{OPT}_1$.

Consider unit t for $t > 1$, and assume that the claim holds for every unit number $\hat{t}$, $\hat{t} < t$. Let m be the machine used by ON to produce unit t. Let

m' be the machine used by OPT to produce unit t, p' its production cost and c' its capital cost.

If $p^t \leq p'$ then we have

$$\mathrm{ON}_t^p = \mathrm{ON}_{t-1}^p + p^t \leq 2 \cdot \mathrm{OPT}_{t-1} + p^t \leq 2 \cdot \mathrm{OPT}_{t-1} + p' \leq 2 \cdot \mathrm{OPT}_t \ .$$

We now consider the case in which $p' < p^t$. It follows that the on-line algorithm did not buy machine m' although it was available before unit t is produced. If this happens one of the following holds:

1. The production cost incurred by the on-line algorithm by time $t-1$ is less than c'. On the other hand, the optimal off-line algorithm buys machine m', incurring a cost of c'. It follows that
$$\mathrm{ON}_t^p = \mathrm{ON}_{t-1}^p + p^t \leq 2 \cdot \mathrm{ON}_{t-1}^p \leq 2c' \leq 2 \cdot \mathrm{OPT}_t \ .$$
2. Some machine of cost c' was previously bought by the on-line algorithm, but the production cost incurred by the algorithm since then is less than c'. Assume that such machine was bought just before unit $\bar{t}$ was produced. As unit t is produced with production cost higher than p', we can conclude that m' was not available before unit $\bar{t}$ was produced. Thus, m' was bought by the off-line algorithm after unit $\bar{t}$ is produced. On the other hand, the on-line production cost since the production of unit $\bar{t}$ is less than c'. Therefore, we have
$$\mathrm{ON}_t^p = \mathrm{ON}_{\bar{t}-1}^p + \sum_{i=\bar{t}}^{t-1} p^i + p^t \leq \mathrm{ON}_{\bar{t}-1}^p + 2 \sum_{i=\bar{t}}^{t-1} p^i \leq 2 \cdot \mathrm{OPT}_{\bar{t}-1} + 2c' \leq 2 \cdot \mathrm{OPT}_t \ .$$

■

We conclude with the following theorem, whose proof is straightforward from the previous lemmata.

Theorem 5.1 *The competitive ratio of the on-line capital investment algorithm described above is* $O(\min\{1+\log C, 1+\log\log P, 1+\log M\})$.

Acknowledgments We thank Ran El-Yaniv for useful comments and the anonymous referees for their useful comments.

References

[1] A. Chou, J. Cooperstock, R. El-Yaniv, M. Klugerman and F. Leighton. The Statistical Adversary Allows Optimal Money-making Trading Strategies. In *Proceedings of the 6th Annual ACM/SIAM Symposium on Discrete algorithms*, 1995.

[2] R. El-Yaniv, A. Fiat, R. Karp and G. Turpin. Competitive Analysis of Financial Games. In *Proc. of the 33rd IEEE Annual Symposium on Foundations of Computer Science*, 1992.

[3] R. El-Yaniv and R.M. Karp. The Mortgage Problem. In *Proceedings of the 2nd Israeli Symposium on Theory of Computing and Systems*, pp. 304–312, June 1993.

[4] R.M. Karp, On-line Algorithms Versus Off-line Algorithms: How Much is it Worth to Know the Future?. In *Proc. World Computer Congress*, 1992.

[5] P. Raghavan. A Statistical Adversary for On-line Algorithms. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, Vol 7:79-83, 1991.

[6] D.D. Sleator and R.E. Tarjan. Amortized Efficiency of List Update and Paging Rules. *Communications of the ACM*. 28:202–208, February 1985.