Manhattan-Connected Exact-Coloring is NP-Complete

Daniel Shawcross Wilkerson

March 31, 2011

1 The Manhattan-Connected Exact-Coloring Problem

Consider a matrix. Two cells are "manhattan adjacent horizontally" if their vertical coordinates are the same and their horizontal coordinates differ by exactly one. Define "manhattan adjacent vertically" similarly. Two cells are "manhattan adjacent" iff they are manhattan adjacent vertically or horizontally. A "manhattan path" is a sequence of cells within which sequence-consecutive implies manhattan-adjacent. A set of cells is "manhattan connected" iff for all pairs of cells in the set there is a manhattan path between them consisting only of cells in the set.

A "coloring" of a matrix is a map from cells to colors. A subset of a colored matrix is a manhattan-connected exact-coloring iff it is a manhattan-connected subset containing each color exactly once. The "manhattan-connected exact-coloring problem" [MCXC] is the question given a colored matrix, does there exist a subset of the cells that is a manhattan-connected exact-coloring ?

2 MCXC is NP-Complete

Theorem: The manhattan-connected exact-coloring problem is NP-Complete.

Proof: By reduction from the version of [At-Most-] 3SAT where not only does each clause have at most three terminals (variables or their negation) in it, but also each variable occurs in at most 3 clauses; see Garey and Johnson 1979 [2], p 259:

> SATISFIABILITY; INSTANCE: Set U of variables, collection C of clauses over U; QUESTION: Is there a satisfying truth assignment for C ? ... Remains NP-complete even if each c in C satisfies $|c| = 3$ (3SAT), or if each c in C satisfies $|c| \leq 3$ and, for each u in U, there are at most 3 clauses in C that contain either u or not(u).

[*dsw:* that is, reading the related theorems more carefully, this problem is not properly "3SAT" plus an additional constraint, but instead a very closely related, yet different, problem I'll call "At-Most-3SAT": the difference is that in 3SAT each clause must have *exactly* 3 terms (a variable or the negation of a variable), whereas in At-Most-3SAT, each clause must have *at most* 3 terms.]

Mapping At-Most-3SAT to the plane: Given a[n] [At-Most-] 3SAT instance as above, map the formula to a planar graph as follows.

- For an AND, recursively map its elements and then connect them in *series*.
- For an OR, recursively map its elements and then connect them in *parallel*.

For each terminal (a variable or its negation), below we will insert a widget that will connect iff that terminal evaluates to true. Note that in 3SAT the AND is on the outside and the ORs are on the inside, so draw the graph, say, vertically: it is arbitrarily high but only widgets wide. Add unique top and bottom nodes. There will be a path in the graph from top to bottom iff the formula is satisfiable.

Coloring the edges: Digitize the the planar graph and draw it on a matrix, leaving at least two cells of background between any two arcs that are not supposed to connect. Make top and bottom unique colors; the manhattan solver must choose top and bottom for the manhattan-connected exact-colored set because they are unique and it must connect them. Make the background all one color; at most one background cell can be chosen and it would take two of them to make a connection across the background between two parts of the graph. Therefore the connection between top and bottom must be made using only cells within the graph. Except for the terminal widgets (below), make every graph arc a unique color, so that they must be chosen.

Coloring the variables: As AND and OR are monotonic, if all occurrences of a variable are positive we can simply set that variable to true and simply the formula. Similarly for a variable where all occurrences are negative. As we chose the version of 3SAT where each variable occurs at most 3 times [(At-Most-3SAT)], without loss of generality a given variable V occurs twice in the positive and once in the negative (should the reverse be the case, invert the variable and convert to the assumed case).

If a variable occurs thrice, choose two unique colors: V_red and V_blue. At one of the sites of the positive occurrence of V, color a single cell V_red; at the site of the other positive occurrence, color a single cell V_blue; at the site of the negative occurrence of V, put two cells in series (say vertically), one colored V_red and the other colored V_blue. Now, the manhattan solver can either choose to connect either (1) the site of the one negative terminal, or (2) both the sites of the two positive terminals, but not both. (Note that in either case the graph arcs leading to the sites of all terminals will be connected in both directions; that is, due to the way graph arcs were constructed, all having unique colors, we mustn't chop off pieces of arc and leave them disconnected from the rest of the graph; we don't). The case where a variable occurs twice is left to the reader.

Showing this mapping is a satisfiability isomorphism: In sum, the manhattan solver can choose a connected subset having each color exactly once iff it chooses all of the arcs of the graph and exactly one (irrelevant) background cell. If the formula is satisfiable, then top will connect to bottom (and the rest of the graph). If top and bottom connect, the formula is satisfiable.

Quod erat demonstrandum.

Further notes: You don't need each edge in the graph to map to a square. You just draw the graph on the plane the way you would on a computer screen and then use a different color for each square. A graph edge is a curve on the plane, so digitize that and then for each square, pick a new color. That is, most of the graph is unique color squares, except for the widgets corresponding to the terms (variables or their negation) where only two squares are one color. Other than background, no color ever has more than two squares.

Also, to make the proof sound simpler, I just said separate each graph component by two background squares but you can force the manhattan solver to pick just one particular background square if you want by making a square a distance of two from the bottom square and surrounded by background. Now the solver has to pick the one background square between bottom and this new square, which means it can't pick any of the other background squares. This makes the proof a paragraph longer, but the diagram easier to draw in your head.

3 The making of . . .

I came to this problem through a friend who was taking Professor Richard M. Karp's CS 270 Combinatorial Algorithms and Data Structures class at The University of California, Berkeley. At the time I was just trying to help a friend with his homework (of course with the intent to provide proper notification to the course grader). After I solved it my friend told me that in fact neither Karp nor his teaching assistant had a solution.

I [Wilkerson] chose the exact name "Manhattan-Connected Exact-Coloring" for the problem. Karp informs me that this problem originates from Richard M. Karp [unpublished] who derived it from a more general question posed in both BHKSS [1] and "independently in a paper on social networks that I [Karp] am not able to track down".

According to my email records, I sent the original email containing the main body of the proof, Section 1 and Section 2, except Subsection "Further notes", to Karp's teaching assistant on Sun, 23 May 2010 15:47:36 -0700. In response to a further question by him, I sent a follow-up email containing Subsection "Further notes" to him on Sun, 30 May 2010 10:14:24 -0700. I forwarded those emails containing all of those sections to Karp on Sun, 13 Jun 2010 02:09:12 -0700. This document is as it was presented in the above emails except for the addition of all formatting, all section headings, all comments in square-brackets, the title, the bibliography, and this section.

References

[1] Sharon Bruckner, Falk Hüffner, Richard M. Karp, Ron Shamir, and Roded Sharan. Topology-free querying of protein interaction networks. In *RECOMB 2009*, pages 74–89, 2009.

[2] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco, 1979.