

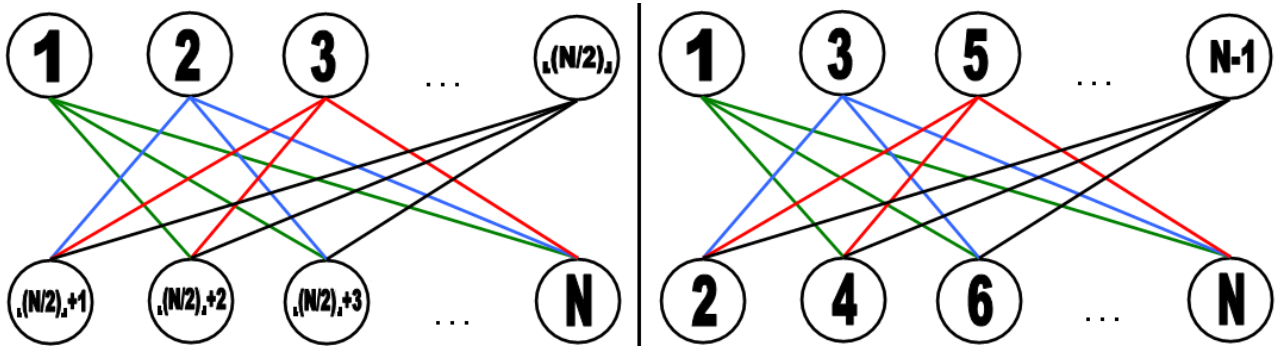
Análisis y Diseño de Algoritmos II – Transformaciones o búsquedas Locales

- Las transformaciones locales o búsqueda local es una estrategia para resolver problemas de optimización sobre espacios de búsqueda exponenciales. La idea básica es:
 - Crear una solución aleatoria S, generalmente mediante heurística.
 - Aplicar a S una transformación de un conjunto de de transformaciones.
 - Elegir la mejor.
 - Repetir lo anterior hasta que ninguna transformación mejore a S.
- Consideraciones:
 - La solución resultante no siempre es optima, depende de la solución aleatoria generada y de las transformaciones aplicadas.
 - Si el conjunto de transformaciones incluye todas las posibles transformaciones, entonces si obtendremos una solución optima, pero su costo seria el mismo a aplicar un algoritmo de fuerza bruta.
 - Esta estrategia solo tiene sentido cuando se puede restringir el tamaño del conjunto de transformaciones y así considerar todas las transformaciones en poco tiempo.
- El coloreo del grafo es un ejemplo en el que se pueden aplicar las transformaciones locales.
Para esto, siguiendo la idea básica, coloreamos el grafo mediante lo que se conoce como coloreo secuencial.

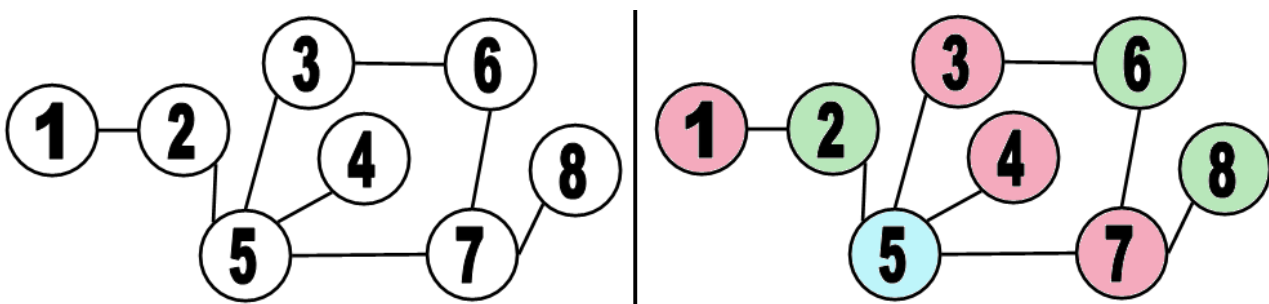
```
/*Se fija si hay algun adyacente del vertice que quiero colorear,
ya coloreado con color*/
bool pueda_colorear(grafo g, int vertice, int color, int Colores[N]){
    list<int> Adyacentes = g.devolver_adyacentes(vertice);
    list<int>::iterator it = Adyacentes.begin();
    while ((Colores[*it] != color) && (it != Adyacentes.end()))
        it++;
    if (it == Adyacentes.end())
        return 1;
    return 0;
}

/*Colores se inicializa en un valor especial '-1'*/
void Coloreo_Secuencial(grafo g, int Colores[N]){
    for (int vertice = 1; vertice <= N, vertice++){
        int color = 1;
        while (!pueda_colorear(g, vertice, color, Colores))
            color++;
        Colores[vertice] = color;
    }
}
```

- Este forma de colorear, generalmente da soluciones muy próximas a las optimas, y en algunos casos la optima, pero en otros da soluciones muy alejadas. El ejemplo típico es el siguiente:



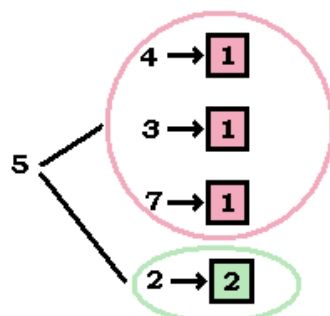
- Para el caso de la izquierda, utilizando coloreo secuencial, se necesitarían tan solo 2 colores, mientras que para el caso de la derecha serian necesarios la parte entera inferior de $(N/2)$ colores.
- Hecha esta aclaración volvemos al problema del coloreo por transformaciones locales. Supongamos el siguiente grafo al cual le aplicamos coloreo secuencial:



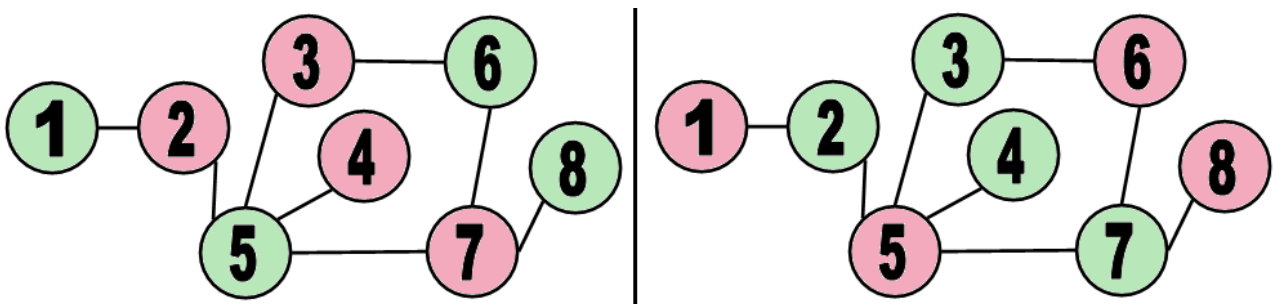
Colores[N] =

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

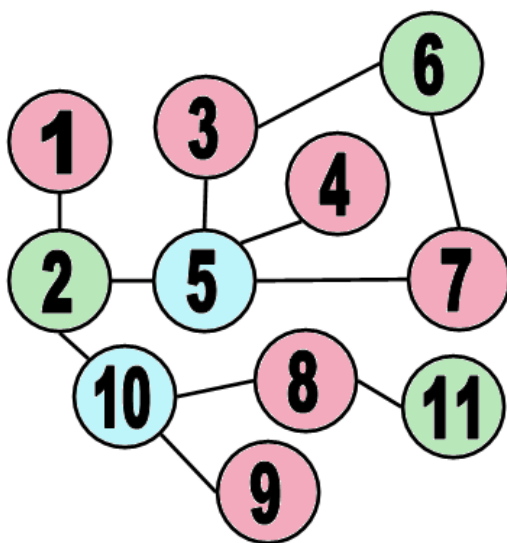
- Vemos claramente en el arreglo Colores, que el problema se encuentra en el vértice 5, el cual está coloreado por un tercer color.
- Lo que debemos hacer es plantear las posibles transformaciones a partir de los adyacentes del vértice 5.



- Tenemos claramente 2 posibles transformaciones, que consistirían en invertir los colores de cada subconjunto de adyacentes.
- Entonces lo primero sería a partir de los adyacentes de 2 invertir los colores de todos sus adyacentes. Osea a 2 ponerle color 1 y a 1 ponerle color 2, luego si es una transformación válida, coloreo 5 con 2 y me quedo con la solución encontrada.
- Después hago lo mismo para el otro conjunto, coloreo 4, 3 y 7 con 2 e invierto todos los colores a partir de sus adyacentes. Si es una transformación válida, coloreo 5 con 1 y me quedo con la mejor solución. Las 2 transformaciones quedarían de la siguiente forma:

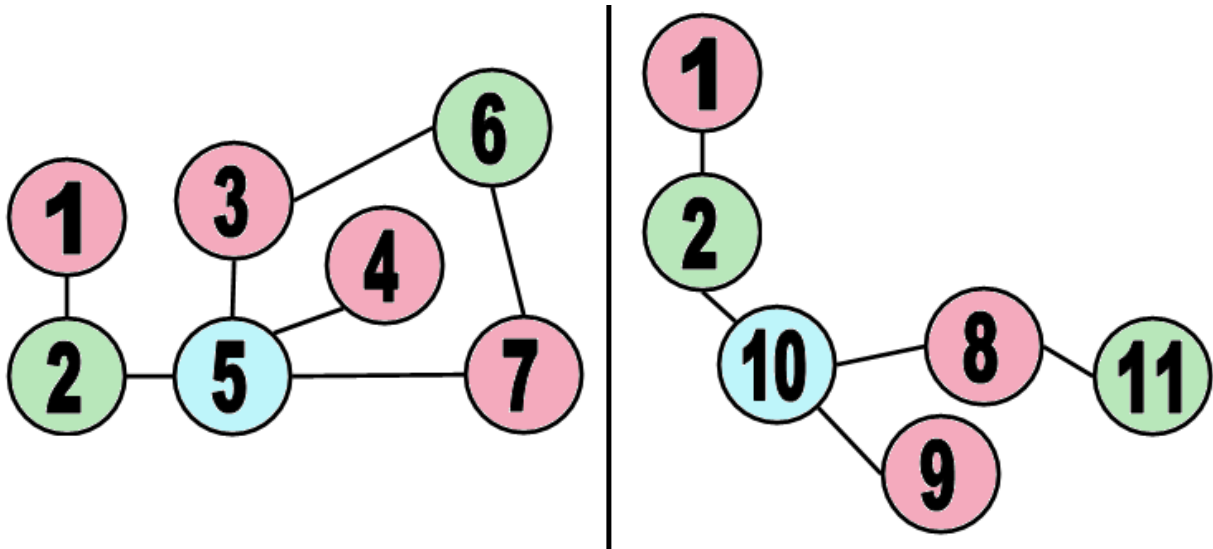


- En este ejemplo cualquiera de las 2 transformaciones es factible y da una solución óptima. Si bien las transformaciones aquí son aplicadas a un grafo coloreado con 2 colores, el problema en general es para mucha más cantidad de colores, y varias transformaciones, pero la idea para resolverlo es la misma.
- Veamos el siguiente grafo coloreado por coloreo secuencial:



- A este ejemplo podemos subdividirlo en 2 problemas, primero el subgrafo del vértice 5 y luego el subgrafo del vértice 10. A cada uno de los subgrafos debemos aplicar la técnica anterior.

- Tener en cuenta que cuando intercambiamos los colores a partir de los adyacentes, el color 3 del vértice 10 no es intercambiado ni tampoco los vértices que son alcanzados a partir de este.
Los 2 problemas de coloreo serian los siguientes:



- Cuando digo que las soluciones sean factibles es si luego de aplicar las transformaciones no quedan vértices adyacentes coloreados con el mismo color.
- Un pseudo del Algoritmo que resuelve el coloreo mediante transformaciones locales:

```
/*pseudo de coloreo por transformaciones locales*/  
/*el arreglo colores esta cargado con coloreo secuencial*/  
void coloreo(grafo g, int Colores[N]) {  
    int ColoresAux[N] = Colores[N];  
    for(c/vertice Vp coloreado con 2 < color <= N)  
        for(c/subconjunto de colores adyacentes (i,j)) {  
            intercambiar_colores(g,Vp,ColoresAux,i,j);  
            if(es_solucion_factible(g,ColoresAux))  
                Colores[N] = ColoresAux[N];  
        }  
}
```