Adjustable Coins

Shlomo Moran and Irad Yavneh

Computer Science Department, Technion. Israel

Abstract

In this paper we consider a scenario where there are several algorithms for solving a given problem. Each algorithm is associated with a probability of success and a cost, and there is also a penalty for failing to solve the problem. The user may run one algorithm at a time for the specified cost, or give up and pay the penalty. The probability of success may be implied by randomization in the algorithm, or by assuming a probability distribution on the input space, which lead to different variants of the problem. The goal is to minimize the expected cost of the process under the assumption that the algorithms are independent. We study several variants of this problem, and present possible solution strategies and a hardness result.

1 Introduction

Some optimization problems concern optimal ordering of certain tasks that aim at achieving some common goal (see, e.g., [1]). A possible scenario can be described as follows: the king's daughter is approaching marrying age. The miser king, who cares about money more than anything else, estimates that if she doesn't marry then he will need to spend E gold pieces to continue supporting her for the rest of their lives. There are n princes he can invite to try to win her heart, and he must choose the order, knowing that the cost of travel room and board for $prince_i$ is μ_i, and the probability that he will win the princess's heart is P_i. What order minimizes the king's expected cost? In a more challenging scenario, there are n kingdoms with several eligible princes in each kingdom, each with his own μ and P, and the king must also decide which prince he should invite from each kingdom (he cannot invite more than one).

Such problems can be described as one player games, which we call *adjustable coins* games. An adjustable coin—A-coin in short—is a coin whose bias can be controlled by the user: the probability of success (rolling on one) can be increased, for a price. Formally, an A-coin is defined by a monotone increasing function $\mu : D_\mu \rightarrow \mathbb{R}^+$ where $D_\mu \subseteq (0,1]$ contains the possible success probabilities, and for $P \in D_\mu$, $\mu(P)$ is the nonnegative fee for tossing the coin with success probability P. If $|D_\mu| = 1$ then μ is a *simple coin*, or just coin, denoted by a pair $c = (P, \mu)$. Thus an A-coin μ can be viewed as a set of the simple coins $\{(P, \mu(P)) : P \in D_\mu\}$.

In the games studied in this paper the player is given a set $\mathcal{C}$ of A-coins and a penalty $E > 0$, and in each step she may select an A-coin $\mu \in \mathcal{C}$, and a probability $P \in D_\mu$, and toss the coin for the fee $\mu(P)$. If the coin rolls on one the player terminates the game without paying a penalty, else she either takes another step, or terminates the game and pays the penalty E. Thus, a strategy for an A-coin game maps each pair $(\mathcal{C}, E)$ of the set of A-coins and a penalty to a sequence SEQ of the coins tossed by the player when the outcomes of all tosses are zeros. The goal is to minimize the expected cost—the total amount paid. Variants of this game are determined by the nature of the coins in $\mathcal{C}$, the rules by which coins can be selected at each step, possible restrictions on the termination rule, etc. Specifically, we distinguish between reusable coins, which can be tossed many times, and one time coins, which can be tossed only once. The latter case corresponds to a scenario where one or more deterministic tests should be taken on a single item selected at random from a known distribution—repeating a test on the selected item just reproduces the initial outcome.

In Section 2 we study variants of the game for simple coins. In Section 3 we study the case where the A-coins are discrete, i.e., defined for finitely many values. In Section 4 we study the game for A-coins which are piecewise continuous functions.

2 Simple coins

In this section we study optimal strategies for the game where all A-coins are simple. A useful property of a simple coin $c = (P, \mu)$ is its *rate*, given by the ratio $r = \mu/P$. The notation $c \sim (P, r)$ means that the simple coin c has probability P and is of rate r, i.e., $c = (P, rP)$.

2.1 Single simple coin, single toss

In the simplest scenario we are given a simple coin $c = (P, \mu)$ and a penalty E, and we must decide whether tossing the coin c is *beneficial*, i.e., reduces the expected cost of the game.

The expected payment when c is tossed is given by

$$COST\,(c, E) = \mu + (1 - P)E = E - (PE - \mu) \tag{1}$$

which means that the *benefit* of using c w.r.t. E is $(PE - \mu)$ (see Figure 1). $COST(c, E)$ as a function of the rate r of c is given by

$$COST\,(c, E) = Pr + (1 - P)E = E - P(E - r) \tag{2}$$

implying that the benefit of c w.r.t. E is $P(E - r)$. This means that an optimal strategy for this case is to toss c if and only if its rate is smaller than E.

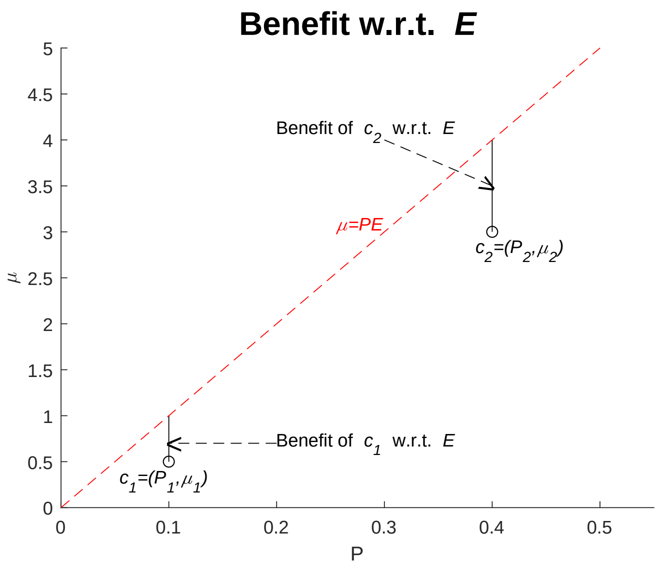

Figure 1: **Benefit for a given penalty.** A coin $c = (P, \mu)$ is beneficial for penalty E iff (P, μ) lies below the line $\mu = PE$, and the benefit of c w.r.t. E is $PE - \mu$. Thus both c_1 and c_2 are beneficial for E. The benefit of c_2 is larger since it is further away from the line $\mu = PE$, i.e. $\frac{\mu_2 - \mu_1}{P_2 - P_1} < E$.

In a natural extension of the "single coin single toss" game we are given a sequence of coins $SEQ = (c_1, \ldots, c_n)$ and a penalty E. Our task is to decide for each coin, in the given order, if it should be tossed or skipped, so that the expected cost of the implied game is minimized. For $n = 1$ this is the single coin single toss game. For $n > 1$ we use *backwards induction*: Suppose we have an optimal strategy for $SEQ' = (c_2, \ldots, c_n)$ and E, whose cost is E'. Then, by a simple calculation, an optimal strategy for SEQ and E is: toss c_1 iff $r_1 < E'$, and then, if the coin rolls on tail, continue with the optimal strategy for SEQ' and E.

2.2 Multiple simple coins, single toss

Suppose next that we are given a (finite) set $\mathcal{C}$ of simple coins and a penalty E, and we wish to select an *optimal* coin $c \in \mathcal{C}$ which minimizes $COST(c, E)$, the cost of the single toss game. Let $c = (P, \mu)$. By Equation (1), $COST(c, E)$ is minimized if and only if the point $c = (P, \mu)$ lies below the line $\mu = PE$ and at a maximal distance (see Fig. 1).

2.3 Reusable simple coins, multiple tosses

Suppose now that the coins in $\mathcal{C}$ are *reusable*, that is, we may toss each coin in $\mathcal{C}$ multiple times. An elementary calculation shows that if there is no bound on the number of tosses, then tossing a coin of minimal possible rate r_{min} until it rolls on one yields an expected cost r_{min}. Thus an optimal strategy is: if $r_{min} < E$ then toss a coin c_{min} of rate r_{min} until it rolls on one, else pay the penalty E and terminate.

Assume now that we are allowed to make no more than k tosses for some $k \geqslant 0$. Given a set of reusable simple coins $\mathcal{C}$ and a penalty E, if $r_{min} \geqslant E$ then do nothing and pay the penalty E. Else an optimal strategy is obtained again by a backwards induction:

If $k = 0$ or $E \leqslant r_{min}$ then do nothing and pay the penalty E. So assume that $E > r_{min}$, and let $E_k = E_k(\mathcal{C}, E, k)$ be the expected cost of an optimal strategy for set of coins $\mathcal{C}$, penalty E and k tosses (in particular $E_0 = E$). Then $E_k > r_{min}$ and an optimal strategy for $k+1$ tosses is obtained by first selecting and tossing a coin $c_{k+1} \in \mathcal{C}$ which minimizes $COST(c_{k+1}, E_k)$ as in Section 2.2 (note that $COST(c_{k+1}, E_k) < E_k$ since $E_k > r_{min}$); if c_{k+1} rolls on one then stop, else execute the optimal strategy for $\mathcal{C}, E$ and k. Thus $E_{k+1} = COST(c_{k+1}, E_k) > r_{min}$.

2.4 One-time simple coins, multiple tosses

We now assume that each coin in $\mathcal{C}$ can be tossed at most once, and we are allowed to toss as many coins as we wish.

Consider first a variant of this game in which termination is possible only if either some coin rolls on one, or after *all* coins rolled on zero. We note that this variant is, in fact, the problem of optimal ordering of independent tests that was studied in [1], where each coin corresponds to a test, and it is needed to check if at least one test fails.

A strategy for this latter game is a permutation of all the available coins. Thus given a set of n one-time simple coins $\mathcal{C} = \{c_1, \ldots, c_n\}$, where $c_i = (P_i, \mu_i)$, and a nonnegative penalty E, we need to find an *optimal ordering* of the coins in $\mathcal{C}$, which minimizes the expected cost.

The expected cost for ordering $SEQ = (c_1, c_2, ..., c_n)$ and penalty E is given by:

$$\begin{aligned} COST(SEQ, E) &= P_1\mu_1 + (1-P_1)P_2(\mu_1+\mu_2) \\ &+\ldots+ (1-P_1)(1-P_2)\cdots(1-P_n)(\mu_1+\ldots+\mu_n+E) \end{aligned} \tag{3}$$

By straightforward induction, $COST(SEQ, E)$ can also be expressed as the following convex combination of the coin rates $r_1, \ldots, r_n$ and E.

$$\begin{aligned} COST(SEQ, E) &= P_1r_1 + (1-P_1)P_2r_2 + \ldots + (1-P_1)\cdots(1-P_{i-1})P_ir_i \\ &+\ldots+ (1-P_1)\cdots(1-P_n)E \end{aligned} \tag{4}$$

Equation (4) implies the following useful lemma:

Lemma 1. *Let $SEQ = (c_1, \dots, c_n)$, where $c_k \sim (P_k, r_k), k = 1, \dots, n$, and let SEQ' be obtained from SEQ by interchanging c_i and c_{i+1}. If $r_i \geqslant r_{i+1}$ then $COST(SEQ', E) \leqslant COST(SEQ, E)$, with equality iff $r_i = r_{i+1}$.*

Proof. By substituting in Equation (4) we get

$$COST(SEQ', E) - COST(SEQ, E) = \left(\prod_{k=1}^{i-1}(1-P_k)\right)P_iP_{i+1}(r_{i+1} - r_i)$$

which is negative for $r_i > r_{i+1}$, and equals zero iff $r_i = r_{i+1}$. □

Lemma 1 implies:

Lemma 2. *Let $\mathcal{C} = \{c_1, \dots, c_n\}$ be a set of one-time simple coins, where the rate of c_i is r_i. Then for each penalty E and each permutation π, $(c_{\pi(1)}, \dots, c_{\pi(n)})$ is an optimal ordering of $\mathcal{C}$ w.r.t E if and only if $r_{\pi(i)} \leqslant r_{\pi(i+1)}$ for $i = 1, \dots, n-1$.*

Observe that the optimal orderings of a set of coins are independent of the value of the penalty E. We note that Lemma 2 is equivalent to Theorem 1 of [1] which considered optimal ordering of independent tests.

Assume now that the player can terminate the game at any time (i.e., even if no coin rolled on one and some coins were not tossed yet). By Lemma 2 and the comment at the end of Section 2.1, an optimal strategy is obtained by an optimal ordering of the coins in $\mathcal{C}$ whose rates are smaller than E.

Lemma 3. *The optimal strategies for a set $\mathcal{C}$ of one-time simple coins and a penalty E are obtained by the optimal orderings of the coins in $\mathcal{C}$ whose rates are smaller than E.*

2.4.1 One-time simple coins, bounded number of tosses

Assume now that the coins are not reusable, and we may use at most k coins for some $k \geqslant 0$. This problem can be solved by the following dynamic programming algorithm. First sort the coins whose rates are smaller than E by increasing rates (ties are broken arbitrarily). Let the sorted list be $(c_1, \dots, c_n)$, where $c_i = (P_i, \mu_i)$.

For $i = 1, \dots, n$ and $j = 0, \dots, \max(k, n-i+1)$, let $OPT(i, j)$ be the value of the optimal strategies for the sequence $(c_i, c_{i+1}, \dots, c_n)$ and penalty E which use at most j coins. Thus our task is to find $OPT(1, k)$. This can be done in kn steps by setting $OPT(i, 0) = E$ for $i = 1, \dots, n$, $OPT(n, j) = \mu_n + (1 - P_n)E$ for $j \geqslant 1$, and then using the following recursive formula for $i = n-1, n-2, \dots, 1, j = 1, 2 \dots k$:

$$OPT(i, j) = \min\Big(OPT(i+1, j), \quad \mu_i + (1-P_i)OPT(i+1, j-1) \Big)$$

This implies the following.

Lemma 4. *Given a set $\mathcal{C}$ of n one-time simple coins and bound k on the number of tosses, an optimal strategy for the implied game can be found in $O(kn)$ time.*

The results for simple coins are summarized in Table 1.

#coins	Reuse	Tosses	Discussed	Comments
1	No	1	Section 2.1	Toss if $rate < E$
1	Yes	Unlimited		$rate < E \rightarrow$ toss until success
n	No	Unlimited, order given		Backwards induction
n	Yes	Unlimited	Section 2.3	Toss c_{min} until success
n	Yes	Bounded		Backwards induction
n	No	Unlimited	Lemma 3	Toss by increasing rates
n	No	Bounded	Lemma 4	Backwards dynamic programming

Table 1: Optimal solutions for Simple Coins

3 Discrete coins

An A-coin μ is discrete if its domain D_μ is finite and contains at least two simple coins. Given a penalty $E > 0$, $COST(\mu, E)$ is naturally defined as

$$COST(\mu, E) = \min_{c \in \mu} COST(c, E)$$

An A-coin μ can be viewed as the set of the simple coins $\{(P, \mu(P)) : P \in D_\mu\}$. We assume that this set does not contain redundant coins, in the sense of the following definitions.

Definition 1. *A coin $c \in \mu$ is* essential *for μ (or just essential when μ is clear) if there is a penalty E_c s.t. for any other coin $d \in \mu$, $COST(c, E_c) < COST(d, E_c)$. Given such c and E_c, the* supporting line for c and E_c *is the line L_c which contains c and is parallel to the line $\mu = PE_c$ (see Fig. 2).*

Note that if c is not essential then for each $E > 0$, $COST(\mu \setminus \{c\}, E) = COST(\mu, E)$, meaning that c can be removed from μ without reducing its quality. Hence, we assume from now on that a discrete coin contains only essential simple coins.

Definition 2. *A discrete coin μ is* efficient *if each coin in μ is essential.*

Efficient discrete coins posses nice geometrical properties, depicted in Fig. 2:

Lemma 5. *Let μ be an efficient discrete coin. Then*

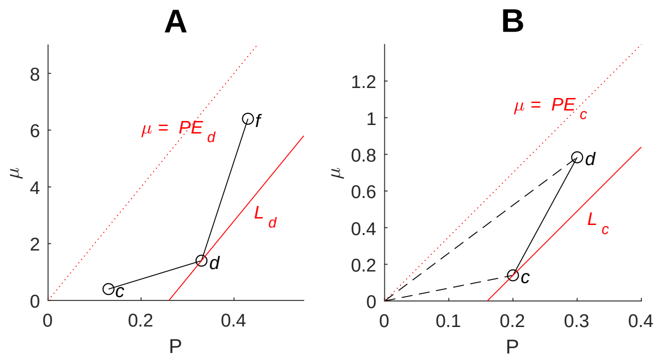

Figure 2: A: L_d, the supporting line for d and E_d, is a supporting line of the convex hull of $\mu = \{c, d, f\}$. B: The rate of c (i.e., the slope of the dashed line segment connecting c to the origin) is smaller than the rate of d, (the slope of the line connecting d to the origin).

1. *μ is a strictly convex function on D_μ, and*
2. *The function $r(P) = \mu(P)/P$ is strictly increasing on D_μ.*

Sketch of proof. Let $c \sim (P_c, r_c)$ be any coin in μ. Then L_c, the supporting line for c and E_c, is a supporting line of the convex hull of μ which contains c but no other coin in μ (see Fig. 2A). This proves (1).
To show (2), let $d \sim (P_d, r_d)$ be another coin in μ, where $P_c < P_d$. Then $r_c < E_c$ (since c lies below the line $\mu = PE_c$), and d lies strictly above L_c, the supporting line for c and E_c, whose slope is E_c. Hence $r_c < r_d$ (see Fig. 2B). □

3.1 Reusable discrete coins

Optimal strategies for reusable discrete coins are implied by such strategies for sets of simple coins: Given a set $\mathcal{C}$ of discrete coins and a penalty E, let $\mathcal{C}'$ be the union of the discrete coins in $\mathcal{C}$. Applying the optimal strategies for reusable simple coins presented in Section 2.3 on $\mathcal{C}'$ and E yields optimal strategies for $\mathcal{C}$ and E. For example, when the number of tosses is unbounded, Lemma 5 implies the following strategy for a reusable efficient discrete coin μ: Let $c_{min} \sim (P_{min}, r_{min})$ be the coin with the minimum success probability in μ. If $E > r_{min}$ then repeatedly toss c_{min} until it rolls on one, else pay the penalty E and terminate.

3.2 One-time discrete coins

Suppose we are given a sequence of one-time discrete coins and we need to decide for each coin μ, in its turn, whether to toss a simple coin from μ, and if so which one. Then we can use a simple extension of the strategy for simple coins in Section 2.1: Given an optimal strategy for a sequence $(\mu_2, \ldots, \mu_n)$ with optimal cost E', an optimal strategy for $(\mu_1, \mu_2, \ldots, \mu_n)$ is obtained by skipping μ_1 if $COST(\mu_1, E') \geqslant E'$, and tossing a coin $c \in \mu_1$ which minimizes $COST(\mu_1, E')$ and upon failure executing the optimal strategy for $(\mu_2, \ldots, \mu_n)$ otherwise.

Finding an optimal strategy for one-time discrete coins when the sequence is not given is complicated by the fact that we need to decide which simple coin should be selected from each discrete coin (if any) before the order of tosses is known. Once these coins are selected, all we need to do is to toss them in an increasing order of their rates, until some coin rolls on one (or all coins roll on zero), as discussed in Section 2.4. Thus, the strategy is determined by the way the simple coins are chosen from the given discrete ones. We next show that such a selection is NP Hard even in a highly restricted variant of the problem.

3.2.1 The $\{0,1\}$ discrete coins problem

We now present a restricted version of the A-coins problem—the $\{0,1\}$ A-coins problem, and prove that it is NP hard. An instance $(\mathcal{C}, E)$ for this problem consists of a set $\mathcal{C} = \{A_1, \ldots, A_n\}$ of one-time discrete coins, where each A_i contains two coins c_i, d_i, s.t. the rate of all c_i coins is 0 and the rate of all d_i coins is 1, i.e., $c_i = (0, P_{i,0})$ and $d_i = (P_{i,1}, P_{i,1})$, where $0 \leqslant P_{i,0} < P_{i,1} \leqslant 1, i = 1, \ldots, n$. Let $h_i = 1 - P_{i,0}$ and $\ell_i = 1 - P_{i,1}$; then $h = (h_1, \ldots, h_n)$ and $\ell = (\ell_1, \ldots, \ell_n)$ are the *failure probability vectors* of c_i's and d_i's (note that $h_i > \ell_i$ for all i).

NP Hardness of the $\{0,1\}$ discrete coin problem

Consider an instance to the $\{0,1\}$ problem with penalty $E > 1$, and let $D = E - 1$. By Lemma 3 an optimal strategy for this instance is obtained by selecting a set $S \subseteq I_n$ of indices i for which the coin c_i is chosen, and tossing the coins in S first. The cost of this strategy can be calculated as follows: charge each event (sequence of tosses) in which the first $|S|$ tosses are zero by one, and in addition charge the event in which all tosses are zero by D. The resulting cost is

$$COST_{h,\ell,D}(S) = \prod_{i\in S} h_i \left(1 + D \cdot \prod_{i\in I_n\setminus S} \ell_i\right) \tag{5}$$

For a vector $a = (a_1, \ldots, a_n)$, let $prod(a)$ be the product $\prod_{i=1}^{n} a_i$. Then $\prod_{[i\in I_n\setminus S]} \ell_i = \frac{prod(\ell)}{\prod_{i\in S}\ell_i}$. Hence, letting $b_i = \frac{\ell_i}{h_i} < 1$ and $b = (b_1, \ldots, b_n)$, we can rewrite (5) as a function of h and b:

$$COST_{h,b,D}(S) = \prod_{i\in S} h_i + \frac{D\cdot prod(\ell)}{\prod_{i\in S} b_i} \tag{6}$$

Let $H_S = \prod_{i\in S} h_i$ and $B_S = \prod_{i\in S} b_i$. Then Equation (6) can be rewritten as

$$COST_{\ell,b,C}(S) = H_S + \frac{C}{B_S} \tag{7}$$

where $C = D \cdot prod(\ell)$. Consider now the case $h = b$ (i.e., $\forall i : \ell_i = h_i^2$).Then we get

$$COST_{h,C}(S) = H_S + \frac{C}{H_S} \tag{8}$$

Since the function $f(x) = x + \frac{C}{x}$ has a unique minimum at $x = \sqrt{C}$, we get that $COST_{h,C}(S) \geqslant \sqrt{C} + \frac{1}{\sqrt{C}}$, with equality iff $H_s = \sqrt{C}$. This implies the following:

Lemma 6. *Let $\mathcal{C} = \{A_i : i = 1, \ldots, n\}$ be a set of A-coins with $A_i = \{c_i, d_i\}$ where*

$$c_i \sim (0, 1 - h_i), \quad d_i \sim (1, 1 - h_i^2)$$

and let $C = D\cdot(prod(h))^2$. Let further $OPT(\mathcal{C}, D{+}1)$ be the value of the optimal solution to the $\{0,1\}$ A-coins problem for $\mathcal{C}$ and penalty $D + 1$. Then

$$OPT(\mathcal{C}, D+1) \geqslant \sqrt{C} + \frac{1}{\sqrt{C}}$$

with equality iff for some $S \subseteq I_n$ it holds that $H_S = \sqrt{C}$.

Lemma 6 now implies:

Theorem 7. *The $\{0,1\}$ A-coins problem is NP Hard.*

Outline of proof. By a reduction from the NP hard *subset product* problem [2]:
Input: An $n + 1$ tuple of natural numbers $(m_1, \ldots, m_n, N)$
Property: There is a subset $S \subseteq I_n$ such that $\prod\limits_{i\in S} m_i = N$.

Given an instance $(m_1, \ldots, m_n, N)$ to the subset product problem, we reduce it to an instance $(\mathcal{C}, D+1)$ to the $\{0,1\}$ problem in which $\mathcal{C} = \{A_1, \ldots, A_n\}$, where for each i, $c_i \sim (0, 1 - \frac{1}{m_i})$ and $d_i \sim (1, 1 - \frac{1}{m_i^2})$ - i.e., $h_i = \frac{1}{m_i}$ and $\ell_i = \frac{1}{m_i^2}$. In addition, we set D to $\left(\frac{prod(m)}{N}\right)^2$.

Let $C = D \cdot (prod(h))^2 = \frac{D}{(prod(m))^2}$. Then from Lemma 6 it follows that $OPT(\mathcal{C}, D+1) \geqslant \sqrt{C} + \frac{1}{\sqrt{C}}$, and $OPT(\mathcal{C}, D+1) = \sqrt{C} + \frac{1}{\sqrt{C}}$ iff there is a subset $S \subseteq I_n$ such that $\prod_{i \in S} m_i = N$. The theorem follows. $\square$

Note: The rates 0 and 1 in Theorem 7 could be replaced by any pair of distinct rates $0 \leqslant a < b$.

A summary of the results for discrete coins appears in Table 2 below.

#Coins	Reuse	Tosses	Discussed	Comments
1	No	1	Lemma 5	Extensions of solutions for simple coins
n	Yes	Unlimited	Section 3.1	
n	Yes	Bounded		
n	No	Unlimited, order given	Section 3.2	
n	No	Unlimited	Theorem 7	NP hard even for {0,1} coins

Table 2: Solutions and hardenss results for Discrete Coins

4 Continuous coins

A continuous adjustable coin, or CA-coin, enables the user to smoothly and continuously adjust the desired success probability. As a possible example, assume an algorithm which gets a composite integer n as an input, and repeatedly attempts to find a divisor of n. $\mu(P)$ is the cost (e.g. running time) of finding a divisor with probability P, and the penalty E is the loss implied by failing to find a divisor. The reusable version assumes that the algorithm is randomized, and the one time version assumes that the composite number is selected at random from a given distribution.

In general, a CA-coin is defined by a non-decreasing cost function $\mu(P) : [P_{min}, P_{max}] \to \mathcal{R}$ where $0 < P_{min} < P_{max} \leqslant 1$ are the minimum and maximum success probabilities supported by the coin. The (optimal) cost of a CA-coin for a penalty E is naturally defined as

$$COST(\mu, E) = \min_{P \in [P_{min}, P_{max}]} COST(P, \mu(P)) = \min_{P \in [P_{min}, P_{max}]} \mu(P) + (1-P)E$$

We assume, as in the case of discrete A-coins, that each CA-coin is efficient, that is, for each $P_0 \in [P_{min}, P_{max}]$ there is a penalty E_{P_0} s.t. $COST((P, \mu(P)), E_{P_0})$ has a unique minimum at $P = P_0$. By arguments similar to the ones used in Lemma 5, this implies that a CA-coin μ is a nonnegative convex function on $[P_{min}, P_{max}]$, and that $r(P) = \mu(P)/P$ is a strictly increasing function of P. We restrict our attention to *regular* CA-coins $\mu(P)$ which are defined and twice continuously differentiable on the segment $[P_{min}, P_{max}]$. This implies that

$$\frac{d\mu}{dP} > 0 \text{ and } \frac{d^2\mu}{dP^2} > 0, \quad P \in (P_{min}, P_{max})$$

4.1 Single CA-coin, single toss

Suppose we are given a single (efficient) CA-coin and are allowed to toss it once at most. That is, given E we must decide whether we wish to use our CA-coin at all, and if we do, which value of P we should choose so as to minimize the expected cost. Recall that the expected cost is given by

$$COST\,(\mu(P), P, E) = \mu(P) + (1-P)E = P \cdot r(P) + (1-P)E$$

where $r(P) = \mu/P$. Differentiating $COST$ with respect to P we obtain

$$\frac{dCOST}{dP} = \frac{d\mu}{dP} - E$$

Differentiating $r(P)$ and noting that it is strictly increasing yield

$$\frac{dr}{dP} = \frac{1}{P}\left(\frac{d\mu}{dP} - r\right) > 0$$

Observe that for $P_0 \in (P_{min}, P_{max})$, for each coin $c_0 = (P_0, \mu(P_0))$ there is a unique supporting line, namely, the tangent to μ at c_0. This yields the following conclusions. Let $r_{min} = r(P_{min})$, E_{low} be the derivative of μ at P_{min}, and E_{high} be the derivative of μ at P_{max}—see Fig. 3. Then

1. If $E \leqslant r_{min}$ then the coin is not beneficial for E, i.e., $COST(\mu, E) = E$. Otherwise the coin is beneficial for E, and the maximum benefit for a given penalty E is attained as follows:

2. If $r_{min} < E \leqslant E_{low}$ then the benefit is maximized at P_{min}.

3. If $E_{low} < E < E_{high}$ then the maximal benefit is attained at the unique internal value $P_{opt} \in (P_{min}, P_{max})$, where $\frac{d\mu}{dP} = E$.

4. If $E_{high} \leqslant E$ then the benefit is maximized at P_{max}.

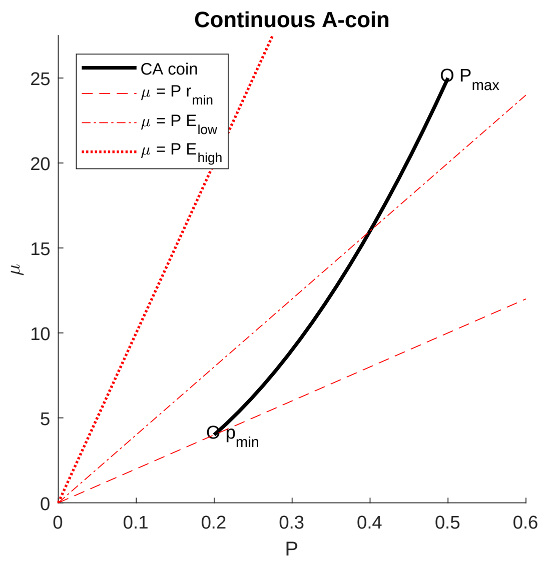

Figure 3: $c_{min} = (P_{min}, \mu(P_{min}))$, $c_{max} = (P_{max}, \mu(P_{max}))$, and $r_{min} = \frac{\mu(P_{min})}{P_{min}}$. If $E < r_{min}$ then c_{min} lies above the line $\mu = PE$, meaning that the CA-coin is not beneficial for E. For $E \in (r_{min}, E_{low}]$ any supporting line of $\mu = PE$ passes through c_{min}. For $E \in (E_{low}, E_{high})$ there is a unique supporting line that passes through an internal point of the coin, and for $E \in [E_{high}, \infty)$ any supporting line passes through c_{max}.

4.2 Reusable CA-coins

Suppose next that we are given a single regular CA-coin which we may toss multiple times, selecting P for each toss. The optimal strategy for minimizing the expected cost as a function of the number of tosses we are allowed, is obtained recursively from the observations of the previous subsection 4.1. If the number of tosses is unlimited and $r_{min} < E$, then we can reduce the expected cost to r_{min} by repeatedly tossing the coin with $P = P_{min}$ until we succeed. If the number of tosses is bounded then we can use the recursive approach described in Section 2.3.

When we are given a few regular CA-coins $\{\mu_1, \ldots, \mu_k\}$ which we may toss multiple times, we adapt a variant of strategies used in Sections 2.3 and 3.2: whenever we need to toss a coin for a given penalty E, we select μ_j for which $COST(\mu_j, E)$ is minimized. This can be done by computing $COST(\mu_i, E)$ for all $i \in [1, k]$.

5 Conclusion

We present the concept of adjustable coins, which aims to model a scenario in which various algorithms for solving a given problem can be applied: each such algorithm is modeled by an adjustable coin, which is characterized by a cost and a probability of success. The related optimization problem is: given a set of independent A-coins and a penalty for failing to solve the problem, find a sequence of coin-tosses which minimizes the expected cost, subject to possible further restrictions.

We note that all our solutions are by offline algorithms, which require that the full set (or sequence) of coins is given before the first coin is tossed. An interesting problem is what conditions enable useful algorithms which use only partial information on the available coins (e.g., that the coins are drawn from a known distribution, or that only a limited number of coins is known ahead of time).

References

[1] D. Berend et al. “Optimal ordering of independent tests with precedence constraints”. In: *Discrete Applied Mathematics* 162 (2014), pp. 115–127.

[2] S. Moran. “General approximation algorithms for some arithmetical combinatorial problems”. In: *Theoretical Computer Science* 14.3 (1981), pp. 289–303.