GRAPH COLOURING: AN ANCIENT PROBLEM WITH MODERN APPLICATIONS

RHYD LEWIS

AS SOMEONE WHO has been colour-blind since birth – trying to tell the difference between blue and purple is a particularly difficult challenge for me – friends of mine always chuckle when I tell them that I spend my research time at Cardiff University studying graph colouring problems.

Graph colouring has its origins in the work of Francis Guthrie who, while studying at University College London in 1852, noticed that no more than four colours ever seemed necessary to colour a map while ensuring that neighbouring regions received different colours. Guthrie passed this observation on to his brother Frederick who, in turn, passed it on to his mathematics tutor Augustus De Morgan who was unable to provide a conclusive proof for the conjecture. Indeed, despite having no real practical significance (cartographers are usually happy to use more than four different colours of ink) the problem captured the interests, and ultimately stumped, many notable mathematicians of the time, including William Hamilton, Arthur Cayley, Charles Pierce, and Alfred Kempe. In fact, it would eventually take more than 120 years, and the considerable use of large-scale 1970s computing resources to prove conclusively what is now known as the Four Colour Theorem.

To achieve this proof, one early but very important insight was to note that any map can be converted into a corresponding *planar graph*. A graph is simply an object comprising some "vertices" (nodes), some of which are linked by "edges" (lines); a *planar* graph is a special type of graph that can be drawn on a piece of paper so that none of the edges cross. To illustrate, consider the map of Wales in Figure 1(a). In Figure 1(b) we produce the corresponding planar graph by substituting each region in the map (plus England and the

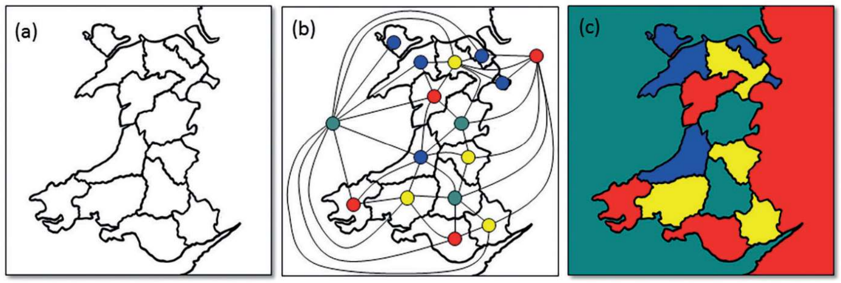
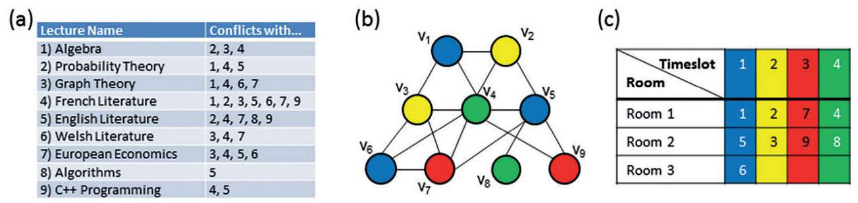


FIGURE 1 - HOW A MAP CAN BE CONVERTED INTO A PLANAR GRAPH. A COLOURING OF THIS PLANAR GRAPH CAN THEN BE USED TO COLOUR THE REGIONS OF THE MAP.



Lecture Name	Conflicts with...
1) Algebra	2, 3, 4
2) Probability Theory	1, 4, 5
3) Graph Theory	1, 4, 6, 7
4) French Literature	1, 2, 3, 5, 6, 7, 9
5) English Literature	2, 4, 7, 8, 9
6) Welsh Literature	3, 4, 7
7) European Economics	3, 4, 5, 6
8) Algorithms	5
9) C++ Programming	4, 5

Timeslot / Room	1	2	3	4
Room 1	1	2	7	4
Room 2	5	3	9	8
Room 3	6			

FIGURE 2 - HOW A TIMETABLING PROBLEM CAN BE CONVERTED INTO A CORRESPONDING GRAPH COLOURING PROBLEM.

Irish Sea) with a vertex and then add edges between pairs of vertices whose corresponding regions share a border. Having produced our planar graph, our task is to now merely to colour the vertices appropriately. That is, we want to use the minimum number of colours (and no more than four) to colour all of the vertices so that those connected by an edge receive different colours. These colours are then transferred back to the graph, as shown in Figure 1(c).

One of the best known applications of graph colouring arises in the production of timetables

Sounds easy doesn't it? But how do we go about producing such a colouring? Moreover, what should we do if we are asked to colour more complicated graphs, such as those that are not planar and where more than four colours are needed? Is there any easy set of steps that we might follow to colour any graph using the minimum possible number of colours? The short answer to the last question is almost certainly "no". Graph colouring is known to be NP-complete in general, meaning that an efficient exact algorithm for the problem almost certainly does not exist. However, graph colouring is definitely a problem that needs to be tackled effectively because is known to underpin a wide variety of real-world operational research problems. Let's now look at some of these.

One of the best known applications of graph colouring arises in the production of timetables. Imagine a college where we need to assign lectures to timeslots. Imagine further that some pairs of lectures *conflict* in that they cannot be scheduled to the same timeslot – perhaps because some student wants to attend both of them. Figure 2(a) demonstrates this. We see, for example, that the Algebra lecture conflicts with Probability Theory, Graph Theory, and (for some reason) French Literature. As a result, in the corresponding graph edges are placed between the associated pairs of vertices, as shown in Figure 2(b). A feasible timetable for this problem can then be achieved by producing a colouring of this graph and mapping the colours to timeslots, as shown in Figure 2(c). The ordering of the timeslots and assignment of lectures to rooms can be taken care of by separate processes.

In reality, timetabling problems usually feature other requirements on top of avoiding clashes. These might include ensuring students' lectures are not too "clumped together", making sure students have a lunch hour, and obeying various staff preferences. Often, automated timetabling methods therefore operate by first producing a graph colouring solution using the required number of colours, and then use specialised operators to "move" from one valid colouring to another to try and remove violations of the remaining constraints. (This has a certainly been a fruitful strategy in many of the International Timetabling Competitions held over the years).

Related to educational timetabling is the problem of constructing sports leagues. In a typical Sunday football league you have a collection of teams, all of whom will want to play each other twice during the year. Setting up this sort of round-robin league is fairly straightforward; but what happens if other constraints are also introduced? For example, nowadays it is common for different teams to share pitches, so we will often see constraints such as, "If team A is playing at home, then team B must play away". Certain fixtures, such as derby matches, might also need to be scheduled to specific weekends to maximise crowd numbers and so on. In fact, these more complicated problems can be modelled through graph colouring in a similar way to university timetabling. Indeed, such techniques have been used to produce professional rugby schedules in both Wales and New Zealand.

Such techniques have been used to produce professional rugby schedules in both Wales and New Zealand

Another place where graph colouring is useful is in solving Sudoku puzzles. I expect most of you know the rules of Sudoku. Whether you love them or hate them, a good Sudoku puzzle should supply sufficient clues so that they are *logic solvable*. In other words, the filled-in cells supplied by the puzzle master should allow you to complete the puzzle without guessing at any point.

Graph colouring algorithms are actually very effective at solving Sudoku puzzles, whether they are logic solvable or not. To do this, as Figure 3 shows using a mini-Sudoku puzzle, we simply map each cell in our grid to a vertex and then add edges between any vertex pairs in the same column, row, or box.

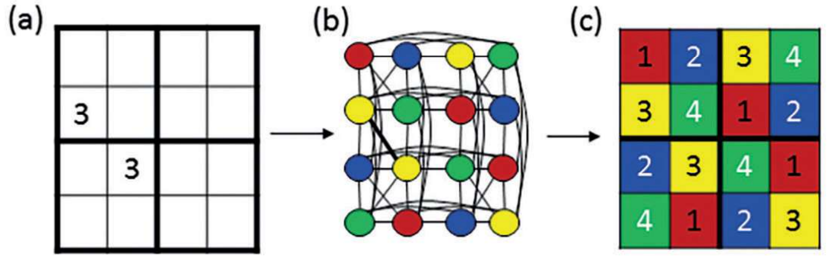


FIGURE 3 - HOW A SUDOKU PUZZLE CAN BE CONVERTED INTO A CORRESPONDING GRAPH COLOURING PROBLEM.

Graph colouring methods form the basis of the optimisation algorithm used on the website www.weddingseatplanner.com

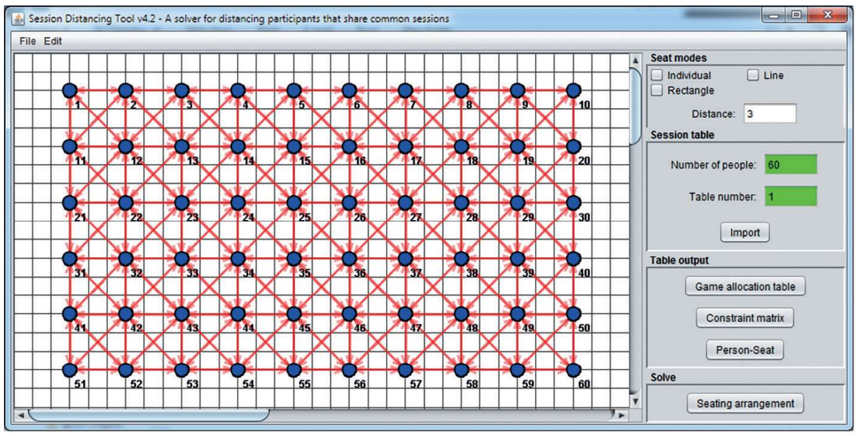
If some cells are filled, we can then also add additional edges – for example, in the figure an edge is also placed between the two vertices whose cells contain a "3". Once we have coloured this graph with the minimum possible number of colours (four in this case), the colours of the vertices then specify the solution.

Happily, some of my own work has shown that graph colouring algorithms can solve very large and difficult Sudoku grids in a matter of milliseconds. Perhaps our days of solving them by hand are over…

We can also design seating plans using graph colouring. Imagine that a school fills its sports hall with desks for the end-of-year exams. To stop copying, the school also wants to ban students from sitting to the left, right, in front of, or behind another student sitting the same paper. The school could go even further and place bans on the diagonals too. The result is a graph colouring problem where the minimum number of colours in the solution indicates the minimum number of different exams that can take place in the venue at the same time. Software for specifying such problems was actually commissioned in 2014 by the Cardiff School of Social Sciences to help them set up controlled experiments using human participants in computing laboratories (see Figure 4).

From a different perspective, now imagine you are in charge of organising a wedding dinner where the guests need to be allocated to tables. Some guests, such as families, will need to be put on the same table, whereas others (divorcees, and so on) might need to be kept apart. How can you keep everyone happy without it turning into a logistical nightmare? Fortunately, graph colouring also lies at the heart of this problem. Indeed, graph colouring methods form the basis of the optimisation algorithm used on the website www.weddingseatplanner.com where such plans can be created for free.

The examples discussed in this article are just a taste of where graph colouring problems can arise in the real world. You may have noticed that at various points in this article I have said something along the lines of "we then produce an appropriate colouring of the graph". Perhaps



Session Distancing Tool v4.2 - A solver for distancing participants that share common sessions

Distance: 3

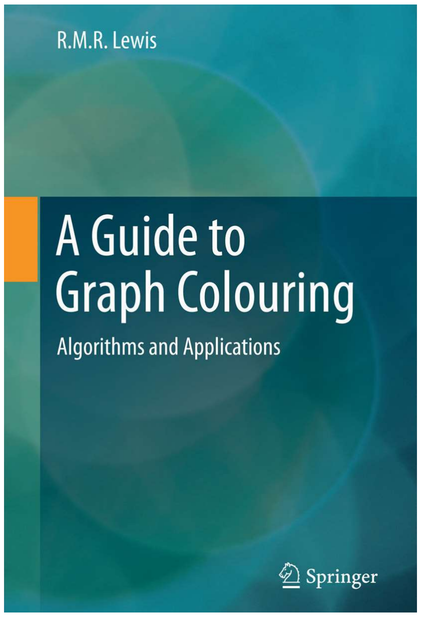
Number of people: 60

Table number: 1

FIGURE 4 - SCREENSHOT OF THE SEATING ALLOCATION TOOL (DESIGNED BY DANIEL WILLIAMS, SCHOOL OF COMPUTING, UNIVERSITY OF SOUTH WALES).

you are now wondering how we actually go about doing this. I have deliberately avoided this pressing issue here, but for now let's just say that unless our graph is very small or belongs to a rather narrow set of types, then we will usually have to resort to heuristic-based approximation algorithms. Many examples of these can be found in my recent book *A Guide to Graph Colouring: Algorithms and Applications* (2015).

Rhyd Lewis is a lecturer in Operational Research at the School of Mathematics, Cardiff University. He continues to confuse blue and purple.



Cover courtesy of Springer © Rhyd Lewis

ADVERTISE IN *IMPACT* MAGAZINE

The OR Society are delighted to offer the opportunity to advertise in the pages of *Impact*. The magazine is freely available at www.theorsociety.com and reaches a large audience of practitioners and researchers across the O.R. field, and potential users of O.R. If you would like further information, please email: advertising@palgrave.com.

Rates

Inside full page: £1000
Inside half page: £700
Outside back cover, full page: £1250
Inside back cover, full page: £1200
Inside front cover, full page: £1200
Inside full page 2, opposite contents: £1150
Inside full page, designated/preferred placing: £1100

Full Page Adverts must be 216x286mm to account for bleed, with 300dpi minimum.