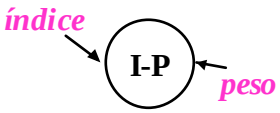
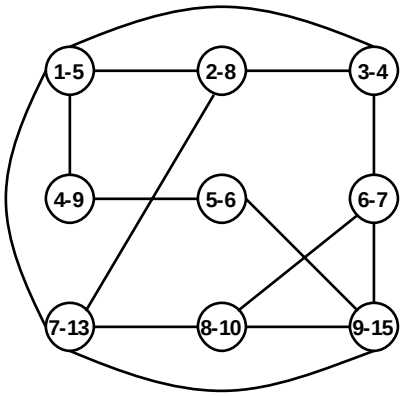
GA em Otimização Combinatorial

- Problemas onde a busca da solução depende da avaliação de diversas combinações (ORDEM) dos elementos considerados
  - **Problema do Caixeiro Viajante**
  - **Problemas de Planejamento**
  - **Problemas de Cronogramas**
  - **Alocação de Salas**
  - **Grafos: Colorir, Particionar, Percorrer**

Problema de Colorir o Grafo

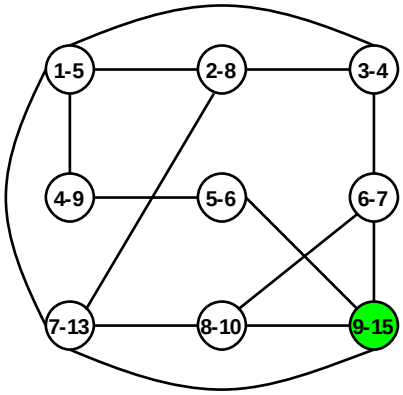




- **Regras:**
  - Colorir os nós do grafo de modo a maximizar a soma total dos pesos.
  - Pares de nós conectados por um arco não podem possuir a mesma cor

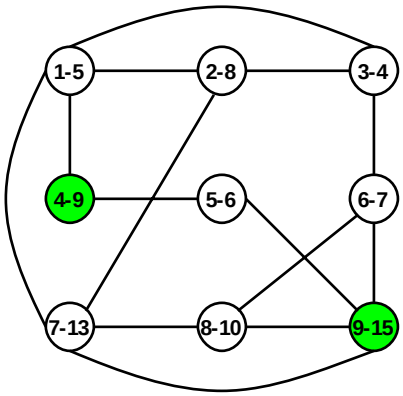
Algoritmo Guloso

- Considera apenas uma das possíveis soluções, colorindo os nós em ordem decrescente de peso.
- Nós em ordem decrescente de peso:
  (9, 7, 8, 4, 2, 6, 5, 1, 3)
- Solução Ótima p/ 1 cor:
  (9, 4, 2) ➨ $\sum P_i = 32$



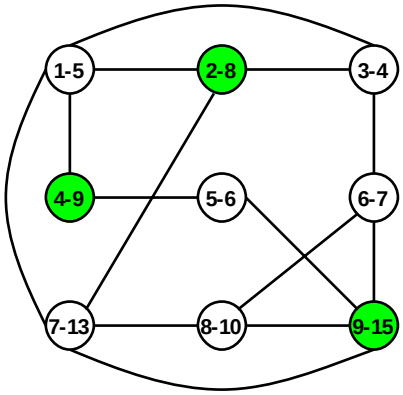
Algoritmo Guloso

- Considera apenas uma das possíveis soluções, colorindo os nós em ordem decrescente de peso.
- Nós em ordem decrescente de peso:
  (9, 7, 8, 4, 2, 6, 5, 1, 3)
- Solução Ótima p/ 1 cor:
  (9, 4, 2) ➨ $\sum P_i = 32$



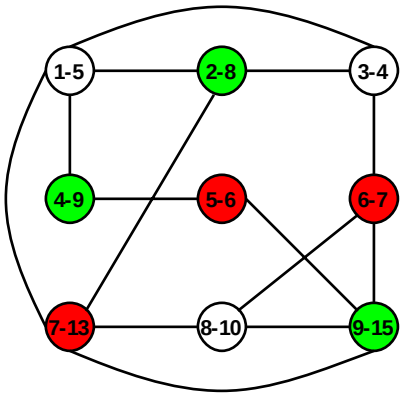
Algoritmo Guloso

- Considera apenas uma das possíveis soluções, colorindo os nós em ordem decrescente de peso.
- Nós em ordem decrescente de peso:
  (9, 7, 8, 4, 2, 6, 5, 1, 3)
- Solução Ótima p/ 1 cor:
  (9, 4, 2) ➨ $\sum P_i = 32$



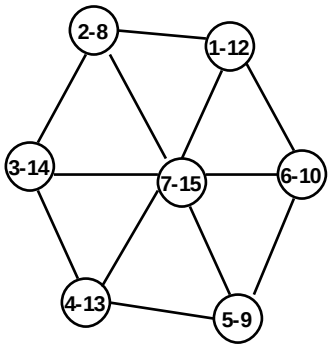
Algoritmo Guloso

- Considera apenas uma das possíveis soluções, colorindo os nós em ordem decrescente de peso.
- Nós em ordem decrescente de peso:
  (9, 7, 8, 4, 2, 6, 5, 1, 3)
- Solução Ótima p/ 2 cores:
  (9, 4, 2) e (7, 6, 5)
  ➨ $\sum P_i = 32 + 26$



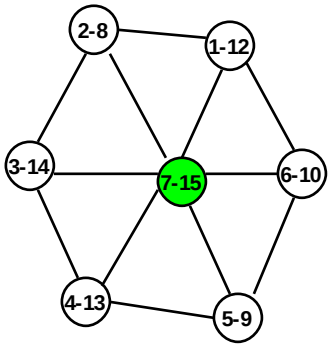
Aplicando o Algoritmo Guloso

- Nós em ordem decrescente de peso:
  (7, 3, 4, 1, 6, 5, 2)



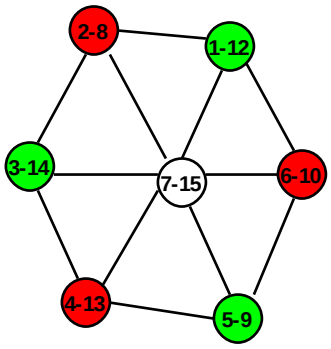
Aplicando o Algoritmo Guloso

- Nós em ordem decrescente de peso:
  (7, 3, 4, 1, 6, 5, 2)
- Algoritmo Guloso começa e para no nó 7



Aplicando o Algoritmo Guloso

- Nós em ordem decrescente de peso:
  (7, 3, 4, 1, 6, 5, 2)
- Soluções Ótimas:
  - 1 cor: (1, 5, 3)
  - 2 cores: (2, 4, 6)



Características do Problema

- A modificação dos pesos, número de cores e arcos, altera radicalmente a solução do problema.
- Estratégias como o algoritmo guloso não funcionam bem para todos os problemas de colorir o grafo.
- Heurísticas (Ex: avaliar número de arcos ou pesos de nós vizinhos antes colorir um nó) podem não ser eficientes para grandes espaços de busca.
- Algoritmos Genéticos oferecem uma solução (sub-ótima ou ótima) para qualquer problema de colorir o grafo

Componentes de um Algoritmo Genético

1. Problema
2. Representação
3. Decodificação
4. Avaliação
5. Operadores
6. Técnicas
7. Parâmetros

Tentando a Representação Binária

- Cada nó do grafo é representado por um campo (gene) no cromossoma:

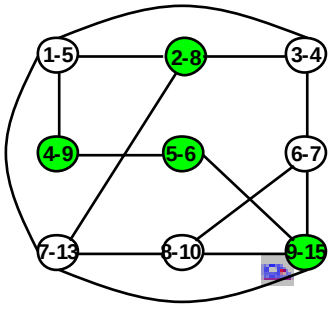
  Símbolo do Campo ≡ Cor do nó

- Podemos usar a representação binária. Para apenas 1 cor :

  0 ≡ não colorido 1 ≡ colorido

Cromossoma ≡ **0 1 0 1 1 0 0 0 1**

Nó *1 2 3 4 5 6 7 8 9*



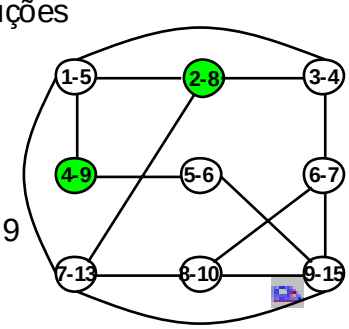
Avaliando Representação Binária

- C é um cromossoma **ILEGAL**.
- Inicialização, crossover e mutação vão gerar soluções ilegais.
- Seria necessário um módulo reparador de cromossomas
- Representação binária permite soluções sub-ótimas:

C ≡ **0 1 0 1 0 0 0 0 0**

Nó *1 2 3 4 5 6 7 8 9*

- C ainda poderia ter o nó 6, ou 8 ou 9 colorido e ser legal.



Representação Baseada em Ordem

- GA Híbrido ≡ Técnicas de GA + Algoritmo Guloso
- **Algoritmo Guloso**:
  - Cria uma lista de nós (ordem decrescente de peso)
  - Constrói a solução: atribui ao próximo nó da lista uma cor legal
- **Algoritmo Genético**
  - Cria uma lista (nós em ordem qualquer)
  - Constrói a solução: atribui ao próximo nó da lista uma cor legal

Exemplo

- Cromossoma = lista

  $C_1 \equiv (9, 7, 8, 4, 2, 6, 5, 1, 3)$

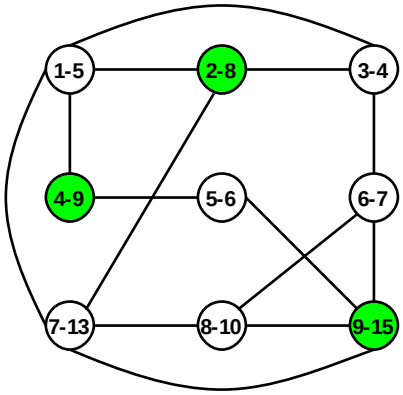
  $C_2 \equiv (2, 3, 7, 4, 9, 6, 5, 1, 8)$

  $C_3 \equiv (4, 5, 1, 2, 9, 6, 8, 7, 3)$

- $C_1\ C_2\ C_3$ resultam na solução ótima p/ 1 cor:

  $(9, 4, 2) \Rightarrow \sum P_i = 32$

- Informação codificada é a **ordem relativa dos nós**



Operadores Genéticos

- Testando o Crossover de 1 ponto:

$P_1 \equiv (9, 7, 8, 4, 2, 6, 5, 1, 3)$

$P_2 \equiv (2, 6, 7, 4, 9, 3, 5, 1, 8)$

$F_1 \equiv (9, 7, \mathbf{8}, 4, 2, 3, 5, 1, \mathbf{8})$

$F_2 \equiv (2, \mathbf{6}, 7, 4, 9, \mathbf{6}, 5, 1, 3)$

- Descendentes são cromossomas ilegais: nós repetidos e ausência de determinados nós.
- Crossover e mutação devem garantir uma lista válida de todos os nós

Modelagem do Algoritmo Genético

1. Problema
   - Problema de Colorir o Grafo
2. Representação
   - Permutação dos índices dos nós
3. Decodificação
   - Da esquerda para direita, atribui uma cor válida ao próximo nó
4. Avaliação
   - ΣP_i
5. Operadores
   - Crossover Uniforme Baseado em Ordem
   - Mutação por Embaralhamento

Crossover Uniforme Baseado em Ordem

- Dados dos genitores P_1 e P_2, criar descendente F_1;
- Gere um padrão de bits do mesmo comprimento que os genitores;
- Preencha F_1, copiando o genitor P_1 nas posições em que o padrão é igual a "1";
- Faça uma lista dos elementos de P_1 associados com os bits "0" do padrão;
- Permute estes elementos de modo que eles apareçam na mesma ordem em que aparecem em P_2;
- Preencha as lacunas de F_1 com os elementos ordenados no passo anterior;

Exemplo

P_1	1	2	3	4	5	6	7	8
P_2	8	6	4	2	7	5	3	1
Padrão	0	1	1	0	1	1	0	0
F_1	-	2	3	-	5	6	-	-
F_2	8	-	-	2	-	-	3	1

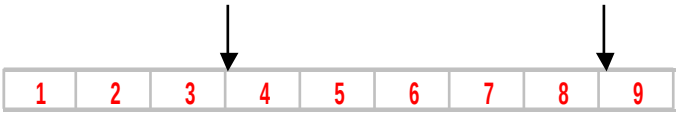
Elementos de P_1 associados a "0": 1, 4, 7, 8. Ordenados segundo P_2: 8, 4, 7, 1

Elementos de P_2 associados a "1": 6, 4, 7, 5. Ordenados segundo P_1: 4, 5, 6, 7

F_1	8	2	3	4	5	6	7	1
F_2	8	4	5	2	6	7	3	1

Mutação por Embaralhamento

- Seleciona aleatoriamente uma sub-lista do cromossoma
- Embaralha sub-lista



GA6-1

- **Módulo de Avaliação**
  - Função de Avaliação: *Avaliador do problema de colorir o grafo* $\sum P_i$
- **Módulo de População**
  - Técnica de Representação: *Lista de nós*
  - Técnica Inicialização da População: *Permutação aleatória*
  - Técnica Eliminação da População: *Elimina o último*
  - Técnica de Reprodução: *Steady State s/ duplicados*
  - Gap: *Testar de 5 em 5*
  - Técnica de Seleção de Genitores: *Roleta*
  - Técnica de Aptidão: *Normalização Linear (100 a 1)*
  - Técnica de Parametrização: *Interpolar taxa de incremento* ***(0,2 a 1,2)***
  - *Population Size:* ***100***
  - *Total de Indivíduos:* ***4000***
- **Módulo de Reprodução**
  - Técnica de Seleção de Operadores: *Roleta*
  - Operadores: *Crossover Uniforme Baseado em Ordem*
  - *Mutação por Embaralhamento*
  - Técnica de Parametrização: *Interpolar Pesos dos Operadores*
  - *de (60 40) a (30 70)*

Problema de Colorir o Grafo

- Grafo com 100 nós ➨ 100! permutações diferentes
- 3 cores possíveis
- Listagem descreve o grafo através de:
  (índice_nó, peso (lista de nós conectados))

(1 62 (20 58 74 82))
(2 183 (6 12 20 28 29 32 51 53 70 79 84 94))
(3 247 (18 24 33 50 88 92))
……
……
(99 254 (29 52 53 67 75 80 84 89))
(100 145 (15 20 22 29 34 44 60 87))

Busca Aleatória

- Muitas vezes não temos como comparar os resultados obtidos por um GA.
- Nestes casos, podemos usar a busca exaustiva como base de comparação.
- Gera-se uma curva média de desempenho para a busca aleatória com o mesmo número de tentativas que o GA.
- Um modelo de GA desempenhando abaixo da busca exaustiva deve provavelmente conter erros de modelagem e/ou programação.

Busca Aleatória

```
procedure busca_aleatória
begin
    t = 0                              primeira geração
    inicializa P(t)                    população inicial aleatória
    avalia P(t)                        calcula f(i) p/ cada indivíduo
    salva_melhor de P(t)               salva melhor indivíduo
while (not total_indivíduos) do
    begin
        t = t + 1                      próxima geração
        inicializa P(t)                população aleatória
        avalia P(t)                    calcula f(i) p/ cada indivíduo
        compara_melhor P(t) com melhor P(t-1)
        salva_melhor
    end
end
```

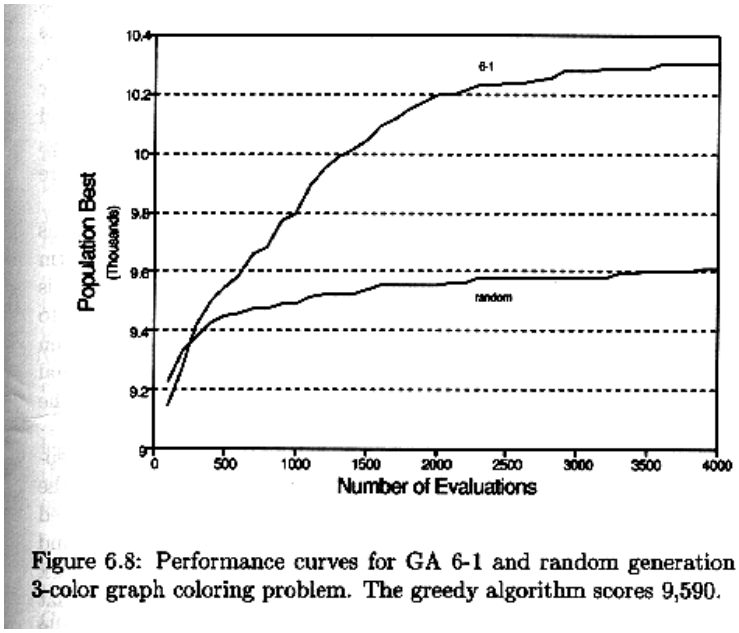


Figure 6.8: Performance curves for GA 6-1 and random generation on a 3-color graph coloring problem. The greedy algorithm scores 9,590.

Resultados do GA 6-1

- GA 6-1 é em média 7,4% que a busca aleatória após 4000 tentativas:
  - média GA 6-1= **10300**; média busca aleatória=**9600**
- GA 6-1 é em média 7,4% que a Algoritmo Guloso:
  - máximo do Alg. Guloso= **9590**
- GA 6-2 com pop_size=1200 e total_indivíduos=10000 encontrou avaliação=**10594**