The Dominating Cycle Problem in 2-Connected Graphs and the Matching Problem for Bag of Bags are NP-Complete*

Peter Z. Revesz
Department of Computer Science and Engineering
University of Nebraska–Lincoln, Lincoln, NE 68588, USA

1 Introduction

In this paper we study the problem of dominating cycles in 2-connected graphs and the matching problem for two bag of bags. These problems occur as two different abstractions of the genome map assembly problem [3], but they can be also interesting in other contexts. In this paper we show first time that these two problems are NP-complete.

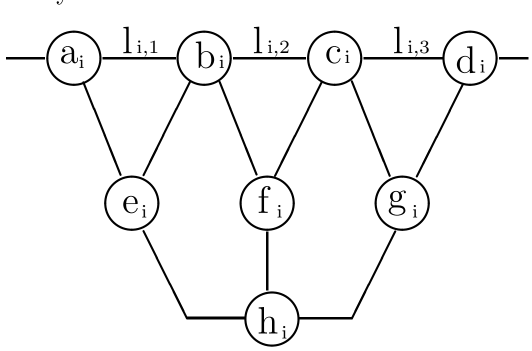
2 Dominating Cycles in 2-Connected Graphs

A *dominating cycle* is a cycle L in graph G for which every vertex of G is adjacent to at least one vertex of L. The dominating cycle problem is known to be NP-complete in the case of planar graphs [2] and bipartite graphs [1]. In this paper we show that this problem is also NP-complete for 2-connected graphs, i.e. graphs in which for every distinct triple of vertices v, w, u there exists a path between v and w not containing u.

Theorem 2.1 The dominating cycle problem is NP-complete for 2-connected graphs.

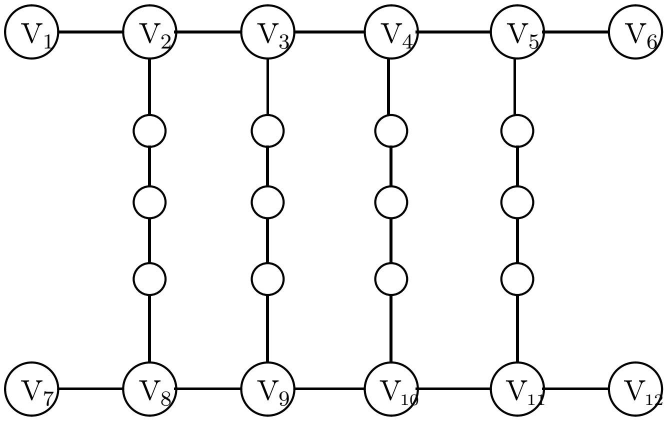
Proof: *(NP)* An algorithm can go through the cycle once and check that all vertices are either on the cycle or adjacent to some vertex on the cycle.

(NP-hardness) This we show by reduction from the 3-CNF boolean formula problem. The idea is to build a graph that has a dominating cycle if and only if the 3-CNF boolean formula is satisfiable. If some variable occurs only positively (or only negatively), then we can assign to it true (or false), which makes all the disjuncts true in which it occurs. The original 3-CNF formula is satisfiable if and only the rest of the disjuncts are satisfiable. Hence we will assume that each variable occurs both positively and negatively in the 3-CNF formula.



*This work was supported by NSF grants IRI-9625055 and IRI-9632871 and a Gallup Research Professorship.

We build a subgraph as shown in the first figure for each disjunct of the 3-CNF formula. We link these subgraphs in a cycle. If the resulting graph has a dominating cycle, then it must go through each subgraph in a way that *not* all of $(a_i, b_i), (b_i, c_i), (c_i, d_i)$ are used, otherwise the vertex h_i will neither be on the dominating cycle nor be adjacent to a vertex on the dominating cycle.



We label the edge (a_i, b_i) with literal $l_{i,1}$, (b_i, c_i) with $l_{i,2}$ and (c_i, d_i) with $l_{i,3}$. We insert a subgraph between each pair of edges (v_1, v_6) and (v_7, v_{12}) labeled with opposite literals (e.g., x and $\neg x$) as shown in the second figure. The size of the whole graph is $O(n^2)$ when the size of the 3-CNF formula is n. In each subgraph any dominating cycle either goes through $v_1, v_2, ..., v_8, v_9, ...v_3, v_4, v_{10}, v_{11}, ..., v_5, v_6$ or through $v_7, v_8, ..., v_2, v_3, ..., v_9, v_{10}, ..., v_4, v_5, ..., v_{11}, v_{12}$. In the first case the edges (v_7, v_8) and (v_{11}, v_{12}) and in the second case the edges (v_1, v_2) and (v_5, v_6) cannot be used. This means that the dominating cycle enters v_1 and leaves v_6 and does not enter v_7 and leave v_{12} or the reverse.

Suppose that the graph has a dominating cycle. Then we show that the 3-CNF formula is true. For each literal l if the edge that it labels is *not* on the cycle, assign it true else assign it false. Since in each ith subgraph not all of the three edges $(a_i, b_i), (b_i, c_i), (c_i, d_i)$ are used, at least one of the literals must be true in each disjunct. Furthermore, if one literal is true, then its negation must be false.

For the other direction, we can show that if the 3-CNF formula is true, then there is a dominating cycle, which can be found by taking any satisfying truth assignment and if the literal $l_{i,1}$ is true then adding edges (a_i, e_i) and (e_i, b_i) and if it is false, then adding edge (a_i, b_i) to the dominating cycle. Similarly, if $l_{i,2}$ is true then we add (b_i, f_i) and (f_i, c_i) else (b_i, c_i), and if $l_{i,3}$ is true then we add (c_i, g_i) and (g_i, d_i) else (c_i, d_i).

Finally, the graph is 2-connected. For a moment ignore the structures added between oppositely labeled edges. Then we have a cycle of subgraphs, which are themselves 2-connected, hence the graph must be 2 connected. Since the internal structure that we added between oppositely labeled edges is also 2-connected. the entire graph must be 2-connected. □

3 The Matching Problem for Two Bag of Bags

In this section we show that the matching problem for two bag of bags is also NP-complete. A *bag* is a multiset in which each symbol is in Σ, where Σ is some finite alphabet, and can occur more than once. A *big-bag* is a multiset whose elements are bags that can occur more than once.

We call each permutation of the bags and permutation of the elements of each bag within a big-bag a *presentation*. A big-bag can have several different presentations. For example, $a_1 = [[1, 3], [2, 4], [5, 7, 8]]$ is one presentation of a big-bag A. Similarly, $b_1 = [[1, 2, 4, 8], [3], [5, 7]]$ is a presentation of another big-bag B.

We say that the ith element of a presentation a, written as $a[i]$, is the ith Σ symbol seen when the presentation is read from left to right. For example, in a_1 the 3rd element is 2 and the 6th element is 7, i.e. $a_1[3] = 2$ and $a_1[6] = 7$. The total number of elements, which is independent of any presentation, in A is 7.

We say that two big-bags A and B both of which have n elements *match* if there are presentations a for A and b for B such that $a[i] = b[i]$ for $1 \leq i \leq n$. For example, A and B match because A can be presented as $a_2 = [[3, 1], [2, 4], [8, 5, 7]]$ and B can be presented as $b_2 = [[3], [1, 2, 4, 8], [5, 7]]$.

Theorem 3.1 Deciding whether two big-bags match is NP-complete in the size of the big-bags.

Proof: *(NP)* For any two given presentations, we can check in linear time by a single scan of them that they have the same elements in each position.

(NP-hardness) This is shown by reduction from the Hamiltonian cycle problem. Let G be any undirected graph with n vertices labeled by the integers from 1 to n and m edges, none of which is a loop, i.e. connects a vertex with itself. Let x, y, z be three distinct labels different any of the vertex labels. Let $d(i)$ be the degree of vertex v_i. We define two big-bags B_1 and B_2 as follows:

$B_1 = [\ldots, [i, j], \ldots, [x], [y, y], [z, z]]$

where each $[i, j]$ is present in B_1 if there is an edge adjacent to v_i and v_j in G for $1 \leq i < j \leq n$.

$B_2 = \{[1, 1], [2, 2], \ldots, [n-1, n-1], [x, n], [n, y], [z], [y, 1^{d(1)-2}, \ldots, n^{d(n)-2}, z]\}$

where the notation l^k within a bag is a shorthand for k subsequent copies of label l.

We show that B_1 and B_2 match if and only if there is a Hamiltonian cycle in G.

(only if): Suppose that B_1 and B_2 match. We have to show that there is a Hamiltonian cycle in G.

First, observe that $[x]$ in B_1 and $[x, n]$ in B_2 must be aligned on the x label. Without loss of generality, assume that n is to the right of x in the bag $[x, n]$.

Claim 1: $[x, n]$ is to the left of $[z]$.

Suppose that the claim is not true. Note that in both B_1 and B_2 all bags except $[x]$ and $[z]$ have an even number of elements. Hence as we scan from x to the right all bags in B_1 must end at an odd position, while all bags in B_2 must end at an even position. Hence B_1 and B_2 cannot match. This is a contradiction. Hence $[x, n]$ must be to the left of $[z]$. (End-of-Claim)

Claim 2: $[y, 1^{d(1)-2}, \ldots, n^{d(n)-2}, z]$ is immediately before $[z]$.

Clearly, $[z]$ in B_2 can be aligned only with one of the zs of $[z, z]$ in B_1, while the other z will be aligned with the z in $[y, 1^{d(1)-2}, \ldots, n^{d(n)-2}, z]$. Therefore, it must be either immediately before or after $[z]$. If it is after $[z]$, then we must have the following situation:

```
[x]       [z, z]      [y, y]
[x, n]    [z][z, ..., y][y, n]
```

Now, again we have the same problem as before. The bags after $[y, y]$ all end at odd positions, while the bags after $[y, n]$ all end at even positions, because all remaining bags have an even size. Therefore, we must have it before $[z]$. (End-of-Claim)

Because of Claims 1 and 2, we must have the following situation:

```
E3 [x]     E1      [y, y] E2  [z, z] E4
C1 [x, n]   C    [n,y][y, ..., z][z] C2
```

where $E1, E2, E3, E4$ and $C, C1, C2$ are lists of bags.

Claim 3: $E3, E4, C1$ and $C2$ must be empty.

Suppose that $E3$ and $C1$ are not empty. Then the bag immediately preceding [x] must have the form $[j, k]$ where $j \neq k$, because all the B_1 bags that do not correspond to edges were already used. However, the bag immediately preceding $[x, n]$ must have the form $[i, i]$ because all the B_2 bags that do not correspond to vertices have been used. Hence even if $i = j$ or $i = k$, the two bags immediately preceding $[x]$ and $[x, n]$ cannot match, contradicting our assumption. Hence $E3$ and $C1$ must be empty. We can prove similarly that $E4$ and $C2$ also must be empty.

Because of Claims 1,2, and 3, we must have the following situation.

```
[x]     E1      [y, y] E2   [z, z]
[x, n]   C    [n,y][y, ..., z][z]
```

Claim 4: $E1$ is a Hamiltonian cycle.

Clearly, C must be a permutation of the bags $[i, i]$ for $1 \leq i \leq (n-1)$ because these are exactly the bags in B_2 that were not used yet. Whenever C contains two adjacent bags $[i, i][j, j]$, list $E1$ must contain a bag $[i, j]$ that aligns with the second element of $[i, i]$ and the first element of $[j, j]$, because all the unused bags in B_1 correspond to edges and there are no self-loops in G. This implies that for each such alignment, there must be an edge between vertices v_i and v_j in G. Therefore, there must be a cycle from vertex v_n to vertex v_n in G that goes through all the vertices exactly once. (End-of-Claim)

(if): Suppose that G contains a Hamiltonian cycle. We have to show that B_1 and B_2 match.

Clearly, we can arrange the bags in B_1 and in B_2 as shown in the previous picture. Therefore, it is enough to show the following claim.

Claim 5: $E2$ matches the dotted part of $[y, ..., z]$.

Clearly, $E2$ is the set of edges that are not used by the Hamiltonian cycle. Now any permutation of the edges of G must contain each vertex v exactly $d(v)$ times. The concatenation of $E1$ and $E2$ is some permutation of the edges in G. Therefore, the concatenation must also contain each v exactly $d(v)$ times. Further, note that in a Hamiltonian cycle each vertex appears exactly twice. By Claim 4, $E1$ is a Hamiltonian cycle. Therefore, in $E2$ each vertex v must appear exactly $d(v) - 2$ times. Since the dotted part of $[y, ..., z]$ contains each vertex exactly that many times and these can be permuted in any order, there must be a permutation that will match $E2$. (End-of-Claim) $\square$

References

[1] P. Ash and B. Jackson. Dominating cycles in bipartite graphs. In: J.A. Bondy, ed. *Progress in Graph Theory*, p. 81-87, 1982, Academic Press (Toronto, Ont).

[2] A. Proskurowski and M.M. Syslo. Minimum dominating cycles in outerplanar graphs. *Int. J. Comput. Inform. Sci.*, 10:127-139, 1981.

[3] P.Z. Revesz. Refining restriction enzyme genome maps. *Constraints*, 2 (3-4): 361-375, 1997.