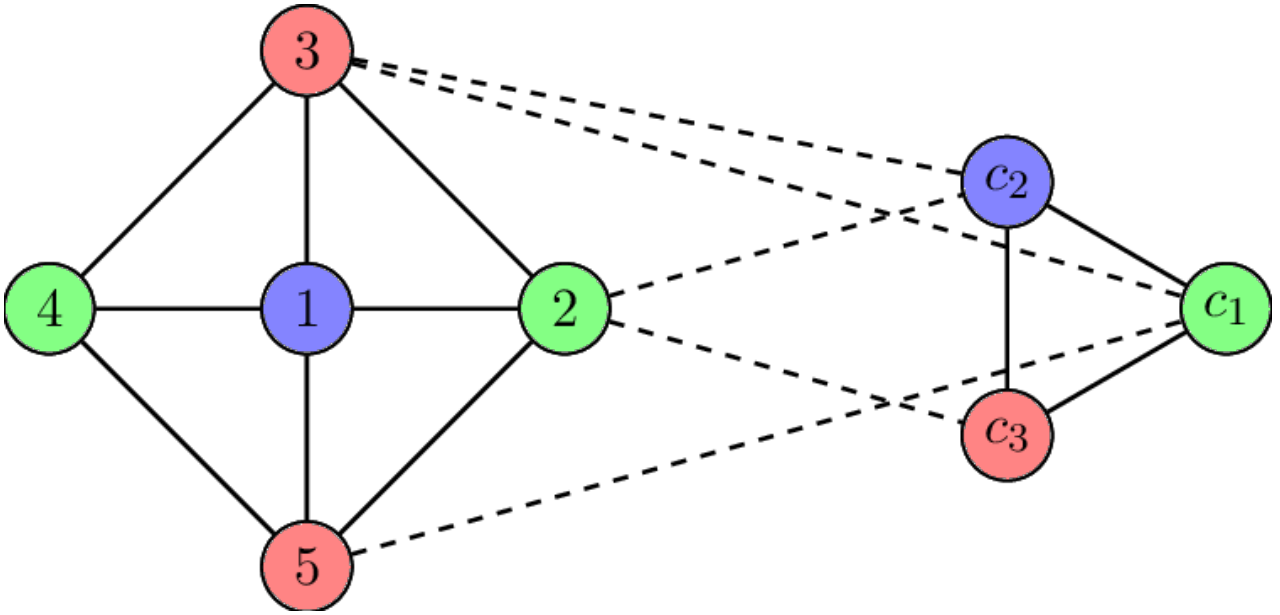
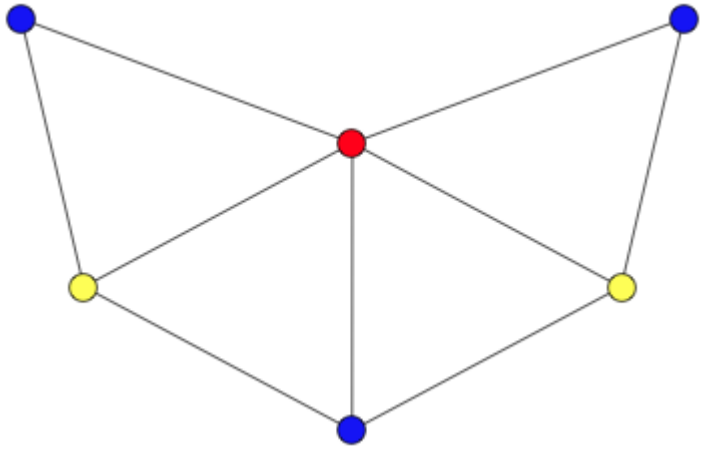


Graph Coloring and its Applications



By Mrinal Kanti Saha
Regd. No. 184415

A little recap...

Definitions :-

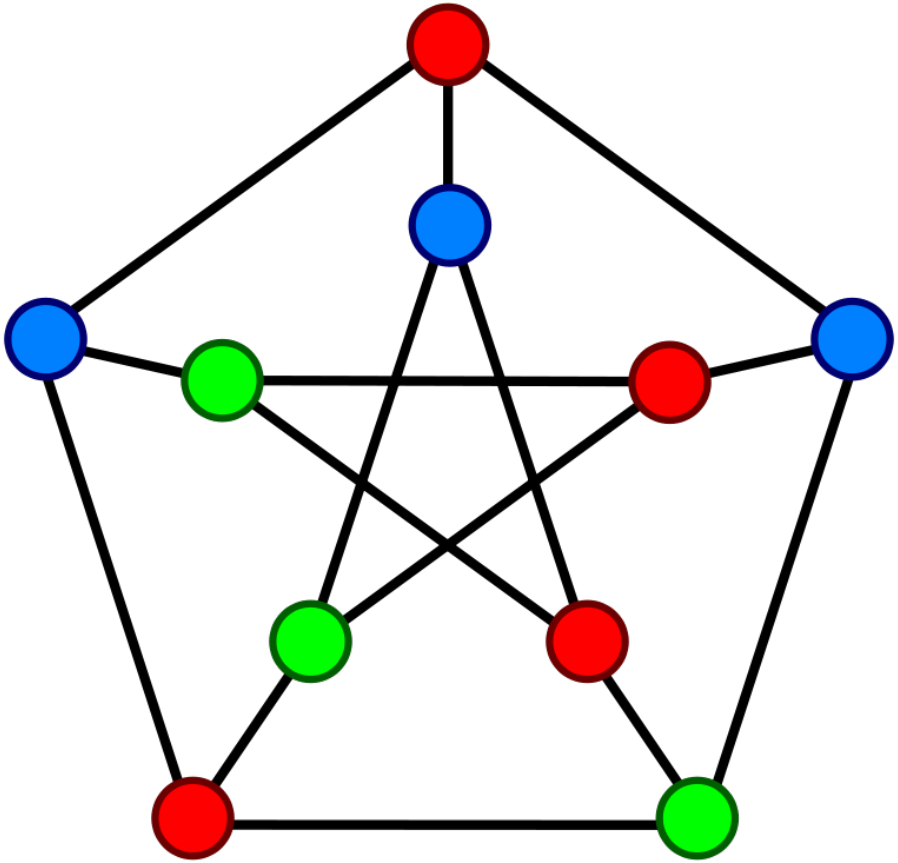
- Graph G – Pair (V(G), E(G)) , where

  V(G) is a non-empty finite set of vertices or points and

  E(G) is a set of unordered pairs of distinct elements of V(G).
- Adjacency – If (u, v) ∈ E(G), the vertex v is adjacent to u.
- Undirected Graphs – If (u, v) ∈ E(G) then (v, u) also ∈ E(G).
- Directed Graphs – If (u, v) ∈ E(G) then (v, u) may or may not ∈ E(G).
- Degree of a Graph – The number of edges incident to the vertex.

Graph coloring

In graph theory, graph coloring is a special case of graph labeling; it is an assignment of labels traditionally called "colors" to elements of a graph subject to certain constraints.



There are basically 3 types of graph coloring (in undirected graphs) :-

(1) Vertex Coloring,

(2) Edge coloring,

(3) Face coloring.

Types of coloring

(1) Vertex coloring **-** A way of coloring the vertices of a graph such that no two adjacent vertices are of the same color.

(2) Edge coloring **-** Assigns a color to each edge so that no two adjacent edges are of the same color.

(3) Face coloring **-** Assigns a color to each face or region so that no two faces that share a boundary have the same color (for maps).

Note : The latter two (edge and face) can be extrapolated into a vertex coloring problem.

History

Timeline :-

- 1852 – **Francis** developed the concept of map coloring in which no two region sharing common boundaries received the same color.
- 1879 – **Kempe** claimed to have proven the 4-color theorem.
- 1890 – **Heawood** proved Kempe's argument wrong.
- 1912 – **Birkhoff** introduced the chromatic polynomial.
- 1976 – **Appel & Haken** finally proved the 4-color theorem.

2002 – **Seymour, Robertson & Thomas** established the strong perfect graph theorem.

Some important terms...

A coloring using at most ***k* colors** is called a (proper) ***k*-coloring**.

Chromatic Number: The smallest number of colors needed to color a graph G.

Chromatic polynomial :-

It counts the number of ways a graph can be colored using no more than a given number of colors.

As the name indicates, for a given *G* the function is indeed a polynomial in *k*.*

* where k is the no. of colors

Algorithms

- NO efficient algorithms available*.
- Only approximate algorithms.
  - NO guarantees on minimum number of colors.
  - Guarantees an upper bound on the number of colors.

* a.k.a. NP-Complete Problem

Greedy Algorithm

Basic Greedy Coloring Algorithm:-

1. Color first vertex with first color.

2. Do following for remaining V−1 vertices.

   a) Consider the currently picked vertex and color it with the lowest numbered color that has not been used on any previously colored vertices adjacent to it.

   b) If all previously used colors appear on vertices adjacent to v, assign a new color to it.

3. Halt.

Heuristics used...

The quality of the resulting coloring depends on the chosen ordering.

If the vertices are ordered according to their degrees, the resulting greedy coloring uses at most colors, at most one more than the graph's maximum degree. Called the **Welsh–Powell** algorithm.

This kind of algorithms are called sequential coloring algorithms.

Applications of Graph coloring

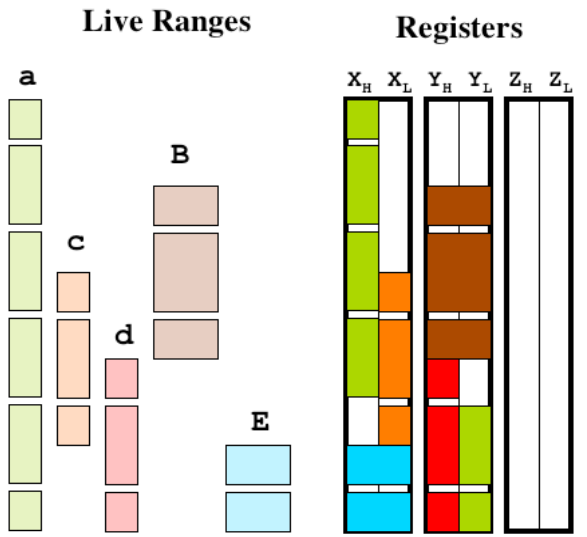
(1) Register Allocation,

(2) Scheduling,

(3) Sudoku,

(4) Map Coloring,

(5) Checking a graph is bipartite or not.



Register Allocation

Understanding Register Allocation

- Register allocation is the process of determining which values should be placed into which registers and at what times during the execution of the program.
- There will always be some pieces of code that require more registers than actually exist. In such cases, the register allocator must insert *spill code* to store some values back into memory for part of their lifetime.
- Minimizing the runtime cost of spill code is a crucial consideration in register allocation.

Register Allocation is done by the compiler

Its corresspondence to Graph Coloring

The recognition of this correspondence is originally due to John Cocke.

First proposed this approach in 1971.

It was first implemented by G. J. Chaitin and his colleagues in 1981.

In the PL.8 compiler for IBM's 801 RISC prototype.

Formulation

- Each node in the graph represents the *live range* of a particular value.

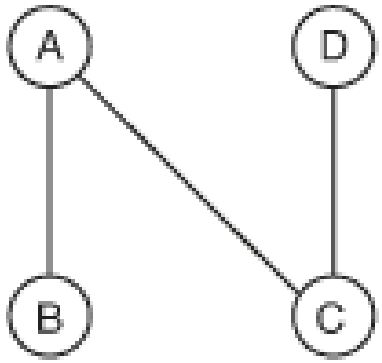
  A live range is defined as a write to a register followed by all the uses of that register until the next write.
- An edge between two nodes indicates that those two live ranges *interfere* with each other because their lifetimes overlap.
- The resulting graph is thus called an *interference graph*.
- The register allocation problem becomes equivalent to coloring the graph using as few colors (registers) as possible.

Interference Graph

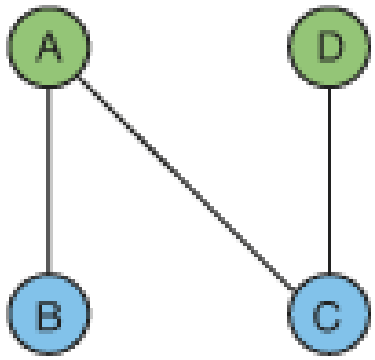
Code Sequence

```
A = ...
B = ...
... B ...
C = ...
... A ...
D = ...
... D ...
... C ...
```

Interference Graph



Colored Graph



Final Allocation

```
R1 = ...
R2 = ...
... R2 ...
R2 = ...
... R1 ...
R1 = ...
... R1 ...
... R2 ...
```

Note : The above graph is only for demonstration purposes.

The graph is never this simple.

Steps

(1) Determine the live ranges.

(2) Construct the interference graph.

(3) Eliminate unneeded copies (Coalesce phase).

(4) Rebuild the graph.

(5) Calculate Spill costs for each node.

(6) Pruning (Simplify phase)

(7) Select phase

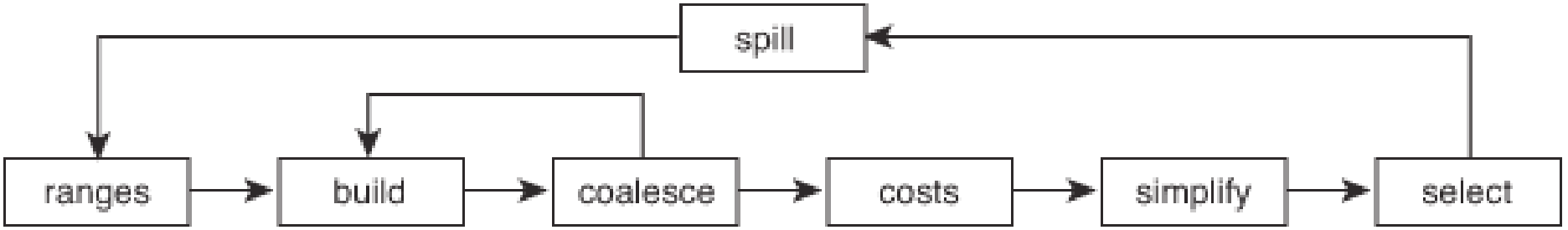


Figure – The graph coloring process.

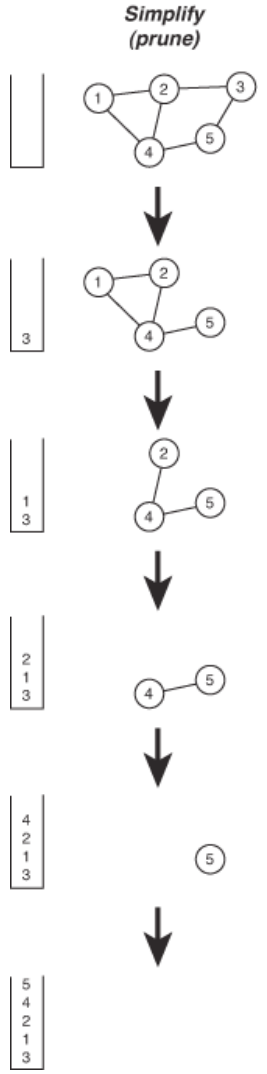
Simplify Phase

Nodes with fewer than k* neighbors are removed.

Each node is removed it is placed on a stack and its edges are removed from the graph.

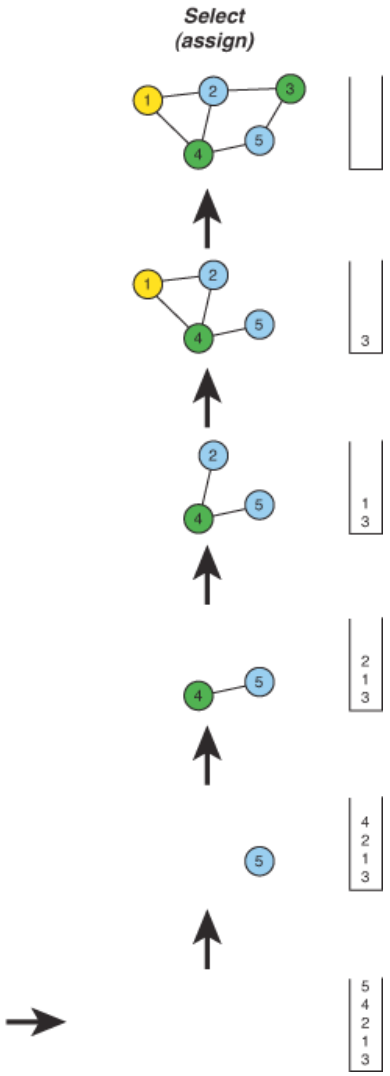
If a point is reached where every node has *k* neighbors (or more), a node is chosen as a possible spill candidate.

Once a spill candidate is chosen, the node is then removed from the graph and pushed onto the stack, just like the others, and the algorithm continues on to the next node.



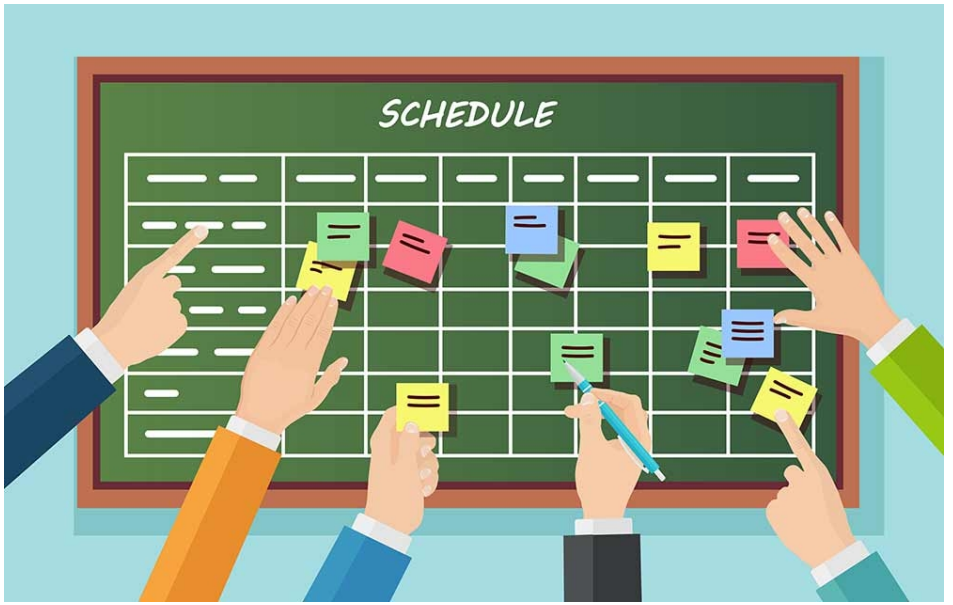
* where k= 3 (here) is the no. of registers available

Select Phase



Popping a node off the stack, re-inserting it into the graph, and assigning it a color different from all of its neighbors.

If a situation occurs where no color is available. the node is marked for spilling and remains uncolored while the algorithm continues on to the next node.



Scheduling

Types of Scheduling Problem

Some of the typical scheduling problems include-

- Timetable Scheduling

a. Course Timetable Scheduling - courses sharing common resources to be scheduled in conflict-free time-slots,

b. Exam Timetable Scheduling - exams sharing common resources to be scheduled in conflict-free time-slots,

- Aircraft Scheduling,
- Job Shop Scheduling.

Course Timetabling

Course Timetabling is the scheduling of a set of related courses in a minimal number of time-slots such that no resource is required simultaneously by more than one event.

In a typical educational institute resources, which may be required by courses simultaneously can be students, classrooms and teachers.

Constraints are the various restrictions involved in creating a schedule.

- No two subjects having common students can be scheduled in the same time-slot,
- No courses can be scheduled to the same classroom at same time-slot,
- More than one course taught by the same teacher cannot be assigned same time of the week.

Formulation

For solving Course Timetable scheduling problems using graph coloring, the problem is first formulated in the form of a graph where courses act as vertices.

Depending on type of graph created, edges are drawn accordingly.

Edges are drawn between conflicting courses having common students, common teachers, common classrooms, etc.

Example

Problem : Given Adjacent table find minimum number of time slots to schedule all the courses without conflict.

List of constraints :

- Courses having common student cannot be allotted at the same time slot onthe same day.
- Total number of available periods is 8.

Sl. No.	List of Honours Subjects	General Subject Combination
1	Physics	Mathematics(compulsory) + Computer Science/Chemistry/Electronics
2	Chemistry	Mathematics(compulsory) + Physics/Computer Science
3	Mathematics	Physics(compulsory) + Chemistry/Computer Science/Statistics
4	Economics	Mathematics(compulsory) + Statistics/Computer Science

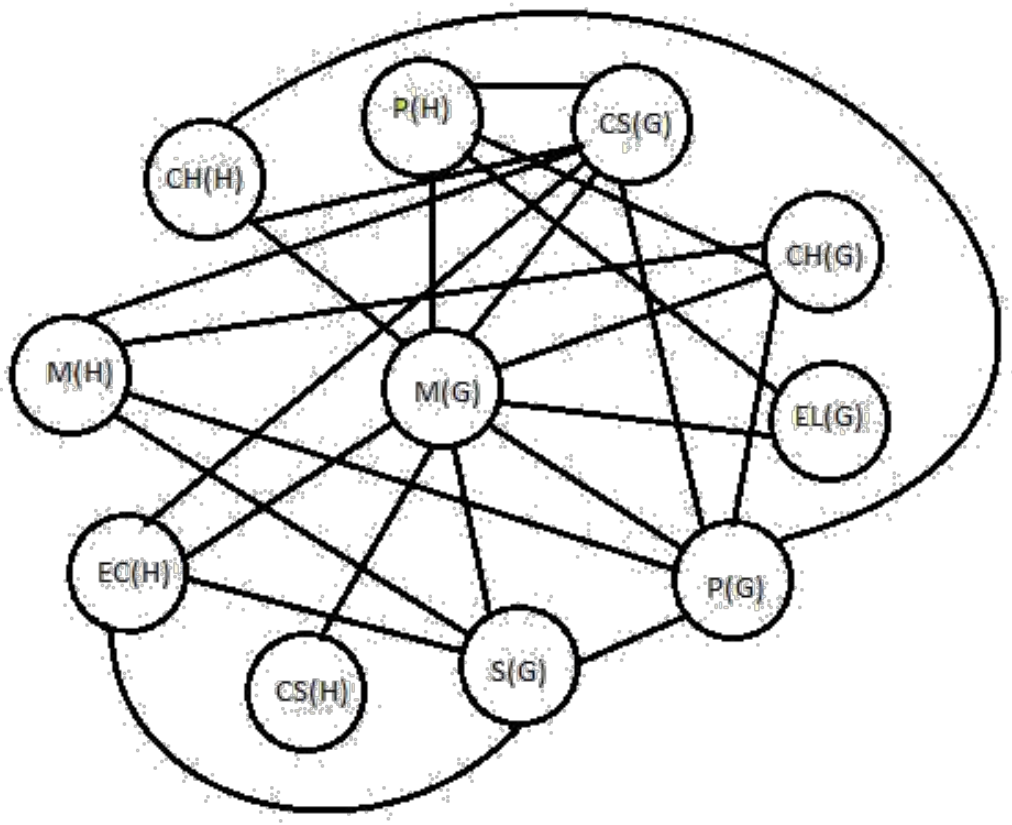
Solution

N.B. (H) and (G) notations represent Honours and General courses respectively.

CH- Chemistry, M- Mathematics,

EC-Economics, CS-Computer Science,

P-Physics, S-Statistics, EL-Electronics.



After applying graph coloring algorithm, the resultant graph in Figure beside is properly colored with chromatic number 4.

Result

In the above solution, the hard constraints are satisfied properly.

The resultant minimum number of time slots needed is 4 which do not exceed the total available 8 periods.

Similarly, no student overlapping between Electronics and Physics General, or students between Chemistry, Computer Science and Statistics General.

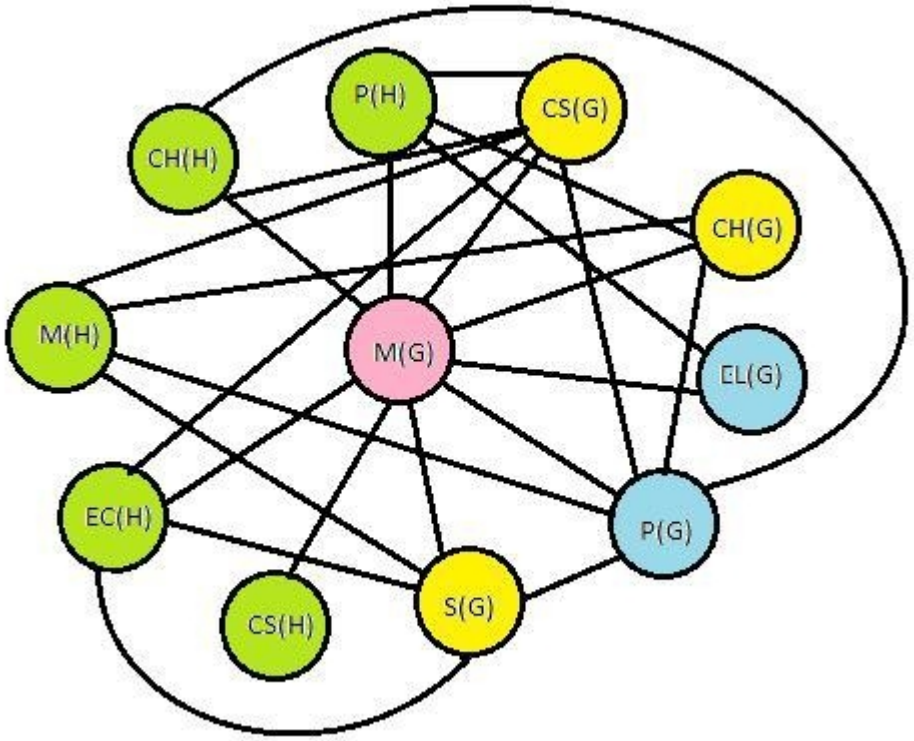


Figure - Colored Conflict Graph

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

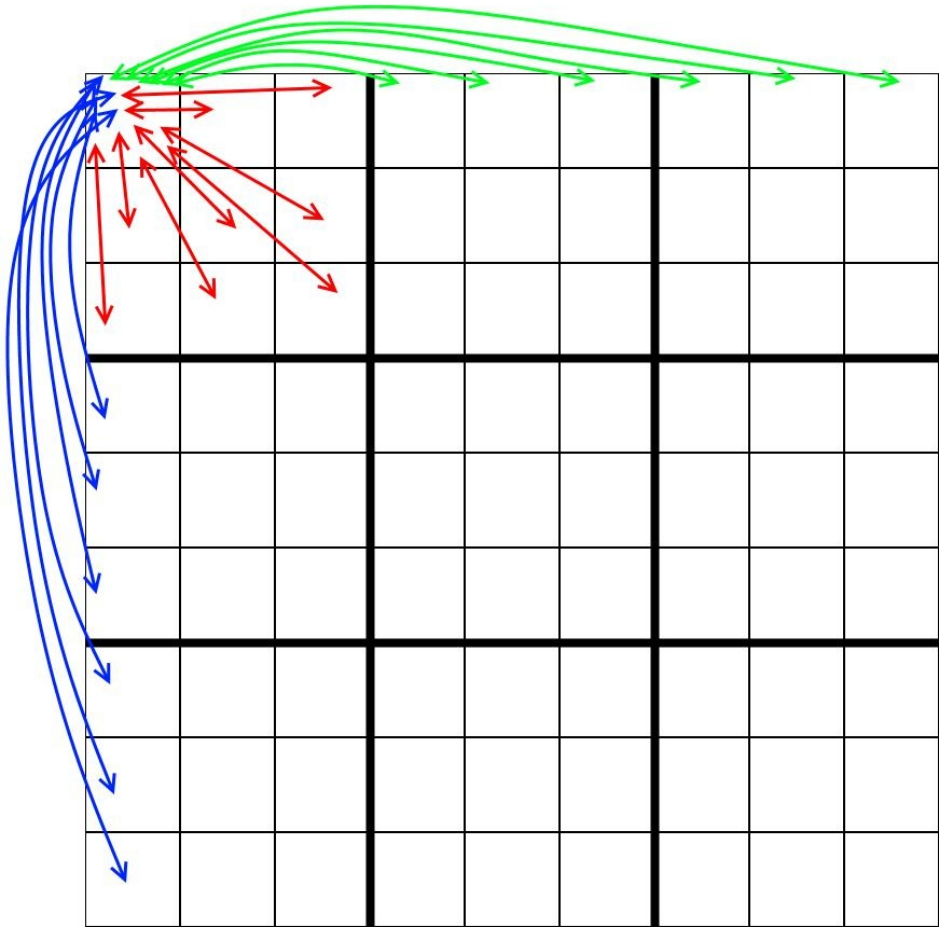
Sudoku

Formulation

Sudoku is one of the most famous number placement-puzzle and it is also a variation of Graph-coloring problem.

Where, every cell represents a vertex.

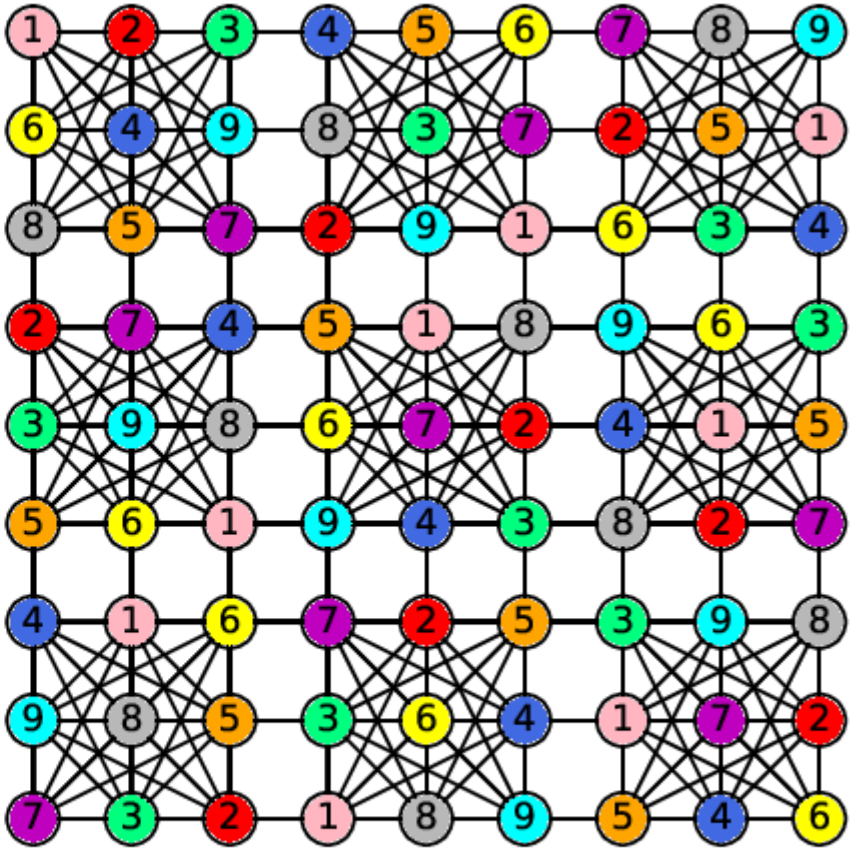
There is an edge between two vertices if they are in same row or same column or same block.



The Sudoku graph

1						7		9
	4				7	2		
8								
	7			1			6	
3								5
	6			4			2	
								8
		5	3				7	
7		2					4	6

(a) Unsolved Sudoku.



(b) Graph coloring of Sudoku.

References

[1] Wikipedia

[2] Geeks for Geeks

[3] Register Allocation By Graph Coloring
(by Jason Robert Carey Patterson)

[4] A Study on Course Timetable Scheduling using Graph Coloring Approach
(by Runa Ganguli and Siddhartha Roy)

[5] Google Images

Thank you...