ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computational Geometry 34 (2006) 35–48
Computational Geometry
Theory and Applications
www.elsevier.com/locate/comgeo

Moving coins

Manuel Abellanas [a], Sergey Bereg [b], Ferran Hurtado [c], Alfredo García Olaverri [d], David Rappaport [e,*], Javier Tejel [d]

[a] *Universidad Politécnica de Madrid, Spain*
[b] *University of Texas at Dallas, USA*
[c] *Universitat Politècnica de Catalunya, Spain*
[d] *Universidad de Zaragoza, Spain*
[e] *Queen's University, Canada*

Received 11 October 2004; received in revised form 13 June 2005; accepted 23 June 2005

Available online 5 October 2005

Communicated by J. Akiyama, M. Kano and X. Tan

Abstract

We consider combinatorial and computational issues that are related to the problem of moving coins from one configuration to another. Coins are defined as non-overlapping discs, and moves are defined as collision free translations, all in the Euclidean plane. We obtain combinatorial bounds on the number of moves that are necessary and/or sufficient to move coins from one configuration to another. We also consider several decision problems related to coin moving, and obtain some results regarding their computational complexity.

1. Introduction

Consider a collection of discs or *coins*. The coins are found resting on a plane surface so that no two overlap. We explore issues involved in moving the coins from their initial positions to some desired final position.

To be more precise, we can move a coin centered at point a to a position centered at point b if the trajectory of the coin along the line segment ab does not collide with another coin. We say that such a translation in one fixed direction is one *move*. We are given as input a set of coins $C = \{c_1, c_2, \ldots, c_n\}$ positioned at initial source locations $P = \{p_1, p_2, \ldots, p_n\}$ and a set of final destinations $Q = \{q_1, q_2, \ldots, q_n\}$, where P and Q are sets of points. Associated with each coin c_i is $a_i \subseteq Q$, a set of possible destinations. As output we need to produce an *itinerary*, an ordered list of moves so that each coin moves to one of its possible destinations. The objective is to produce an efficient itinerary. The *cost* of an itinerary is simply the number of moves used.

* Corresponding author.
E-mail addresses: mabellanas@fi.upm.es (M. Abellanas), besp@utdallas.edu (S. Bereg), hurtado@ma2.upc.edu (F. Hurtado), olaverri@unizar.es (A.G. Olaverri), daver@cs.queensu.ca (D. Rappaport), jtejel@unizar.es (J. Tejel).

0925-7721/$ – see front matter © 2005 Published by Elsevier B.V.
doi:10.1016/j.comgeo.2005.06.005

1.1. Motivation

This problem is motivated by measuring the difference between various configurations. For example one can measure the difference between two strings of text by their *edit distance* [10]. The edit distance is the minimum number of text editor operations needed to go from one string to another. A distance with a more geometric flavor is the *earth movers distance* [13]. The earth movers distance measures the minimum amount of work needed to go from one configuration to another. The notion of work is flexible and conforms to the application. Thus our problem of moving coins is in the same vein as the previous examples. We are interested in the minimum number of move operations needed to go from one configuration of coins to another.

Our problem can also be viewed as a simplified model of multi-robot path planning. Consider a collection of robots, whose footprints are discs, maneuvering in a common workspace. A robot's tasks may take it from one destination to another. Our notion of moving a coin to one of its possible destinations is a simplified way to model this type of situation. A survey paper by Hwang and Ahuja [9] discusses the general robot path planning problem and multi-robot path planning in particular.

Erik and Martin Demaine with Helena Verrill [4] examine coin moving puzzles, that is, moving coins from one configuration to another subject to some given constraints. They consider a model where unlabeled unit coins are located on a grid, and may be picked up (as opposed to sliding on the plane) and placed on an unoccupied grid position adjacent to at least two other coins. They present theorems on the solvability of such puzzles, and algorithms to produce worst case optimal solutions when they exist. They also include references to other similar puzzles, including some sliding coin puzzles.

1.2. Variations

We consider several different versions of our coin moving problem. Some of our results are combinatorial and relate to upper and lower bounds for the number of moves that are necessary or sufficient to go from one configuration to another. In these cases our upper bound arguments imply polynomial time algorithms, however, we do not dwell on the actual complexity of the algorithms. We also consider the computational complexity of some coin moving problems.

For our combinatorial results we assume that we are given an image of both the initial and final configurations, so that we know the size of the coin at its destination. For the case of congruent coins the set of possible destinations for each coin is all destinations. For the case of coins with a variety of sizes, the set of possible destinations for a coin of diameter d is the set of all destinations of diameter d.

In some of our upper bound arguments we need to use intermediate moves that are very far from both the initial position and the final destination. This motivated us to examine cases where moves are confined to a smaller area. We enumerate the different confining assumptions that are used.

Unbounded No bounds placed on moves.

Narrow All n coins are of unit diameter with the union of initial and target positions lying in an $a \times b$ bounding box, where $a \geqslant n$ and $b \geqslant 1$. We confine moves to the bounding box.

Wide Coins are of various diameter with value D representing the sum of the diameters. The union of the initial and target positions lie on an $a \times b$ bounding box, where $a \geqslant D$ and $b \geqslant D$. We confine moves to the bounding box.

TooTight All n coins are of unit diameter with the union of the initial and target positions lying in an $a \times b$ bounding box. The bounding box itself may be too tight to allow sufficient movement, so we confine the moves in a small box of dimension $\lceil a \rceil \times (b + \lceil n/a \rceil)$.

Using the descriptors for confining moves Table 1 summarizes our combinatorial results.

We also explore some computational complexity issues related to coin moving. We consider the problem of deciding whether we can move each coin directly to its destination in a single move. For the special case where we have n coins of various sizes, there is a unique destination for every coin and the source and destination do not overlap we have a $O(n^2)$ algorithm. Actually we develop an output sensitive algorithm that may be more efficient in certain cases,

Table 1
This table lists combinatorial results that we have obtained

Diameter	Confining assumption	Necessary	Sufficient
Unit	Unbounded	$\lfloor 8n/5 \rfloor$	$2n-1$
Various	Unbounded	$2n$	$2n$
Unit	Narrow	$\lfloor 8n/5 \rfloor$	$3n$
Various	Wide	$2n$	$4n$
Unit	TooTight	$\lfloor 8n/5 \rfloor$	$6n$

the details will be given in Section 3. At the other end of the spectrum if we allow the set of possible destinations to be at least two per coin then we show that deciding whether there is an itinerary of cost n is NP-complete.

We also consider the coin placement problem, that is, we do not have an image of the final configuration, just the coins centers and we need to determine whether the set of destinations can accommodate all of the coins without overlap. We show that deciding a non-overlapping coin placement is NP-complete.

2. Upper and lower bounds

In this section we determine bounds on the number of necessary and sufficient moves needed to produce a valid itinerary.

Consider a set of n coins with sources P and destinations Q. We assume we are given an image of both the initial and final configurations, so that we know the size of the coin at its destination. By structuring the problem in this way we avoid having to determine a feasible placement of the coins. As it is shown in Section 4, simply deciding whether a set of coins of various sizes can be centered at a set of destination points is NP-complete.

Throughout this section, (x_i, y_i) and (x'_i, y'_i) will be the coordinates of sources p_i and destinations q_i, respectively, and, without loss of generality, all the coordinates will be positive. Notice that, if a valid itinerary is found for moving the coins from P to the destinations Q, then reversing the process we have a valid itinerary for moving the coins from Q to the destinations P. Using this reasoning, *we move the destination coins*, meaning that, in fact, we do the reverse moves.

For two distinct sources p_i and p_j we say that p_i precedes p_j in a lexical ordering whenever $x_i = x_j$, and $y_i < y_j$, or $x_i < x_j$. We lexically sort the n sources. Without loss of generality, we assume a labeling that has the coins $c_1, \dots, c_n$ in lexical order. Then, the following lemma holds.

Lemma 1. *There exists an $\varepsilon > 0$ such that for all σ, $0 \leqslant \sigma \leqslant \varepsilon$, c_n can be moved to infinity in the positive y-coordinate half-plane following the line passing through the center of c_n and that forms an angle σ with the x-axis.*

Proof. Let r_n denote a ray tangent to the top of c_n and pointing to the right. Observe that r_n does not intersect the interior of any other coin because c_n is lexicographically last. Now rotate the coin c_n and the ray r_n counterclockwise about the center of c_n until the r_n meets another coin c. Let ε denote the angle of rotation, see Fig. 1. Observe that $\varepsilon > 0$, because c_n is lexically the last coin. Then, for all σ, $0 \leqslant \sigma \leqslant \varepsilon$, c_n can be moved to infinity on the ray emanating from the center of c_n that forms an angle σ with the x-axis. □

Obviously, as the coordinate axes can be rotated, once a moving direction is established, there always exists a coin that can be moved to infinity in this direction. Notice that this property holds even if tangent coins are allowed.

Lemma 2. *There is an itinerary of cost $2n$ for any configuration of coins.*

Proof. Without loss of generality, we can assume that the largest diameter of any coin is 1. Let $Y = \max\{y_1, \dots, y_n, y'_1, \dots, y'_n\}$.

By the previous lemma, c_n can be moved to a point at infinity either horizontally or with an angle less than ε_n; then c_{n-1} can be moved either horizontally or with an angle less than ε_{n-1}, and so on. Let us choose $\varepsilon = \min\{\varepsilon_1, \dots, \varepsilon_n\}$.

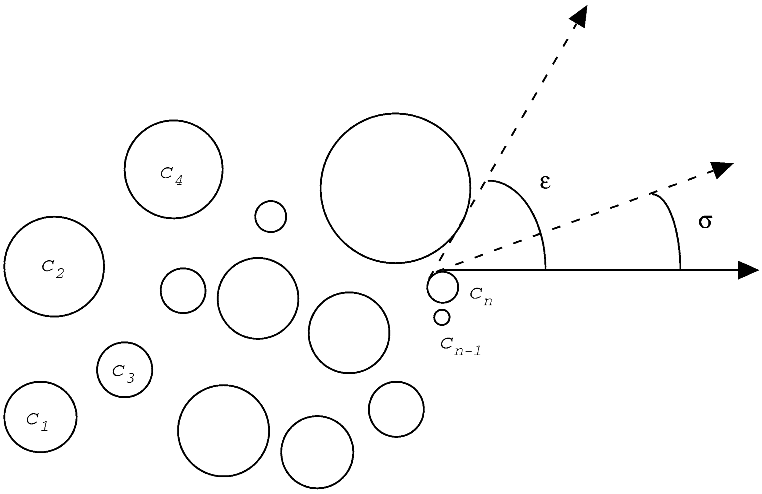

Fig. 1. Coin c_n can be moved to infinity.

The key observation is that, for a sufficiently large value M, we can always move $c_1, \ldots, c_n$, in such a way that the coins can be placed at the positions $(M, Y+2), (M, Y+4), \ldots (M, Y+2n)$ in any order, by first moving c_n, then c_{n-1} and so on.

In fact, we can choose any value M satisfying the following conditions:

• $\arctan \frac{Y+2n-y_i}{M-x_i} < \varepsilon$, for all i, (this assure that any coin can be moved to $(M, Y+2j)$, for all j, using an angle less than ε).

• $y_i - (Y+2n) > \sqrt{3}(x_i - M)$, for all i. (Supposing that we have the biggest coin located at position $(M, Y+2n)$ and a copy of it at position $(M, Y+2n-2)$, the equation $y - (Y+2n-1) = \sqrt{3}(x-M)$ corresponds to the line tangent to these two coins with positive slope. This condition assures that there are no collisions when we move the coins to their aligned positions, because all the coins are completely on the upper half-plane defined by this line.)

The previous process can also be applied to the coins starting at their destinations, and suitable angles and M' exist to move the coins from their destinations to positions $(M', Y+2), (M', Y+4), \ldots (M', Y+2n)$ in any order, by first moving c'_n, then c'_{n-1} and so on.

By choosing $M'' = \max\{M, M'\}$, we can move c_1 to $(M'', Y+2)$, c_2 at $(M'', Y+4)$ and so on, by moving first c_n, then c_{n-1}, and so on. Then, we can reverse the process to move the coins from the aligned positions to the destinations.

The number of moves that we use is $2n$. □

Consider the case now where we have a set of congruent discs and every disc can move to any of the destinations, that is, $a_i = Q$, for all i. The following result follows from our previous lemma.

Corollary 1. *There is an itinerary of cost* $2n-1$ *for any configuration of congruent coins.*

Proof. We can use the almost same moving strategy as in the proof of Lemma 2. The only exception is that c_1 can go directly to the destination c'_1 in a single move. □

Note that we can implement the strategy of the previous lemma in $O(n)$ time, after the coins are lexically ordered. Moreover, $2n$ moves is a tight bound as is shown in the following lemma.

Lemma 3. *There are configurations of n coins of various diameter that require* $2n$ *moves for a valid itinerary.*

Proof. Fig. 2 shows two different coins c_1 and c_2 tangent to the same point of an horizontal line, L. The destinations coins are also tangent in a common point on the same horizontal line, but in the reverse order. If there exists an itinerary using less than 4 moves, then at least one of the coins goes to its destination in a single move. Without loss of generality suppose it is the coin c_1, since choosing c_2 results in a similar symmetric argument. Observe that before c_1

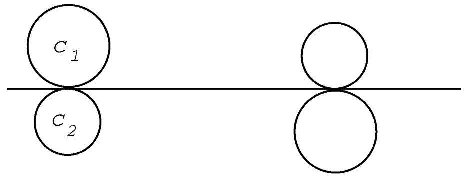

Fig. 2. A configuration of 2 coins that requires 4 moves.

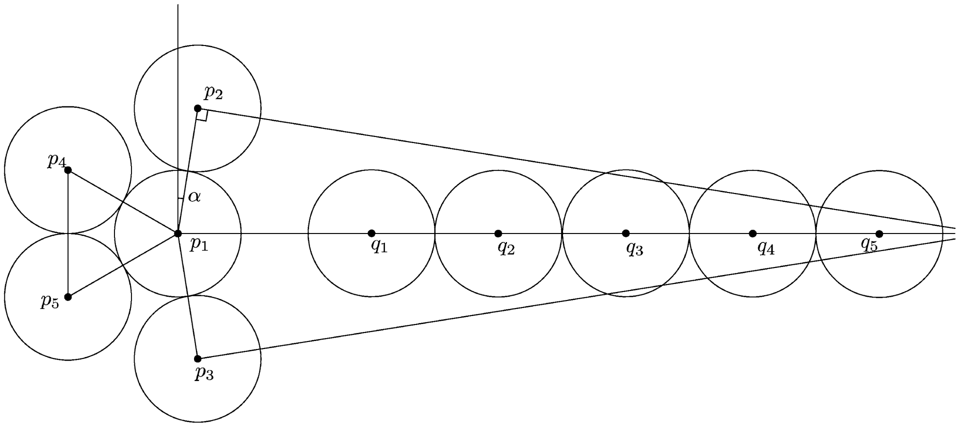

Fig. 3. Configuration of 5 congruent coins that needs 8 moves for a valid itinerary.

can go directly to its destination, c_2 must be moved out of the way. In doing this c_2 can only be placed in the half-plane below L. After moving c_1 to its destination, any trajectory for moving c_2 from below L directly to its destination is blocked. Thus c_2 must be moved to a position above L before it can be moved to its destination. Thus if c_1 moves directly to it destination in a single move c_2 needs at least three moves: one move to the lower half-plane, another move from the lower half-plane to the upper half-plane, and a third move to its destination. Hence we need at least four moves even if c_1 moves directly to its destination in a single move.

A configuration formed by n copies of the previous figure, each pair and its destinations tangent on an arbitrary common line in opposite orders, needs $4n$ moves. □

We remark in passing that the previous result uses the fact that the source and destination positions come in tangent pairs. If tangent coins are not allowed, then the best example we have shows that n coins need only $2n - 1$ moves for a valid itinerary. It would be interesting to settle the question of whether one can force $2n$ moves without using tangencies.

For congruent coins and $a_i = Q$, for all i, we have the following result.

Lemma 4. *At least $\lfloor 8n/5 \rfloor$ moves are needed to move a set of n coins to their destinations.*

Proof. First we show that there exists a set of 5 coins so that at least 8 moves are needed to move them to their destination. The sources P of the coins are given as follows. Let the $p_1 = (0, 0)$ and $p_2 = (2\sin\alpha, 2\cos\alpha)$. We choose α small enough so that the centers of 5 destination coins can be located on the x-axis below the line passing through p_2 with angle $-\alpha$. The centers p_1, p_4, p_5 are the vertices of the regular triangle with side length 2 and p_4p_5 vertical, see Fig. 3.

A coin is *good* if it makes just one move, otherwise the coin is *bad*.

We prove that there are at least 3 bad coins. Clearly, either c_1 or c_2 (or both) is bad. Similarly, either c_1 or c_3 is bad. Also, either c_4 or c_5 is bad.

If $\{c_1, c_2, c_3\}$ contains at least 2 bad coins then the claim holds. Suppose that $\{c_1, c_2, c_3\}$ contains only one bad coin (no bad coins is impossible). It should be c_1. At the time of the first move of c_1 the coins c_2 and c_3 are at the initial

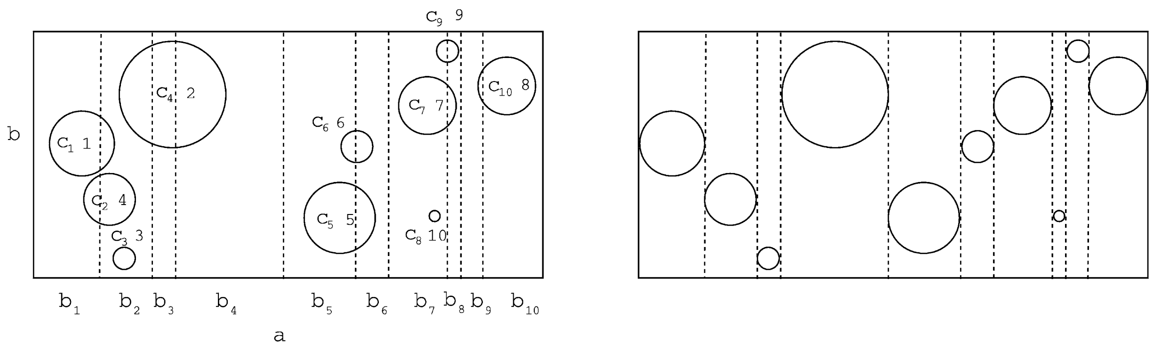

Fig. 4. Separating the coins horizontally with n moves.

positions. Therefore, c_2 moves by (x, y) such that $x < 0$. Then c_4 and c_5 must have already been moved which means that they are bad. Thus the total number of bad coins is 3.

We prove the claim when n is a multiple of 5 by repeating the construction for 5 coins as follows. We place $n/5$ groups of coins by shifting 5 coins horizontally so that they do not overlap. The shifting vector can be $(-4, 0)$ for example. The destination positions are placed to the right of the source positions on the x-axis. The angle α is chosen so that the centers of all destination coins are below the line with angle $-\alpha$ as defined by every group of 5 coins. This can be verified by simply checking the leftmost group of coins with the rightmost destination positions. The argument above generalizes for this construction.

Now consider the case when $n \equiv m \pmod 5$ where $1 \leqslant m \leqslant 4$. Starting with the construction for $5\lfloor n/5 \rfloor$ place $\lfloor m/2 \rfloor$ pairs of coins as the pair c_1, c_2 in Fig. 3. Every such pair requires 3 moves. If m is odd then place one coin anywhere in the plane. We place all additional destinations to the right of the others on the x-axis. The total number of bad moves is $3\lfloor n/5 \rfloor + \lfloor m/2 \rfloor$. Thus, the total number of moves is $n + 3\lfloor n/5 \rfloor + \lfloor m/2 \rfloor = \lfloor 8n/5 \rfloor$. □

2.1. Confined workspaces

For the previous results we were able to move coins arbitrarily far to obtain our upper bounds. If the coins are confined to a smaller workspace, we need to apply different strategies. Let us assume that we have n coins so that the union of the sources and destinations lie in an $a \times b$ bounding box. Without loss of generality we assume that $a \geqslant b$. Let $d_1, d_2, \ldots, d_n$ denote the diameters of the coins and let $D = \sum d_i$.

Now, let us assume that we have divided a box B of size $D \times b$ into n non-overlapping boxes from left to right, in such a way that box b_1 has size $d_1 \times b$, the second box b_2 has size $d_2 \times b$, and so on. Then the following lemma holds.

Lemma 5. *We can translate each coin i to its corresponding box b_i with a total of n horizontal moves.*

Proof. Let (x_i'', y_i) be the center of coin i if it is moved horizontally to lie inside b_i. We classify the coins into two classes. A coin i will be type $-$ if $x_i \leqslant x_i''$ (the source is on the left of the box b_i), and type $+$ otherwise.

If we examine the coins from left to right, we obtain a sequence of minuses and pluses according to the types of the coins. If the sequence starts with a plus, then we can move the first coin horizontally to the left until it reaches box b_1. Now recursively solve a problem with $n - 1$ coins and $B' = b_2 \cup \cdots \cup b_n$.

If the sequence starts with a minus, then we follow the sequence until a plus appears. If this run of minuses has size i, we move coin i horizontally to the right until it reaches box b_i (there are no collisions because there is a plus in position $i + 1$), then we move coin $i - 1$ to the right until it reaches b_{i-1}, and so on. When the first i coins are located in their corresponding boxes, we continue the process with a problem with fewer coins and a smaller box. □

Fig. 4 shows an example of the lemma with $a = D$. The table given below illustrates how pluses and minuses are used to obtain an ordered list of moves for the example of Fig. 4.

Coin	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}
Type	+	−	−	−	+	−	+	−	−	−
Order	1	4	3	2	5	6	7	10	9	8

In this example, first, c_1 is moved to the left to put it inside b_1, then c_4, c_3, c_2 are moved to the right in this order to put them inside b_4, b_3, b_2, respectively, then c_5 is moved to the left to put it inside b_5, and so on.

If the coins are ordered then only $O(n)$ time is used to implement the algorithm from the previous lemma.

In the following, we will only describe the processes to obtain the number of moves desired, and not the correct order in which the coins must be moved (in most of the situations, this order is evident). In addition, we will use the terms "source coin" and "destination coin" to mean that the coin is at the source and that the coin is at the destination, respectively.

Corollary 2. *Given n unit coins, with sources and destinations lying in the confines of an $a \times b$ confining box, if $a \geqslant n$, $b \geqslant 1$ and $a_i = Q$, then we can always determine a valid itinerary of cost at most $3n$.*

Proof. We can apply the previous lemma to the source coins and to the destination coins. This leaves one source coin and one destination coin in each box b_i. The sequence of moves sources to their correct box, followed by a move within each box of a source to its destination, and finally the reverse application of the lemma of the destination coin inside its box to its actual destination location. □

Corollary 3. *Given n coins of various diameters, with sources and destinations lying in the confines of an $a \times b$ confining box, if $a \geqslant D$, $b \geqslant D$ and $a_i = q_i$, then we can always determine a valid itinerary of cost at most $4n$.*

Proof. We use $2n$ horizontal moves and $2n$ vertical moves. Suppose that the coins $c_1, \dots, c_n$ have been horizontally separated to positions $(\bar{x}_1, y_1), \dots, (\bar{x}_n, y_n)$ using Lemma 5, and in the same way the destination coins have been vertically separated to the positions $(x'_1, \bar{y}'_1), \dots, (x'_n, \bar{y}'_n)$. If the destination of the coin c_i is $(x'_{\pi(i)}, y'_{\pi(i)})$, then, first move each coin $(\bar{x}_i, y_i)$ vertically to $(\bar{x}_i, \bar{y}'_{\pi(i)})$ and then, move each coin $(\bar{x}_i, \bar{y}'_{\pi(i)})$ horizontally to $(x'_{\pi(i)}, \bar{y}'_{\pi(i)})$. □

If the size of the box is too small, the coins may be blocked. Therefore, in order to move the coins, we have to allow moves in a bigger box. For this situation, we have the following result.

Corollary 4. *Given n unit coins, with sources and destinations lying in the confines of an $a \times b$ confining rectangle, if we allow moves in an $\lceil a \rceil \times (b + \lceil n/a \rceil)$ confining box, then we can always determine a valid itinerary of cost $6n$.*

Proof. As the size of the confining box is $\lceil a \rceil \times (b + \lceil n/a \rceil)$, we have added at the top of the $a \times b$ rectangle $\lceil n/a \rceil$ empty rows of size $\lceil a \rceil \times 1$.

Let us assume that we have sorted the coins in non-increasing order according to their y-coordinates. Then, we process the first $\lceil a \rceil$ coins by moving them vertically upwards, then horizontally separating them by applying Lemma 5, and finally moving them vertically upwards until they reach the top row of the box. So, doing $3\lceil a \rceil$ moves, the first $\lceil a \rceil$ coins are placed in the top row.

We process the second $\lceil a \rceil$ coins putting them into the second row of the box, and so on. At the end, we have located all the coins into the $\lceil n/a \rceil$ empty rows. We can ensure that there are no collisions when using this process by applying an inductive argument.

As usual, the same process can be used to put the destination coins into the $\lceil n/a \rceil$ empty rows. Hence, by reversing this second process, we can move the n coins from the sources to the destinations using $6n$ moves. □

3. Decision problems

In this section we consider the problem of deciding whether there is an itinerary of cost at most n for n coins.

In the first instance consider the case where each coin has a single possible destination. Observe that unless each coin has a distinct destination there is no valid itinerary. Thus without loss of generality, we can designate the destination for coin i d_i, that is, $a_i = \{d_i\}$. Furthermore, a valid itinerary exists only when no two destinations overlap.

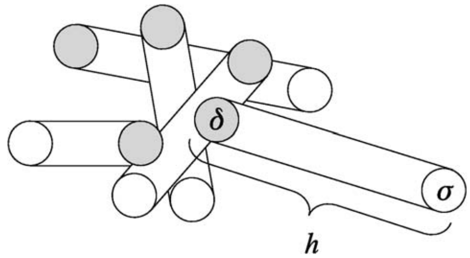

Fig. 5. The hippodromes corresponding to coin trajectories. The shaded discs represent destinations. We have labeled one hippodrome to illustrate the areas h, σ, and δ.

Our algorithm begins by constructing an outline of the trajectory of each coin from its source location to its destination. This outline takes the shape of a racetrack or *hippodrome*, thus we use h_i to denote the area of the hippodrome for the trajectory of coin i from s_i to d_i. Observe that the geometry of a hippodrome is the union of a rectangle and two discs. Let σ_i and δ_i, respectively, denote the area of h_i that contains c_i when it is in it's source and destination position. See Fig. 5 for an illustrative example.

Our strategy will be to construct a directed graph G. Each vertex in G corresponds to a coin, and a directed edge (i, j) will be used to specify that coin i must be moved before coin j. It remains to show how G is constructed.

For every ordered pair of coins c_i, c_j we assign directed edges as follows: (i, j) if σ_i intersects h_j because c_i must move before c_j, and (j, i) if δ_i intersects h_j because c_j must move before c_i.

Theorem 1. *There is an itinerary of cost at most n if and only if G does not contain a directed cycle.*

Proof. Suppose there are no directed cycles in G. We can begin the itinerary with all coins that correspond to vertices in G with no incoming edges. We can now remove these vertices from G and find a new set of vertices with no incoming edges. This process can be repeated until all of the coins have been moved.

On the other hand, suppose that G does contain a directed cycle. Let $v_{\gamma_1}, v_{\gamma_2}, \ldots, v_{\gamma_k}, v_{\gamma_1}$ denote the shortest directed cycle in G. The resulting conundrum is that v_{γ_1} must before v_{γ_k} and v_{γ_k} must move before v_{γ_1}, and that clearly is impossible. □

We can compute all of the intersections and construct G in $O(n^2)$ time. We can use an output sensitive algorithm to compute the intersections in $O(n \log n + k)$ time where k denotes the number of intersections [1,2]. This value k is also proportional to the number of edges in G. We can traverse G and determine whether there are directed cycles also in $O(k)$ time. Thus the complexity of our algorithm is $O(n \log n + k)$. Note that this approach is similar to Buckley's results on coordinating the motion of multiple robots [3].

We now show that a similar problem is intractable. Our results are related to the general multi robot path planning which is known to be intractable [7,8]. In this version of our problem we put no restriction on the size of a_i, it may contain an arbitrary number of destinations. Thus we have:

One Move per Coin (OMC)
Instance: A set of coins, sources, destinations and possible destinations for each coin.
Question: Can the coins be moved from their sources to destinations with an itinerary that uses at most one move per coin?

Our reduction will be from the following problem which Plesník [12] proved to be NP-complete.

Hamilton Path in Directed Bipartite Graph (HPDBG)
Instance: A directed bipartite graph (V, E) where the vertices in V are labeled $1, \ldots, n = 2k$. Vertices with odd labels and have out-degree 2 and in-degree 1, except for vertex 1 which has out-degree 2 and in-degree 0. Vertices with even labels have out-degree 1, and in-degree 2, except for vertex 2 which has out-degree 0 and in-degree 2.
Question: Is there a directed Hamilton path in G starting at vertex 1 and ending at vertex 2.

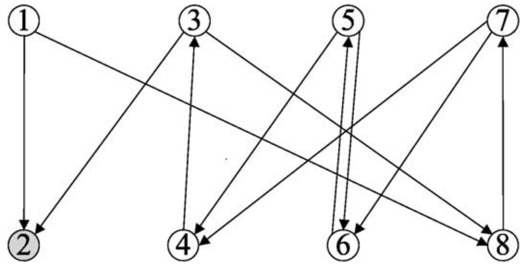

Fig. 6. There are coins at every position except 2. Observe that we can move the coins with an itinerary: 3 → 2, 4 → 3, 5 → 4, 6 → 5, 7 → 6, 8 → 7, and 1 → 8. This itinerary traces a Hamilton path from 1 to 2 in reverse.

Suppose we have an instance of HPDBG, then for every vertex i in G, except for vertex 2, we use a coin c_i. The coins are positioned in two rows, spaced out sufficiently, in a way one would normally draw a bipartite graph. The possible destinations for the coin c_i corresponds to the outgoing neighbours of vertex i, as is illustrated in Fig. 6. More precisely, let $G = (V, E)$ with $V = \{1, 2, \dots, n = 2k\}$. Now consider a set of points $S = \{(\delta i, l_1), (\delta i, l_2)\colon i \in \{1, \dots, k\}$ and δ, l_1 and l_2 are three suitably defined constants$\}$. Now we have $n - 1$ coins with sources at the points in S excepting the point (δ, l_2). The destinations also come from the set S except for the point (δ, l_1). In this way we construct an instance of OMC from HPDBG in time proportional to the size of G. It is routine to verify that a Hamilton path in G corresponds to an itinerary using one move per coin. Observe that the first move, of any itinerary that uses $n - 1$ moves, must move one of two coins to the destination at point (δ, l_2). This observation leads to an inductive argument showing that any itinerary consisting of $n - 1$ moves in the given instance of OMC corresponds to a Hamilton path in the graph G.

The preceding discussion leads us to conclude with the following theorem:

Theorem 2. *OMC is polynomial reducible from HPDBG, thus OMC is NP-complete.*

This result shows the remarkable difference between the coin moving problem with one destination per coin compared to the case where a fixed fraction of the coins have a choice between two destinations.

4. Placing coins

Consider a set, S, of n coins of various diameters and a set, P, of n destinations points in the plane. Is there a way to place the coins so each coin is centered at a point of P and no two coins overlap? Let us call this decision problem the Coin Placement Problem (CPP).

We show that CPP is NP-complete by reducing it from a variant of 3SAT. In this variant, named 1-in-3SAT, we insist that each clause has exactly one variable set to true and two variables set to false. With this added constraint we do not need to consider negations of variables [6]. Thus an instance of 1-in-3SAT consists of a set V of n boolean variables, and a set C of clauses of the disjunction of three literals, each referring to a variable in V. We then ask, is there an assignment of truth values to the variables such that the conjunction of the clauses is true, and no clause has more than one variable set to true?

Given an instance of 1-in-3SAT we construct an instance of CPP. We will use some gadgets to make sure that there is a valid placement if and only if there are assignments that satisfy the instance of 1-in-3SAT. There are two types of gadgets. For each clause we have a *clause gadget* that ensures that each clause has exactly one true literal. For each variable we have a *consistency gadget* to ensure that the truth assignments over all literals for that variable are consistent.

At this point some readers may benefit by consulting the example shown in Fig. 9 to obtain an overview of the completed construction.

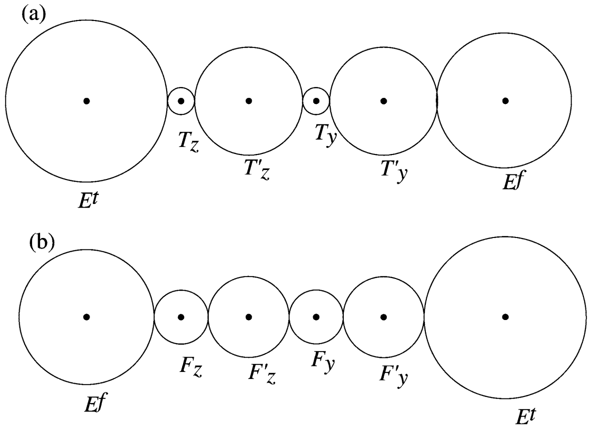

Fig. 7. We see two ways to arrange the coins in a consistency enforcing gadget. In (a) the coins are arranged in a true sequence, that is they represent setting two literals z and y of the same variable to true. In (b) the arrangement of coins represents setting the same literals to false.

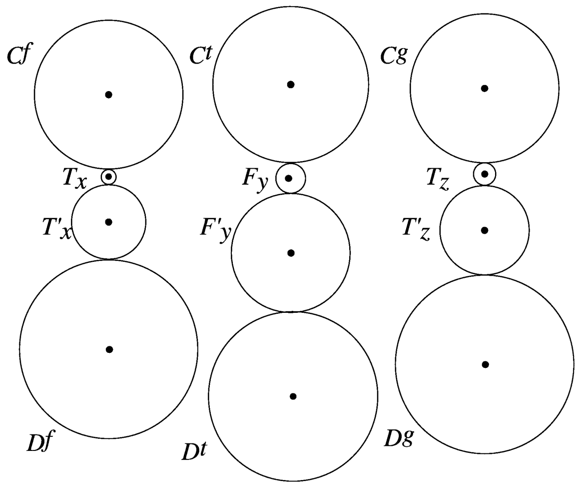

Fig. 8. A clause gadget. In this clause the literal y is set to true, and both x and z are set to false. Note: The coins labeled F represent true and T represent false when they are used in the clause gadget.

4.1. Construction details

For each literal we use four coins which we label T, F, T', and F'. The basic strategy is to use T and T' in either the clause gadget or the consistency gadget, and F and F' in the other gadget. These pairs of coins can be arranged in one of two ways, representing an assignment of true or false.

The consistency gadget uses two of its own coins E^t and E^f. The placement points of this gadget accommodate the coins in exactly two ways, one representing setting the variable $v = \texttt{true}$ and the other $v = \texttt{false}$. See Fig. 7.

The clause gadget has three pairs of coins of its own C^t, D^t, C^f, D^f, and C^g and D^g. The first pair fits with coins that represent a true assignment and the second two pairs fit with coins that represent false. See Fig. 8.

The placement points of the consistency enforcing gadget are collinear, and spaced using three distinct distances, d_1, d_2 and d_3. The truth enforcing gadget may also be laid out on a single line, however, in the interest of clarity, we use a layout using three distinct lines in our example. Let $r(F_i)$ (and similarly for other coins) denote the radius of a coin that is used to represent variables v_i. We set the radii so that the following equations hold.

$$r(F_i) + r(F_i') = r(T_i) + r(T_i') = d_1. \tag{1}$$

We also have:

$$r(E_i^t) + r(T_i) = r(E_i^f) + r(F_i) = d_2, \tag{2}$$

$$r(E_i^t) + r(F_i') = r(E_i^f) + r(T_i') = d_3. \tag{3}$$

For each literal within a clause j we place coins spaced using three distances as well. The distance d_1 in Eq. (1) is used, as well as:

$$r(C_j^t) + r(F_i) = r(C_j^f) + r(T_i) = r(C_j^g) + r(T_i) = d_4, \tag{4}$$

$$r(D_j^t) + r(F_i') = r(D_j^f) + r(T_i') = r(D_j^g) + r(T_i') = d_5. \tag{5}$$

The differences between true and false coins is a common value ε, that is: $\varepsilon = r(F) - r(T) = r(T') - r(F') = r(E^t) - r(E^f) = r(C^f) - r(C^t) = r(C^g) - r(C^t) = (D^t) - r(D^f) = r(D^t) - r(D^g)$.

Given an instance of 1-in-3SAT we can construct an instance of CPP that has a valid placement if the instance of 1-in-3SAT is satisfiable. To show that every valid placement of an instance of CPP I_C corresponds to a satisfiable instance of 1-in-3SAT, we set the sizes of the coins so that the distances d_i for $i = 1, \ldots, 5$ are unique for each variable and clause. Let $N = \max(|C|, |V|) + 1$. We will represent the radii of the coins using positive integers in base N. For the variable v_i we have:

$r(T_i) = \mathtt{0000i0}_\mathtt{N}$,
$r(F_i) = \mathtt{0000i1}_\mathtt{N}$,
$r(T_i') = \mathtt{000i01}_\mathtt{N}$,
$r(F_i') = \mathtt{000i00}_\mathtt{N}$,
$r(E_i^t) = \mathtt{00i001}_\mathtt{N}$,
$r(E_i^f) = \mathtt{00i000}_\mathtt{N}$.

And for the clause c_j:

$r(C_j^t) = \mathtt{0j0000}_\mathtt{N}$,
$r(C_j^f) = \mathtt{0j0001}_\mathtt{N}$,
$r(C_j^g) = \mathtt{0j0001}_\mathtt{N}$,
$r(D_j^t) = \mathtt{j00001}_\mathtt{N}$,
$r(D_j^f) = \mathtt{j00000}_\mathtt{N}$,
$r(D_j^g) = \mathtt{j00000}_\mathtt{N}$.

Now to specify the points comprising the gadgets. We start with the consistency gadget. Suppose there are t occurrences of the variable i in the given instance of 1-in-3SAT. We need a total of $2t + 2$ points. Since the points are collinear we only use a single number to specify the points. Specifying point one as the constant $p(i, 1)$, the second point $p(i, 2) = p(i, 1) + \mathtt{00i0i1}_\mathtt{N}$. The next $2t - 1$ are given by the expression $p(i, k) = p(i, k - 1) + \mathtt{000ii1}_\mathtt{N}$. The final point is at $p_{2t+2} = p_{2t+1} + \mathtt{00ii01}_\mathtt{N}$. For each clause gadget we have three groups of four points, one for each variable in the clause. Let $q(j, l, 1)$ now represent the first point used for the lth variable v_i in a clause c_j. The subsequent three points are given by: $q(j, l, 2) = q(j, l, 1) + \mathtt{0j00i1}_\mathtt{N}$, $q(j, l, 3) = q(j, l, 2) + \mathtt{000ii1}_\mathtt{N}$ and $q(j, l, 4) = q(j, l, 3) + \mathtt{j00i01}_\mathtt{N}$.

Lemma 6. *Given an instance of I_S, 1-in-3SAT we can construct an instance I_C of CPP in polynomial time that has a valid placement whenever I_S is satisfiable.*

Proof. It is a routine matter to place the coins once satisfiable truth assignments are given. □

In Fig. 9 we give an illustration of our construction given the 1-in-3SAT instance (x, y, z), (x, y, w), and the truth assignment $y =$ **true** and $x = z = w =$ **false**. Note that for practical reasons the coins are not drawn to scale.

To show that every valid placement of I_C corresponds to a satisfiable truth assignment of I_S we make use of the distinct sizes of coins and spaces between the coins.

We begin by showing that the coins C and D that we use in the clause gadgets are forced to go there.

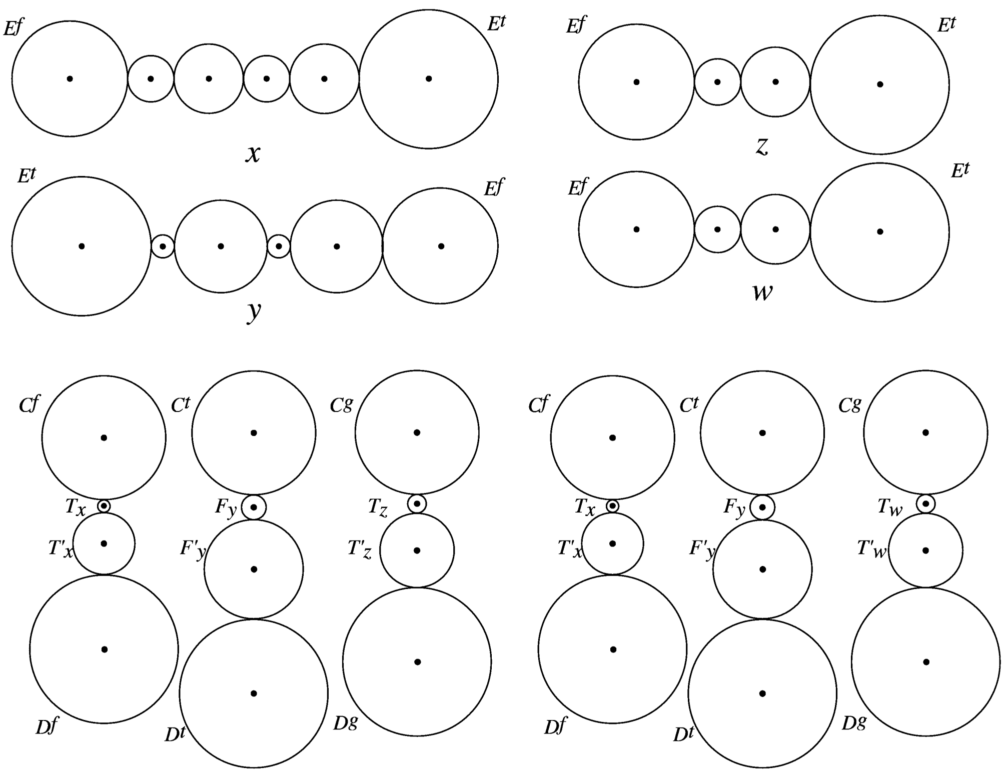

Fig. 9. A placement of coins that corresponds to the 1-in-3SAT instance $(x, y, z), (x, y, w)$, and the truth assignment $y =$ **true** and $x = z = w =$ **false**. Note that this representation does not assign distinct sizes to the coins as specified by the construction. Using sizes as specified would require a drawing that is too big to be practical. We also have altered our notation in the figure to better suit the example. The intended meaning should be clear.

Lemma 7. *In any valid placement of coins the coins C_j^t, C_j^f, C_j^g must be placed at one of the points $q(j,1,1), q(j,2,1)$ and $q(j,3,1)$ and the coins D_j^t, D_j^f and D_j^g must be placed at one of the points $q(j,1,4), q(j,2,4)$ and $q(j,3,4)$.*

Proof. Let m denote the number of clauses. Due to their size, a radius of at least $\mathtt{m00000}_{\mathrm{N}}$, the only placement of the coins D_m^t, D_m^f and D_m^g is at one of the points $q(m,1,4), q(m,2,4)$ and $q(m,3,4)$. Otherwise these coins overlap two or more placement points. Once these coins are placed the remaining available locations forces the placement of D_{m-1}^t, D_{m-1}^f and D_{m-1}^g at one of the points $q(m-1,1,4), q(m-1,2,4)$ and $q(m-1,3,4)$. Continuing in the same way, we must place the coins D_j^t, D_j^f and D_j^g at one of the points $q(j,1,4), q(j,2,4)$ and $q(j,3,4)$. Similarly the coins C_j^t, C_j^f, C_j^g must be placed at one of the points $q(j,1,1), q(j,2,1)$ and $q(j,3,1)$. □

The consistency gadget coins E^t and E^f are also forced into place by a similar argument. Thus:

Lemma 8. *In any valid placement of coins the coins E_i^t and E_i^f must be placed at the points $p(i,1)$ and $p(i,2t+2)$ where the variable v_i appears t times in the instance of I_S.*

Proof. Again due to the sizes once the coins D and C are placed we force the placement of the coins E_n^t and E_n^f where n denotes the number of variables in I_S. The subsequent coins are forced in a similar manner. □

We are left with the coins representing the variables themselves that is T_i, F_i, T_i' and F_i'. We call a placement of the coins, T_i, F_i, T_i' and F_i', *tight* if each of these coins are incident to exactly two others. Observe that the placement we obtain from a satisfiable truth assignment is tight. Once the coins C, D E^t and E^f are placed we see that all valid placements must be tight. This remark is formalized by the next lemma.

Lemma 9. *Every valid placement uses a tight placement of T_i, F_i, T_i' and F_i', in either the true position or the false position. Thus every valid placement of I_C corresponds to a satisfiable assignment of I_S.*

Proof. The placement corresponding to a satisfiable truth assignment is tight. Thus the linear measure of the coins and the spaces remaining are exactly equal. We show that this implies that every placement must be tight.

Consider the consistency gadget for the variable v_i, where v_i appears t times.

First consider placing E_i^t at $p(i,1)$ then the gap between the edge of E_i^t and $p(i,2)$ is of length $\mathtt{0000i0}_{\mathtt{N}}$ which is exactly the radius $r(T_i)$. This forces a placement of the coins in a true sequence with E_i^f at $p(i,2t+2)$. Now we have t copies of the coins F_i and F_i' that need to be placed. Without loss of generality assume that there is an occurrence of the variable v_i as the first literal in clause c_j. A tight placement forces C_j^t at $q(j,i,1)$, D_j^t at $q(j,i,4)$ and the coins F_i and F_i' at $q(j,i,2)$ and $q(j,i,3)$, respectively.

On the other hand, assume that E_i^f is placed at $p(i,1)$. This forces the coins in a false sequence. Furthermore the t copies of T_i and T_i' must be placed in their appropriate places within their clause gadgets.

Therefore we conclude that if we have a valid placement of the coins I_C then we can assign truth values satisfying I_S. □

We conclude with the main theorem of this section.

Theorem 3. *CPP is NP-complete.*

Proof. Observe that it is possible to determine whether a placement of coins is valid, in polynomial time, thus the problem CPP is in NP. By the arguments in Lemmas 6 and 9 we can conclude that the NP-complete problem 1-in-3SAT is polynomial reducible from CPP. Therefore CPP is NP-complete. □

Note: our construction can be laid out so that all coin centers lie on a common line.

Alberto Márquez [11] has proposed an alternate proof to show that CPP is NP-complete. The reduction is to planar 3-SAT and uses coins with only two different sizes. The construction is closely related to that of Forman and Wagner in their paper on map labeling [5].

5. Discussion

We have presented some combinatorial and algorithmic results on moving coins in the plane. There are several issues that remain unresolved, and we conclude by briefly summarizing them.

The number of moves required to satisfy an itinerary of n unit coins in an unrestricted work space has a lower bound of $\lfloor 8n/5 \rfloor$ and an upper bound of $2n-1$. It would be interesting to close this gap.

In terms of complexity we have shown that some decision problems related to moving coins are hard. The complexity of determining the optimum number of moves to satisfy the itinerary of a set of unit coins remains open.

Acknowledgements

Ferran Hurtado is partially supported by projects DURSI 2001SGR00224 and MCYT BFM2003-0368. Alfredo García Olaverri and Javier Tejel are partially supported by project DGA 228-61. David Rappaport is partially supported by NSERC of Canada Discovery Grant 9204.

References

[1] I. Balaban, An optimal algorithm for finding segment intersections, in: Proc. 11 Annu. ACM Sympos. Comput. Geom., 1995, pp. 211–219.

[2] J.-D. Boissonnat, J. Snoeyink, Efficient algorithms for line and curve segment intersection using restricted predicates, Computational Geometry 16 (1) (2000) 35–52.

[3] S.J. Buckley, Fast motion planning for multiple moving robots, in: Proc. of IEEE Int. Conf. on Robot's and Automation, 1989, pp. 322–326.
[4] E. Demaine, M. Demaine, H. Verrill, Sliding coin puzzles, in: R.J. Nowakowski (Ed.), More Games of No Chance, Cambridge University Press, Cambridge, 2002, pp. 405–431. Collection of papers from the MSRI Combinatorial Game Theory Research Workshop, Berkeley, CA, 2002.
[5] M. Formann, F. Wagner, A packing problem with applications to lettering of maps, in: SCG'91: Proceedings of the Seventh Annual Symposium on Computational Geometry, ACM Press, New York, 1991, pp. 281–288.
[6] M.R. Garey, D.S. Johnson, Computers and Intractability, W.H. Freeman and Company, New York, 1979.
[7] J. Hopcroft, J. Schwartz, M. Sharir, On the complexity of motion planning for multiple independent objects pspace hardness of the warehouseman's problem, Int. J. Robotics Res. 3 (4) (1984) 76–88.
[8] J. Hopcroft, G.T. Wilfong, Reducing multiple object motion planning to graph searching, SIAM J. Comput. 15 (3) (1986) 768–785.
[9] Y. Hwang, N. Ahuja, Gross motion planning—a survey, ACM Comput. Surv. 24 (3) (1992) 219–291.
[10] V.I. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals, Soviet Phys. Dokl. 10 (1966) 707–710.
[11] A. Márquez, Lettering and covering, in: Abstracts from JCDCG 2004, the Japan Conference on Discrete and Computational Geometry, 2004, pp. 116–119.
[12] J. Plesník, The NP-completeness of the Hamiltonian cycle problem in planar digraphs with degree bound two, Inform. Process. Lett. 8 (4) (1979) 199–201.
[13] Y. Rubner, C. Tomasi, L.J. Guibas, The earth movers distance as a metric for image retrieval, Internat. J. Computer Vision 40 (2) (2000) 99–121.