J Math Chem (2014) 52:198–212
DOI 10.1007/s10910-013-0255-3

ORIGINAL PAPER

Canonical labels for protein spots of proteomics maps

Milan Randić · Rok Orel

Received: 7 May 2013 / Accepted: 19 August 2013 / Published online: 31 August 2013

Abstract We consider the problem of canonical labeling for a class of maps, which include proteomics maps, which consist of a set of vertices or protein spots. If this problem is solved and followed, different laboratories studying proteomics maps will arrive at the same numbering of spots, which would facilitate comparisons of data from different sources. In addition, the proposed canonical numberings of protein spots would allow compiling a catalog of proteomics maps just as canonical labeling allows making catalogs graphs, or molecules, and other canonically labeled systems, which would make search for similar sets of maps very efficient. We approach the problem by modifying the algorithm of Jeffrey for graphical representation of DNA based on the chaos game. Graphical representation of DNA as a chaos game map has an important property in that this representation allows one to assign sequential labels to spots in a DNA map. We have modified the approach for sequential labeling of chaos game map representations to graphical representation of any tabular data, such as listing of (x, y) coordinates of protein spots of proteomics maps.

Keywords Canonical labels · Proteomics maps · Chaos game

1 Introduction

We consider the problem of finding canonical labeling of vertices of a map, with particular interest in proteomics maps, consisting of a set of protein spots viewed as vertices

M. Randić (✉)
National Institute of Chemistry, Hajdrihova 19, Ljubljana, Slovenia
e-mail: mrandic@msn.com

R. Orel
XLAB, Pot za Brdom 100, Ljubljana, Slovenia
e-mail: rok.orel@xlab.si

of a map, which would assign to vertices of a map unique labels. Canonical labeling of vertices of maps is of interest in comparative studies of maps because identical maps will have identical vertex labels, which would facilitate their recognition. In Graph Theory canonical labeling of vertices makes possible to recognize identical graphs and thus to solve the problem of graph isomorphism [1]. The particular canonical rule that we use on graphs is based on search for numbering of vertices of a graph such that it's adjacency matrix, when it's rows are read from left to right and from top to bottom, gives the smallest binary number possible for the graph [2,3]. The resulting canonical labels allow one to determine the symmetry of graphs (or a map) [4,5] and they also allow constructions of graphs of certain classes [6,7], but these additional properties of canonical labels are of no interest for proteomics maps, which as a rule lack any symmetry. One disadvantage of above mentioned canonical rules is that finding canonical labels for larger and more complex systems could be computationally difficult.

In this article we will consider the problem of finding canonical labels for vertices of a map, such as the map illustrated in Fig. 1, where the points can represent centers of proteins spots in proteomics maps. Observe that vertices of maps have definite positional coordinates, which fully determines the geometry of such maps. Once one obtains canonical labels one can construct a binary code for such maps so that set of maps can be ordered lexicographically and be catalogued. It appears that this important problem, as far as we were able to find, has not been considered hitherto in the chemical literature. There has been one report from our own laboratory [8] in which we have assigned canonical labels to proteins spots of proteomics map of liver cells of healthy male Fisher F344 rats and the rats treated with different peroxisome proliferators. Canonical labeling of spots in this case was obtained by constructing a graph based on partial ordering, the so called Hasse diagram, which has been superimposed over the map. Thus the search for the canonical labels for a map was converted to a search for the canonical labels of a graph.

Here we will outline a different route for determining labels for a spots of a map, which is based on a property of the chaos game maps, introduced by Barnsley [9] for representation of lengthy sequences of numbers. In particular we will follow the work of Jeffrey [10], who modified the chaos game rules for representation of DNA. Let us first outline the representation of DNA sequence as 2-D map of Jeffrey. The standard representation of DNA sequence of n bases [(nucleotides adenine (A), cytosine (C), guanine (G) and thymine (T)] consist of listing sequentially nucleotides, encoded by the four letters, which appear as a random sequence of ACGT letters.

Let us illustrate the construction of 2-D DNA map of Jeffrey on a short string of 92 bases standing for the first exon of human β globin gene, which starts with: ATGGTGCACCTGAC . . . . First one draws a square and assigns to its four corners the labels A, C, G and T, as illustrated in Fig. 1. To obtain the 2-D graphical representation of this sequence, according to Jeffrey, one starts at the center of the square and moves half way toward corner assigned to the first base in the sequence, which is corner A, and arrives at the site for the first nucleotide A. From here one moves half way towards the corner T, which is the second base in the sequence, which gives the position of the second nucleotide. Again from this location one continues to move half way to the corner of the third nucleotide, which is the corner G, and finds the site within the

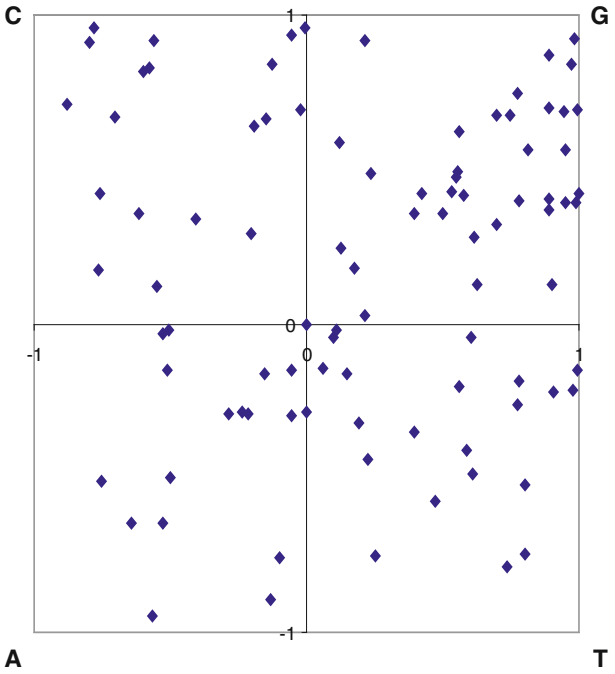

Fig. 1 Graphical representation of the first exon of human â-globin gene

square for third nucleotide in the sequence, and so on. After all 92 bases have been depicted following the above algorithm, one obtains Fig. 2.

At first sight there is nothing remarkable about this map, which appears as a collection of random points within a square, but what is significant about the map is that there is *no loss of information* about DNA in transforming the 1-dimensional sequences of letters into 2-dimensional DNA map. This means that from the constructed 2-D DNA map one can reconstruct the DNA sequence, and hence recover the sequential numbers of all nucleotides. For more about the chaos game representation of DNA we direct readers to a recent review article on graphical representation of proteins, which also elaborates on graphical representations of DNA [11].

Let us briefly outline the reconstruction of DNA from its map. Again one starts at the center of the square and looks for four sites in the square which are half way from the center and the four corners A, C, G, T. One of the four sites will indicate the position of the first nucleotide, as the vertex 1 of the map. One continues starting now from the vertex 1 and looking for four sites which are half way from vertex 1 and the four corners A, C, G, T. Again one of the four sites will be »occupied« by a nucleotide, which receive label 2. In continuation one finds the next nucleotide looking for the four sites half way from vertex 3 and the four corners and continues in the same way to find next vertex, and so on. The process ends when the last vertex is found and all spots received labels.

In this way one recovers the sequential labels for vertices of an 2-D DNA map, because they have been constructed using the same algorithm. The problem that we

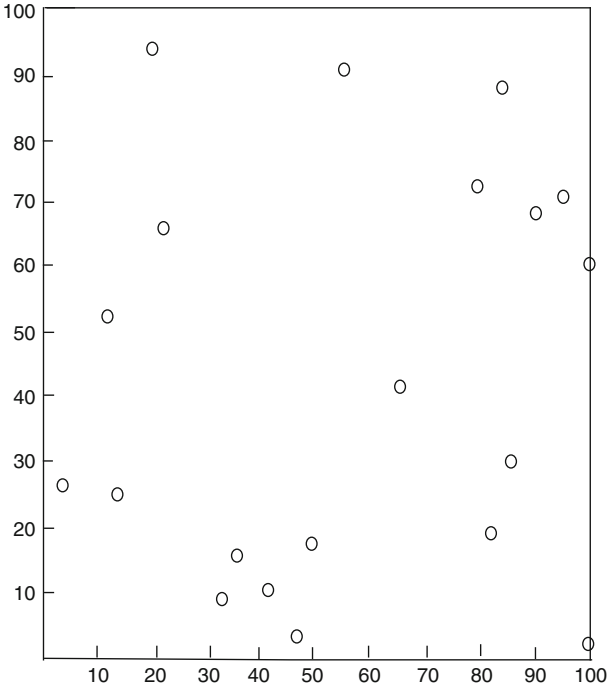

Fig. 2 Map having 20 unlabelled vertices at random positions

wish to consider is how can one find »sequential« labels for an arbitrary 2-D map and convert it to a 1-D sequence. Clearly the above algorithm for DNA maps will not hold for general maps, because when selecting an arbitrary vertex to initiate the search for labels, connecting it to the nearest corner of the square and moving in the opposite direction in general case there will be no spot at the twice the distance from the corner and the process could not be continued. Can we design an algorithm for assignment of »sequential« labels to spots of an *arbitrary* map of n vertices scattered inside a square? The answer is positive as will be seen in the next section, where we have outlined a non-trivial modification of the chaos game algorithm for DNA. In this way it has been possible to assign canonical labels to an arbitrary 2-D map of vertices, and if convert any 2-D man into a map »line notation« (1-D sequence).

Before we go to the next section let us emphasize the significance of the above mentioned transformation of 2-D map into a 1-D sequence. One can make a parallel with the searches in chemical documentation for molecular codes, which have been around for over 60 years. In 1949 Wiswesser [12,13] introduced molecular codes in his "line notation" (that is 1-dimensions linear notation) known since as the Wiswesser Line Notation, also referred to as WLN. This was the first line notation capable of describing completely complex molecules. By "completely" we mean giving description of molecules without loss of information, so that once the WLN code is given, the molecule can be reconstructed—which is the primary and essential property for molecular codes. Since then a number of alternative approaches for construction of

"line formulas" for chemical structures have been proposed. One of these schemes of recent time, known as SMILES [14–17], is still widely used. However, very few of over two dozen alternative linear encoding of molecular structures are sufficiently mathematical, which in our view is a highly desirable quality for codes. These include, among others, the approach of Balaban and Harary [18] for benzenoid hydrocarbons, the approach of Read [19,20] for saturated acyclic and cyclic hydrocarbons, and the approach of Lozac'h, Goodson et al. [21–24] based on nodal properties of graphs, the approach of Knop, Trinajstić et al. [25,26] using n-tuple code, Zupan et al. on uniform spectral representation of chemical structures [27–30], and other [31,32]. The topic of chemical nomenclature, canonical labeling of atoms and line notation are still of considerable interest in chemistry, particularly efforts to use as much as possible mathematical background and as little as possible conventional rules. In this respect very promising is the most recently proposed algorithm named a "Walk Above Graph" approach [33,34]. This approach, which may be viewed as purely mathematical (i.e., devoid of conventional inputs), has been earlier used only for acyclic graphs [33], but now has been generalized to cyclic graph using letters (or alternatively non-binary numbers) to label vertices of ring closure [34].

Our problem on designing canonical labels for spots of maps is conceptually similar to searching for canonical labels for atoms in a molecule, because once one obtains canonical labels for spots of a map one can order points (each characterized by a pair of coordinates) into a sequence, a 1-dimensional object. This may be of interest for computer manipulations of maps in searching for similar or identical maps and compiling catalogue of maps of certain kind, such as proteomic maps.

2 Modified labeling algorithm for general maps

We will outline the construction of canonical labels for general maps, which includes proteomics maps. The approach is related to the construction of graphical representation of DNA introduced by Jeffrey that was described in the previous section. Let us start with an arbitrary map on 20 vertices illustrated in Fig. 2. The map of Fig. 2 has been selected for illustration of the modified algorithm for labeling of vertices of arbitrary maps and is obtained by using random number generator to give random (x, y) coordinates in the domain (1, 100) and excluding repetition of numbers. In Table 1 we show the x, y coordinates for the 20 unlabelled spots of Fig. 2.

It is not essential to exclude repetition of random numbers unless they produce the same pair of (x, y) coordinates that have already been selected. Equally, it is not essential that we use integer coordinates, but using integers between 1 and 100 makes the calculations and search for labels more transparent. Let us label the four edges of the 100×100 units square with labels S (South, bottom), E (East, right), N (North, top), and W (West, left). Then the four corners of the square assume labels SW, SE, NE and NW for southeast, southwest, northwest and northeast, which have the coordinates: (0, 0), (100, 0), (100, 100) and (0, 100), respectively.

We will adopted the following canonical rule:

Table 1 The random (x, y) coordinates for 20 points of Fig. 2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
x	3	19	23	86	13	100	95	65	89	46	50	35	56	79	82	41	11	99	32	87
y	26	94	66	30	24	59	71	43	68	4	17	15	91	72	18	1	53	2	10	88

Label 1 is assigned to the vertex closest to the center of one of the four half diagonals from the center ($x_0 = 50$, $y_0 = 50$) to the four corners (0, 0), (100, 0), (100, 100) and (0, 100). Let us also assume that there is only one point (x_i, y_i) satisfying the four conditions, to which thus we assign the canonical label 1. The point to obtain the canonical label 2, is the point which is closest to the center of one of the four line-segments from the point 1 (x_1, y_1) to the four corners SW, SE, NE and NW. Let again assume that there is only one such point, having coordinates (x_2, y_2), which satisfies the condition to be the closest to one of the new half distances from point 1 to the four corners, to which we assign label label 2. The process continues. Continuing from point 2 the next point, that will obtain canonical label 3, is the point which is closest to the center of one of the four line-segments from the point 2 (x_2, y_2) to the four corners SW, SE, NE and NW, and so on.

If it happens that at any step in the process there are more than one point at the same distance from the line-segment centers one selects the point having smaller x coordinate. If two points have the same x coordinate one selects the point having smaller y coordinate.

In Table 2 we illustrate the search for the spot to be given canonical label 1. We have to calculate the distances of all the 20 spots of Fig. 2 from the four centers of the four line segments between the origin (50, 50) and the corners SW, SE, NE, and NW of the square having coordinates: (0, 0), (100, 0), (100, 100) and (0, 100), respectively. As one can see from Table 2 the smallest entry (shown in bold) in Table 2 is in the row 16 and the column NE, signifying that the spot currently having (an arbitrary) label 14 should have the canonical label 1.

In the next step one calculates distances of the remaining 19 points from the mid points of the four line segments from the point having coordinates (79, 72) and the four corners SE, SW, NW and NE. If one does this one finds that the point at (87, 88), in the quadrant NW, is at the smallest distance from the mid of the line-segment, giving thus to this point canonical label 2. To obtain the remaining canonical labels one continues the process. In Table 3 we have listed for the sought canonical labels the quadrants, the minimal distances, the (x, y) coordinates, the old (arbitrary) labels and the binary labels for the quadrants of spots. The new, canonical labels for the map of Fig. 2 are illustrated in Fig. 3, which gives the solution to the problem of unique canonical labeling of unlabeled map of Fig. 2. The first, fourth and fifth column of Table 2 have essential information for labeling (or relabeling) of any geometrical map based on set on n spots (vertices). The listed minimal distances are of no interest except for verifying the accuracy of the reprorted calculations of canonical labels. All calculations shown are made using microsoft Excel program.

The list of quadrants may, however, be of some interest as it offers construction of a binary code for the map. The code is based on fusing the x, y coordinates of the four

Table 2 Distances of the 20 spots of Fig. 2 from the centers of the line segments between the origin (50, 50) and the corners SW, SE, NE, and NW of the square having coordinates: (0, 0), (0, 100), (100, 100) and (0, 100), respectively

Spot	SW	SE	NE	NW	Spot	SW	SE	NE	NW
1	22.02	72.01	87.09	53.71	11	26.25	26.25	63.16	63.16
2	69.26	88.87	59.14	19.92	12	14.14	41.23	72.11	60.83
3	41.05	66.22	52.77	9.22	13	72.92	68.68	24.84	34.89
4	61.20	12.08	46.32	75.80	14	71.59	47.17	**5.00**	54.08
5	12.04	62.01	80.28	52.39	15	57.43	9.90	57.43	80.61
6	82.35	42.20	29.68	76.69	16	28.84	41.62	81.44	75.71
7	83.76	50.16	20.40	70.11	17	31.30	69.86	67.68	26.08
8	43.86	20.59	33.52	51.22	18	77.49	33.24	76.84	103.95
9	77.10	45.22	15.65	64.38	19	16.55	45.54	77.94	65.38
10	29.70	35.81	76.69	74.04	20	88.39	64.13	17.69	63.35

Table 3 The quadrants, the minimal distances, the (x, y) coordinates, old (arbitrary) labels and binary labels for the quadrants of spots

Canonical	Quadrant	Distance	Coordinates	Old label	Binary code
1	NE	5.00	(79, 72)	14	11
2	NE	3.20	(87, 88)	20	11
3	NW	12.85	(56, 91)	13	10
4	NW	9.12	(19, 94)	2	10
5	SW	6.18	(11, 53)	17	00
6	SW	2.55	(3, 26)	1	00
7	SE	4.27	(50, 17)	11	01
8	SW	7.16	(32, 10)	19	00
9	NE	11.70	(100, 59)	6	11
10	NE	9.86	(95, 71)	7	11
11	SE	12.75	(86, 30)	4	01
12	NE	5.00	(89, 68)	9	11
13	SE	20.30	(82, 18)	15	01
14	SW	7.07	(46, 4)	10	00
15	NE	12.04	(65, 43)	8	11
16	SW	6.98	(35, 15)	12	00
17	NW	10.12	(23, 66)	3	10
18	SW	9.12	(13, 24)	5	00
19	SE	19.01	(41, 10)	16	01
20	NE	21.27	(99, 2)	18	11

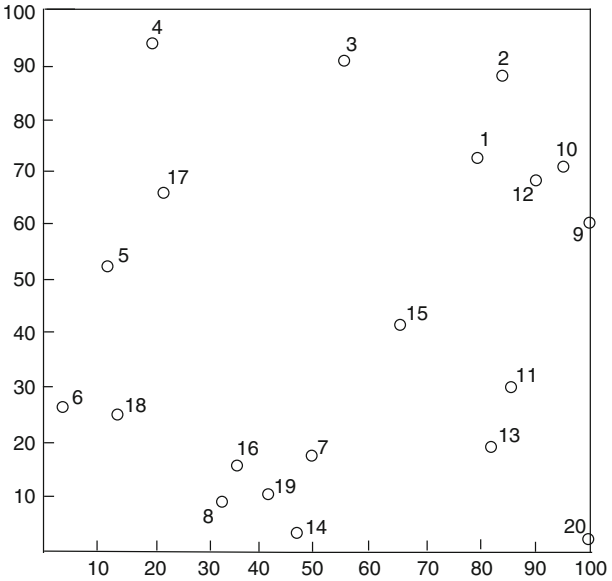

Fig. 3 Canonical labels for vertices of the map of Fig. 2

corners of the square assuming that the square has unit sides. Then one obtains for the four corners SW, SE, NE, and NW the binary codes 00, 01, 11, and 10, respectively. A list of binary codes of quadrants in which successive spots are located yields for the map of Fig. 2 the code:

1 1 1 1 1 0 1 0 0 0 0 0 0 1 0 0 1 1 1 1 0 1 1 1 0 1 0 0 1 1 0 0 1 0 0 0 0 1 1 1

An advantage of this code is that it is very simple, though not necessarily unique. Hence, such code can be used for pre-screening data (catalogues of maps). If in pre-screening one finds two maps to have the same binary code then additional, more discriminating codes, will have to be used to establish if two maps are identical or are different. We will describe one such more discriminating binary code of approximately the same length in the next section.

Alternatively, one may consider a code based on listing coordinates of the points. Usually, already coordinates of few initial spots may suffice to discriminate between non-isomorphic maps. In the case of the map of Fig. 2 the code based on the (x, y) coordinates of spots will start with: 797287885691. . . . One can truncate such codes after a dozen digits, because it is very unlikely that non isomorphic maps will have long list of ordered points in the same locations.

3 On characterization of canonical maps

Once one has found canonical labels for a map one can construct a selection of matrices that can generate map invariants. Such list of invariants can be augmented by including also the list of row sums, as these have already been ordered by canonical

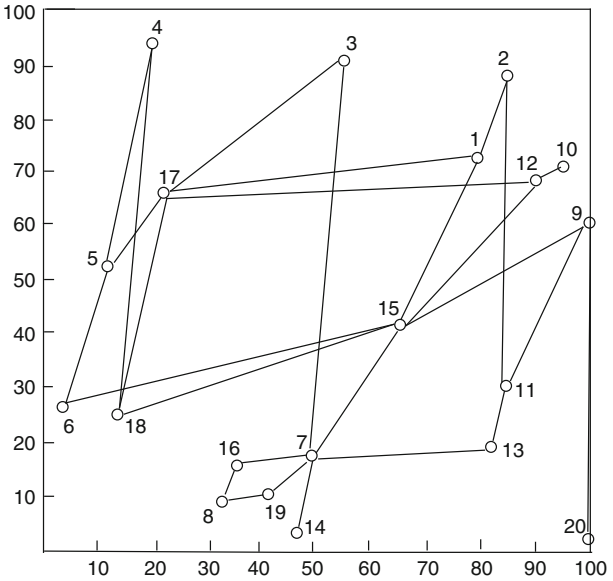

Fig. 4 The graph of partial ordering of vertices of the map of Fig. 2

labels. One can consider characterization of the map based on the distance matrix as a source of numerous distance-related map invariants analogous to distance related graph invariants. However, the distance matrix has a disadvantage of being dense, and consequently computationally intensive, which could be time-consuming. Use of sparse matrices for construction of invariants would be preferred alternative. One can associate sparse matrices with maps by superimposing geometrical objects over the map, the adjacency matrix and distance-weighted adjacency matrix of which will be sparse matrices. Such an approach has been considered for characterization of proteomics maps [35,36] and proteins [11]. Here we will consider two such possibilities: (1) Construction of a graph of partial order for the vertices of the map [37,38]; and (2) Construction of a graph of sequential nearest neighbors for the vertices of the map [39,40]. In both cases map will be represented by a sparse binary matrix.

3.1 Graph of partial ordering of vertices of a map

In Fig. 4 we show the diagram of partial ordering of the 20 vertices of Fig. 3, which is based on domination of coordinates (x_i, y_i) for pairs of vertices. If vertex i dominates vertex j, or the other way round, the two vertices will be connected by a line. Observe that all the connecting lines have positive slope, which implies that the (x, y) coordinate of a vertex above dominate (are numerically greater) the corresponding coordinates of the vertex below. From the adjacency matrix the distance-adjacency matrix, DA, can easily be constructed by replacing the non-zero entries in the adjacency matrix with the corresponding distances between adjacent vertices. The DA matrix is therefore

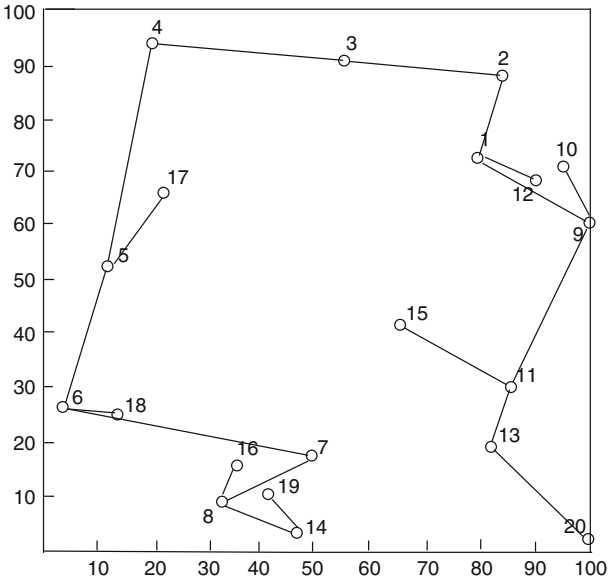

Fig. 5 The graph of sequential nearest neighbors

weighted adjacency matrix with its elements given by the geometrical distances of adjacent vertices.

3.2 Graph of sequential nearest neighbors for the vertices of the map

In Fig. 5 we show the graph of the sequential nearest neighbors for the 20 vertices of the map of Fig. 3. Observe that we have not said: »for the 20 vertices of the map of Fig. 2« , even though map 2 and map 3 is the same map. The difference is that map of Fig. 2 has no labels for its vertices and the map of Fig. 3 has unique canonical (not arbitrary) labels. In order to construct the sequential nearest neighbors graph we need to have labeled graph. If we want a unique solution we should base our construction on graph having unique canonical labels.

The construction starts by connecting the vertex 1 and vertex 2. Vertex 3 is then connected to either vertex 1 or vertex 2, depending on which of the two already connected vertices are closer to vertex 3. If both vertices are at the same distance vertex 3 is, by convention, connected to the vertex having the smaller label. One continues construction of the sequential nearest neighbors graph by connecting vertex 4 to the closest vertex among those already connected. When all vertices have been connected the process ends and the result is an acyclic graph superimposed over the map illustrated in Fig. 5. In the case of the map of Fig. 3 it turned out that only in the case of vertex 19 there were two vertices at the same distance, the vertices 14 and 16. By following our convention we connected vertex 19 to vertex 14, having smaller label.

The graph of sequential nearest neighbors also allows one to construct generalized distance-adjacency matrix by replacing the non-zero entries in the adjacency matrix

with the corresponding distances between adjacent vertices. However, the graph of sequential nearest neighbors has additional advantages over the graph based on partial ordering, which are of interests for characterization of maps. First, being acyclic graph, it allows one to construct terminal matrix [41–44], in which are recorded only the distances between terminal vertices of the graph, which are this case vertices: 10, 12, 15, 16, 17, 18, 19 and 20. This offers representation of the map of Figs. 2 or 3 by a matrix of smaller order, a 7×7 distance matrix instead of a 20×20 matrix. According to a theorem of Zaretskii [45,46] there is no loss of information with terminal matrix, as it contains all information on a graph, which can be reconstructed form it.

The graph of the sequential nearest neighbors also allows one to construct an alternative binary code for the map that has a high discriminatory power then previously mentioned binary code for the map, even though it is also not unique. To arrive at this binary code we will use a »walk above the graph« [33] for the construction of its binary code. The »walk above the graph« algorithm for construction of binary code for acyclic graphs can be viewed as suitably and significantly modified »walk around a tree« of Read [47]. The »walk around code« for trees assigns to each edge of a graph binary labels 0 and 1 in the following way: One starts the »walk around a tree« at any vertex and moves clockwise or counter-clockwise around the graph assigning label 0 to any edge that one pass for the first time. When the same edge, as the walk around the tree progresses, is viewed again from the other side of the graph one assigns to it label 1. Clearly such labeling of edges is not unique because it depends on the vertex selected to initiate the walk around, on the sense of moving around the graph (clockwise or counter-clockwise), and on the 2-D pictorial representation of the graph, that is, the final code will depend also on how graph is drawn. The code of the graph is constructed by listing the binary labels in order of their assignment.

In the case of the sequential nearest neighbors, because the graph that is embedded over the map this eliminates alternative pictorial representations of the graph. But, even if graph would be drawn arbitrarily, as long as vertices retain the same canonical labels, the resulting code will be the same, because we have a rule how to proceed from one vertex to another. Because the graph has canonical labels, this fixes the starting point for the construction of the binary code. Finally, because will be moving *above* the graph, not around, and we will at every site move in the direction of the nearest vertex having the *smallest* label, this eliminates the ambiguity of choosing between clockwise or counter-clockwise directions and choosing between different edges at branching vertices. As a result our binary code will always unique. However, it is possible to have two or more non-isomorphic maps that will have the same graph of sequential nearest neighbors, though one does not expect this to be frequently the case.

Let us now construct the »walk above the graph« binary code for the sequential neighboring graph of Fig. 5. Looking at Fig. 5 we start from vertex 1 and move toward vertex 2, 3, 4 arriving at vertex 5, which is a branching vertex. Hence, to edges (1,2), (2,3), (3,4) and (4,5) are assigned label 0. Thus our code starts with 0 0 0 0. At the branching vertex 5 according to our rule we move toward vertex 6 (having the smaller label) and continue to 7. Similarly when arriving at vertex 8 we go to vertex 14 and end with vertex 19. In this way we have additional five zeros for edges (5, 6), (6, 7), (7, 8), (8, 14) and (14, 19), giving for the code 0 0 0 0 0 0 0 0 0 when we arrived to vertex 19. Now we have to go back to vertex 14 and 8 and have to assign to edges

(19, 14) and (14, 8) label 1, because we passed these edges before, which thus gives for the code: 0 0 0 0 0 0 0 0 0 1 1. Coming back to vertex 8 we have first go to vertex 16, because edge (8, 16) has not yet been visited, rather than returning to vertex 7, because edge (8, 7) has already been visited. The rule is that edges that have not yet obtained label 0 have precedent over edges that already have binary assignment zero. Our code thus continues with 0 for edge (8, 16), then 1 for edge (16, 8) and 1 for edge (8, 7), giving up to this point: 0 0 0 0 0 0 0 0 0 1 1 0 1 1.

With this introductory information it is now not difficult to proceed and complete the code which in its entirety is:

0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 0 0 1 0 0 0 1 1 0 1 1 1 0 1

Which has 38 binary characters, twice the number of edges of the graph. Observe that to an edge (i, j) is assigned label 0 if $i < j$ and label 1 if $i > j$.

3.3 Dual of the map

The »walk above the graph« code, besides offering binary code for a map allows one to construct a dual of the sequential neighbor graph embedded on the map, to which we will refer as the dual of the map. To obtain this dual one follows the algorithm outlined in the book of Rouse Ball [48,49], which starts by replacing the binary entries 0 and 1 of the code by the left and the right brackets. For the above »walk above the graph« code this gives:

(((((((((()))))())()))))(()((())))()

In the following step one connects the adjacent left and right brackets forming thus circles. In the above case there are eight such instances, corresponding to eight terminal vertices. Starting from left one can assign labels 19, 16, 18, 17, 10, 20, 15, and 12 to the eight constructed circles:

Then one ignores the circles, and pretending that they have been erased, continues to connect adjacent left and right brackets. Erasing circles corresponds to pruning a tree (that is, erasing terminal vertices and their incident edges). In the above case, after pruning eight terminal vertices, one obtains a path of length 11 with terminal vertices 14 and 13. In this way one obtains two novel circles, the first belongs to vertex 19 and the second belongs to vertex 20. The process continues till all left brackets are connected to corresponding right brackets, which results in map dual shown in Fig. 6.

From the dual of Fig. 6 one can reconstruct not only the acyclic graph on which it has been based but also the labels of the vertices. To obtain the graph one inserts a vertex in each of the circles or ellipses of the dual graph including adding vertex to the outside area. Then one connects vertices in adjacent areas which are separate by single curved line. The result is graph of the acyclic graph used for construction of the binary code. Then one can follow with assignment of labels as outlined earlier if

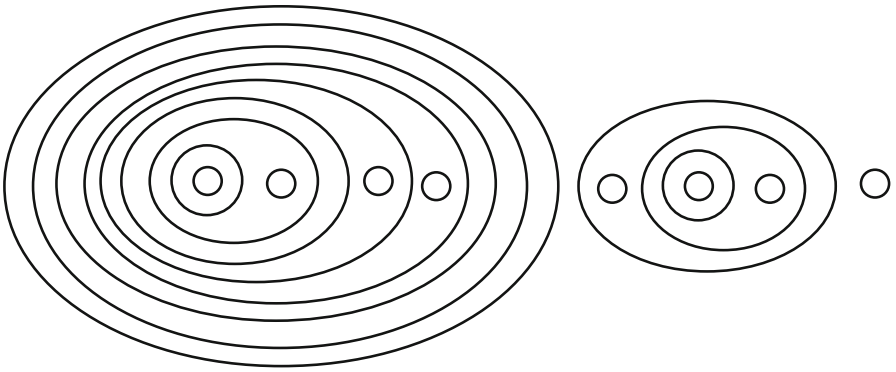

Fig. 6 Dual of the map based on the sequential nearest neighbor graph of Fig. 5

desired. Once one has the canonical labels of vertices from Table 3 one can recover the coordinates of spots of the map and one can thus reconstruct the map. Observe that the map dual in Fig. 6 consists of three disjoint segments, which correspond to three branches of the starting vertex 1 of the map. Similarly, if one erases the outer ellipse in the central segment, which belongs to the edge (1, 12), one has two disjoint segments, which correspond to the two remaining branches of the vertex 9, and so on. Map duals offer a qualitative representation of a map, which may have some visual advantages in comparative studies of maps and map classifications.

4 Concluding remarks

Canonical labeling of vertices (spots) of maps are essential not only for establishment of isomorphism of maps, which in turn is important in comparative studies of maps, but also as it allows different researchers in different laboratories to come to a common labeling and common coding for maps. A side benefit of canonical labels is the possibility of using matrix entries as additional map invariants because they have been already ordered. The canonical labels here introduced are conceptually and computationally simple, because finding them does not involve exponential complexity, which is characteristic in searches for canonical labels based on the adjacency matrix, which when its rows are read from left to right and from top to bottom, produces the smallest binary number possible. Nevertheless, the novel canonical labels outlined in this article (based on modified chaos game representation of DNA and the sequential nearest neighbor graph of the map) will allow one to construct binary codes of high discriminatory power for maps and thus make possible to build catalogues of maps that could be more readily searched for data on the same map from different sources and searching for similar maps.

Acknowledgments Rok Orel thanks for the financial support the European Union, European Social Fund. Milan Randić thanks the Laboratory for Chemometrics, National Institute of Chemistry, Ljubljana, Slovenia, for hospitality. This work has been supported in part by the Ministry of Science and Higher Education, of Republic of Slovenia under Research Grant P1017.

References

1. R.C. Read, D.G. Corneil, The graphs isomoprhism desease. J. Gr. Theory **1**, 339–363 (1977)
2. M. Randić, On the recognition of identical graphs representing molecular topology. J. Chem. Phys. **60**, 3920–3928 (1974)
3. M. Randić, On canonical numbering of atoms in a molecule and graph isomorphism. J. Chem. Inf. Comput. Sci. **17**, 171–180 (1977)
4. M. Randić, Mathematical methods in contemporary chemistry, in *Similarity Methods of Interest in Chemistry*, ed. by S. Kuchanov (Gordon & Breach Publ. Inc., New York, 1995), pp. 1–99
5. T. Pisanski, M. Randić, Bridges between geometry and graph theory, in *Geometry at Work*, ed. by C.A. Gorini (Math. Assoc. America, Washington, DC, 2000), pp. 174–194
6. J.V. Knop, W.R. Müller, K. Szymansky, N. Trinajstić, *Computer Generation of Certain Classes of Molecules* (SKTH / Kemija u Industriji, Zagreb, 1985)
7. M. Randić, A graph theoretical basis for structural chemistry. 1. Structures based on trivalent graph with n = 10 vertices. Acta Cryst. A **34**, 275–282 (1978)
8. M. Randić, N. Lerš, D. Vukičević, D. Plavšić, B.D. Gute, S.C. Basak, Canonical labeling of proteome maps. J. Proteome Res. **4**, 1347–1352 (2005)
9. M.F. Barnsley, *Fractals Everywhere*, 2nd edn. (Academic Press, Boston, 1993)
10. H.J. Jeffrey, Chaos game representation of gene structure. Nucleic Acid Res. **18**, 2163–2170 (1990)
11. M. Randić, J. Zupan, A.T. Balaban, D. Vikić-Topić, D. Plavšić, Graphical representation of proteins. Chem. Rev. **111**, 790–862 (2011)
12. W.J. Wiswesser, How the WLN began in 1949 and how it might be in 1999. J. Chem. Inf. Comput. Sci. **22**, 88–93 (1982)
13. W.J. Wiswesser, Historical development of chemical notation. J. Chem. Inf. Comput. Sci. **25**, 258–263 (1985)
14. D. Weininger, A. Weininger, J.L. Weininger, SMILES, a modern chemical language and information system. Chem. Des. Autom. News **1**, 2–15 (1986)
15. D. Weininger, Chemical language and information system. Introduction to methodology end encoding rules. J. Chem. Inf. Comput. Sci. **28**, 31–36 (1988)
16. D. Weininger, A. Weininger, J.L. Weininger, SMILES 2. Algorithm for generation of unique SMILES notation. J. Chem. Inf. Comput. Sci. **29**, 97–101 (1989)
17. R.G.A. Bone, M.A. Firth, R.A. Sykes, SMILES extensions for pattern matching and molecular transformations: applications to chemo informatics. J. Chem. Inf. Comput. Sci. **39**, 846–860 (1999)
18. A.T. Balaban, F. Harary, Chemical Graphs V. Enumeration and proposed nomenclature of benzenoid cata-condensed polycyclic aromatic hydrocarbons. Tetrahedron **24**, 2505–2516 (1968)
19. R.C. Read, A new system for designation of chemical compounds. I. Theoretical preliminaries and the coding of acyclic compounds. J. Chem. Inf. Comput. Sci. **23**, 135–149 (1983)
20. R.C. Read, A new system for designation of chemical compounds. 2. Coding of cyclic compounds. J. Chem. Inf. Comput. Sci. **25**, 116–128 (1985)
21. N. Lozac'h, A.L. Goodson, W.H. Powel, Die nodal nomenklature—Allgemeine Prinzipen. Angew. Chem. **91**, 951–964 (1979)
22. A.L. Goodson, Graph-based chemical nomenclature. 1. Historical background and discussion. J. Chem. Inf. Comput. Sci. **20**, 167–172 (1980)
23. A.L. Goodson, Graph-based chemical nomenclature. 2. Incorporation of graph-theoretical principles into Taylor's nomenclature proposal. J. Chem. Inf. Comput. Sci. **20**, 172–176 (1980)
24. A.L. Goodson, Application of graph-based chemical nomenclature to theoretical and preparative chemistry. Croat. Chem. Acta **56**, 315–324 (1983)
25. N. Trinajstić, Ž. Jeričević, J.V. Knop, W.R. Müller, K. Szymanski, Computer generation of isomeric structures. Pure Appl. Chem. **55**, 370–390 (1983)
26. J.V. Knop, W.R. Müller, K. Szymanski, N. Trinajstić, *Computer Generation of Some Classes of Molecules* (SKTH Press, Zagreb, 1985)
27. C. Rücker, G. Rücker, Nomenclature of organic polycycles out of the computer—how to escape the jungle of the secondary bridges. Chimia **44**, 116–120 (1990)
28. E. Kirby, Coding and enumeration of trees that can be laid upon hexagonal lattice. J. Math. Chem. **11**, 187–197 (1991)
29. M. Novič, J. Zupan, A New General and Uniform Structure Representation, in *Software development in chemistry 10, GDCh*, ed. by J. Gasteiger (Frankfurt an M, 1996), pp. 48–58

30. M. Novič, M. Vračko, Comparison of spectrum-like representation of 3D chemical structure with other representations when used for modelling biological activity. Chemom. Intel. Lab. Syst. **59**, 33–44 (2001)
31. J. Zupan, M. Vračko, M. Novič, New uniform and reversible representation of 3d chemical structures. Acta Chim. Slov. **47**, 19–37 (2000)
32. J. Zupan, M. Novič, Optimization of structure representation for QSAR studies. Anal. Chim. Acta **388**, 243–250 (1999)
33. M. Randić, M. Novič, D. Vikić-Topić, D. Plavšić, Novel mathematical code for molecular graphs. Croat. Chem. Acta (submitted)
34. M. Randić, M. Novič, A.R. Choudhury, D. Plavšić, On graphical representation of trans-membrane proteins. SAR QSAR Environ. Res. **23**, 327–343 (2012)
35. M. Randić, Quantitative characterization of proteomics maps by matrix invariants, in *Handbook of Proteomics Methods*, ed. by P.M. Conn (Humana Press Inc, Totowa, NJ, 2003), pp. 429–450
36. M. Randić, A graph theoretical characterization of proteomics maps. Int. J. Quantum Chem. **90**, 848–858 (2002)
37. M. Randić, A graph theoretical characterization of proteomics maps. Int. J. Quantum Chem. **90**, 848–858 (2002)
38. M. Randić, M. Novič, M. Vračko, D. Plavšić, Study of proteome maps using partial ordering. J. Theor. Biol. **266**, 21–28 (2010)
39. M. Randić, M. Novič, M. Vračko, Novel characterization of proteomics maps by sequential neighbourhood of protein spots. J. Chem. Inf. Model. **45**, 1205–1213 (2005)
40. M. Randić, N. Lerš, D. Plavšić, S.C. Basak, Characterization of 2-D proteome maps based on the nearest neighborhoods of spots. Croat. Chem. Acta **77**, 345–351 (2004)
41. E.A. Smolenskii, A method for the linear recording of graphs. Zh. Vychisl. Mat. Mat. Fiz. **2**, 371–372 (1962)
42. E.A. Smolenskii, On coding the structural formulas of organic compounds. Dokl. Chem. **380**, 237–241 (2001)
43. M. Randić, J. Zupan, D. Vikić-Topić, Graphical representation of proteins by star-like graphs. J. Mol. Graph. Model. **26**, 290–305 (2007)
44. B. Horvat, T. Pisanski, M. Randić, Terminal polynomials and star-like graphs. MATCH Commun. Math. Comput. Chem. **60**, 493–512 (2008)
45. K.A. Zaretskii, Constructing a tree on the basis of a set of distances between the hanging vertices. Uspekhi Mat. Nauk **20**, 90–92 (1965). (in Russian)
46. E.A. Smolenskii, E.V. Shuvalova, L.K. Maslova, I.V. Chuvaeva, M.S. Molchanova, Reduced matrix of topological distances with a minimum number of independent parameters: distance vectors and molecular codes. J. Math. Chem. **45**, 1004–1020 (2009)
47. R.C. Read, Survey of graph generation techniques. Lect. Notes Math. **884**, 77–89 (1981)
48. W.W. Rouse Ball, *Mathematical Recreations and Essays* (MacMillan and Co., Ltd., London, 1922)
49. W.W. Rouse Ball, H.S.M. Coxeter, *Mathematical Recreations and Essays* (University of Toronto Press, Toronto, 1974)