Zastosowanie łańcuchów ograniczających do poprawnego kolorowania grafu

Antoni Szustakowski

Poprawne kolorowanie grafu

Dany jest graf G=(V,E) (V - zbiór wierzchołków, E - zbiór krawędzi). Niech n oznacza liczbę wierzchołków. **Poprawnym k-kolorowaniem grafu G** nazwiemy:

$$x : V \to \{1, ..., k\}^n, \forall e \in E, \forall v, w \in e : x(v) \neq x(w)$$

Do późniejszych rozważań wprowadźmy również zbiór kolorów C, na którym będziemy operować.

Czyli jeśli dwa wierzchołki są połączone krawędzią - nie mogą mieć tego samego koloru.

Liczba chromatyczna

Liczbą chromatyczną grafu G, nazwiemy minimalną liczbę kolorów k, dla której istnieje poprawne k-kolorowanie grafu G.
Twierdzenie (ograniczenie górne na liczbę chromatyczną):
G - dowolny graf, $\chi(G)$ - liczba chromatyczna grafu G, $\Delta(G)$ - maksymalny stopień grafu G, wtedy:

$$\chi(G) \leq \Delta(G) + 1$$

Cel i standardowe metody

Celem projektu jest próbkowanie z rozkładu jednostajnego określonego na poprawnych kolorowaniach danego grafu. Standardowym podejściem w takim problemie jest skorzystanie z Próbnika Gibbsa oraz algorytmu Coupling From the Past (CFTP).

Algorytm Proppa-Wilsona

Algorytm Proppa-Wilsona pozwala na dokładne próbkowanie z zadanego rozkładu stacjonarnego. Jego standardowa wersja przebiega następująco. Niech t=1,2,4,...

- wystartuj Łańcuch Markowa z każdego możliwego stanu
- łańcuchy przechodzą do kolejnego stanu zgodnie z przekształceniem losowym opartym na zmiennych z rozkładu jednostajnego na [0,1]
- jeżeli po czasie t wszystkie łańcuchy zatrzymają się w tym samym stanie: zakończ
- w przeciwnym wypadku: zwiększ t dwukrotnie

Skuteczność algorytmu potwierdza poniższe twierdzenie.

Twierdzenie o wyniku algorytmu Proppa-Wilsona

Twierdzenie(o wyniku algorytmu Proppa-Wilsona):

Niech P będzie macierzą przejścia nieprzywiedlnego i nieokresowego łańcucha Markowa z przestrzenią stanów $S = \{s_1, s_2, ..., s_n\}$ i rozkładem stacjonarnym $\pi = (\pi_1, ..., \pi_n)$. Niech Φ będzie przekształceniem dla macierzy P z algorytmu Proppa-Wilsona. Rozważmy algorytm Proppa-Wilsona z $(N_1, N_2, ...) = (1, 2, 4, 8, ...)$. Przypuśćmy, że algorytm zbiega z prawdopodobieństwem 1 i niech Y będzie wynikiem tego algorytmu. Wtedy dla każdego $i \in 1, ..., n$ zachodzi:

$$P(Y = s_i) = \pi_i$$

Próbnik Gibbsa

Za przejścia pomiędzy kolejnymi stanami w algorytmie CFTP będzie odpowiadał Próbnik Gibbsa.

Przypuśćmy, że możemy bardzo łatwo uzyskać prawdopodobieństwa warunkowe $\pi(X(v) = i | X(V \setminus \{v\}) = \xi)$ dla wszystkich kombinacji $v \in V$, $i \in C$ oraz $\xi \in C^{V \setminus \{v\}}$. Schemat postępowania w t+1 iteracji wygląda następująco:

- Wylosuj jednostajnie wierzchołek $v \in V$
- Dla pozostałych wierzchołków:

$$X_{t+1}(w) = X_t(w), \forall w \in V \setminus \{v\}$$

- Dla wierzchołka v:

$$X_{t+1}(v) \stackrel{d}{=} \pi(\cdot | X_t(V \setminus \{v\}))$$

Prawdopodobieństwa przejścia

Na zajęciach (kartka 8) pokazaliśmy, jakie prawdopodobieństwa przejścia ma tak skonstruowany łańcuch:

p(x,y)=$\frac{\pi(x_u^{'}|x_1,x_2,\ldots,x_{u-1},x_{u+1},\ldots,x_{|V|})}{|V|}$, gdy $x_u^{'} \neq x_u$ i na pozostałych współrzędnych $x = y$

p(x,y)=$\frac{\sum_{u=1}^{|V|}\pi(x_u|x_1,x_2,\ldots,x_{u-1},x_{u+1},\ldots,x_{|V|})}{|V|}$, gdy $x = y$

p(x,y)=0, w przeciwnym wypadku

Ergodyczność łańcucha

Otrzymany w ten sposób łańcuch jest:

- nieprzywiedlny - każdy stan możemy uzyskać z niezerowym prawdopodobieństwem
- nieokresowy - z niezerowym prawdopodobieństwem możemy zostać w tym samym stanie
- o skończonej przestrzeń stanów

Ostatecznie $\{X_t\}$ jest ergodyczny, zatem ma rozkład stacjonarny π, z uwagi na dobrane prawdopodobieństwa przejścia i zbiega do niego przy $t \to \infty$. Zatem warunek z twierdzenia o wyniku algorytmu Proppa-Wilsona jest spełniony i jesteśmy w stanie próbkować z rozkładu π.

Problem z CFTP

W klasycznej wersji algorytm CFTP jest niezmiernie złożony obliczeniowo i pamięciowo. Jest to szczególnie widoczne dla tak złożonego problemu jakim jest kolorowanie grafu, sama liczność przestrzeni stanów to $|C^V|$, niemożliwym jest przeprowadzenie CFTP w sensownym czasie, jak sobie poradzić z tym problemem?

Inny sposób na zapisanie warunku stopu CFTP

W algorytmie CFTP warunkiem stopu jest zatrzymanie się wszystkich łańcuchów w danym stanie. CFTP wykorzystuje przekształcenie losowe $\Phi : C^V \times [0,1] \rightarrow C^V$, zatem, aby warunek stopu był spełniony wystarczy, by od pewnego miejsca w czasie -t:

$$\Phi(X_{-t}(s), U_{-t}) = const., \forall s \in C^V$$

Czyli wystarczy, aby przekształcenie losowe od pewnego miejsca było stałe.

Łańcuchy ograniczające

Wprowadźmy 2 łańcuchy: X i Y. X będzie zawierał informację o kolorze danego wierzchołka, natomiast Y o wszystkich możliwych kolorach dla danego wierzchołka.

Przestrzenie stanów

Dane są 2 zbiory: C i V. Zbiór C odpowiada możliwym wartościom, natomiast zbiór V - wymiarom. Bazową przestrzenią stanóW dla X będzie $\Omega \subseteq C^V$. Natomiast dla Y - $(2^C)^V$.

Definicja

Definicja 1:
Powiemy, że Y jest *łańcuchem ograniczającym dla X*, jeśli:

$$X_t(v) \in Y_t(v) \forall v \in V \Rightarrow X_{t+1}(v) \in Y_{t+1}(v) \forall v \in V$$

Zastosowanie łańcuchów ograniczających

Główne operacje będą wykonywane na łańcuchu Y:

- sprawdzamy wszystkie możliwe kolorowania dla losowo wybranego wierzchołka
- jeden z wybranych kolorów dla Y będzie pokolorowaniem wierzchołka odpowiadającego w X
- powtarzamy, dopóki dla wszystkich wierzchołków liczba możliwych pokolorowań będzie 1

Startowanie łańcuchów

Chcemy spełnić definicję 1. łańcucha ograniczającego w każdym obrocie pętli. Zatem:

- $Y_0(v) = C, \forall v \in V$, wtedy mamy pewność (dzięki konstrukcji algorytmu), że łańcuch Y zawsze będzie ograniczał łańcuch X
- $X_0(v) = 1, \forall v \in V$.

Algorytm

Niech n oznacza liczbę wierzchołków w grafie G=(V,E), Δ - maksymalny stopień G, $t \in \mathbb{N}$ - aktualną iterację, it - maksymalną liczbę iteracji, a k - maksymalną liczbę kolorów. Głównym algorytmem dla zadanego problemu będzie:

Algorytm

1. **Dla t=1,2,...,it**:
2. **Jeżeli** $\forall v \in V : |Y_{t-1}(v)| = 1$:
   - Przerwij i zwróć $X_{t-1}(v)$.
   - W przeciwnym wypadku przejdź do punktu 3.
3. **Niech** $Y_t = Y_{t-1}$ oraz $X_t = X_{t-1}$.
4. **Wylosuj jednostajnie** $v \in [n]$, niech N_v oznacza zbiór sąsiadów v.
5. **Jeżeli** $|Y_t(v)| = 1$:
   - Wróć do punktu 3.
   - W przeciwnym przypadku przejdź do punktu 5.
6. **Niech** $Y_t(v) \leftarrow \varnothing$.

Algorytm

7. **Dopóki** $c \notin \bigcup_{w \in Nv} Y_t(w)$ lub $|Y_t(v)| > \Delta$:
8. **Wylosuj jednostajnie** $c \in [k]$
9. **Jeżeli** $\forall w \in Nv : Y_{t+1}(w) \neq \{c\}$:
   - $Y_t(v) \leftarrow Y_t(v) \cup \{c\}$
10. **Niech** $X_t(v) = c$
11. **Jeżeli** warunek z punktu 2. dla iteracji o numerze it nie jest spełniony:
    - it $\leftarrow 2 * it$
    - **Powtórz** algorytm od samego początku

Uzasadnienie formalne

W swojej pracy [2] Haggstrom i Nelander podają twierdzenie wraz z dowodem, kiedy powyższy algorytm będzie zbiegał oraz co będzie zwracał.

Twierdzenie

Dla dowolnego $a \in (2^C)^V$ oznaczmy $card(a)$ jako liczbę elementów $s \in C$, takich, że $s \in a$. Jeżeli istnieje $n < \infty$, takie, że:

$$P(\forall v \in V : card(Y_n(v)) = 1 | \forall v \in V : Y_0(v) = C) > 0$$

To powyższy algorym P-prawie na pewno zbiega i zwraca nieobciążoną próbkę z rozkładu π.

Liczba kolorów

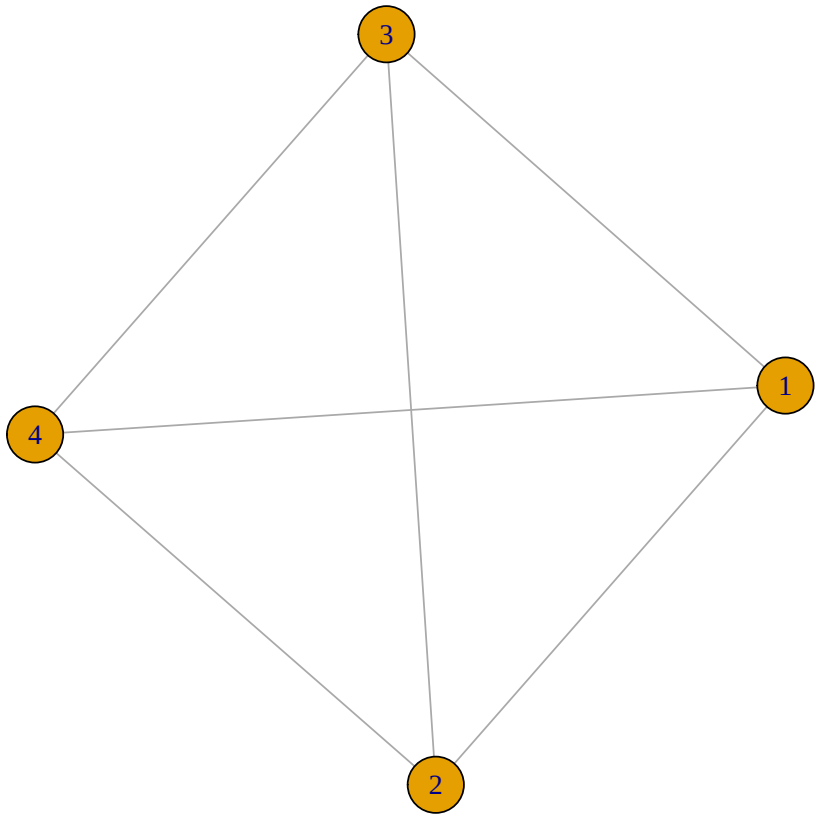
Dobranie odpowiedniej liczby kolorów jest kluczowe, aby łańcuchy przechodziły z jednego stanu do istotnie różnego kolejnego stanu. Jeśli dobierzemy ich zbyt mało, kolorowanie będzie niemożliwe lub algorytm będzie wyłącznie losowo dobierał kolory. Natomiast jeśli dobierzemy ich zbyt dużo - z dużym prawdopodobieństwem każdy wierzchołek zostanie pokolorowany na inny kolor. Dlatego na podstawie pracy Hubera [1] zastosowałem 2 liczby:

- $k > \frac{11\Delta}{6}$
- $k \geq \Delta(\Delta + 2)$

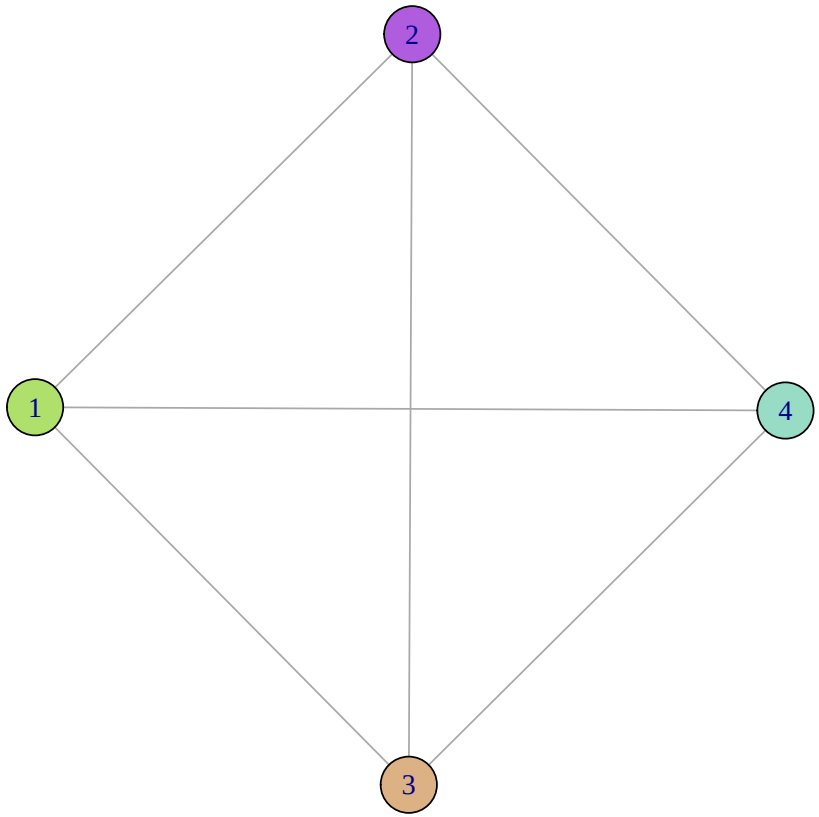
Działanie algorytmu na konkretnych grafach

Zobaczmy, jak działa algorytm dla konkretnych grafów:

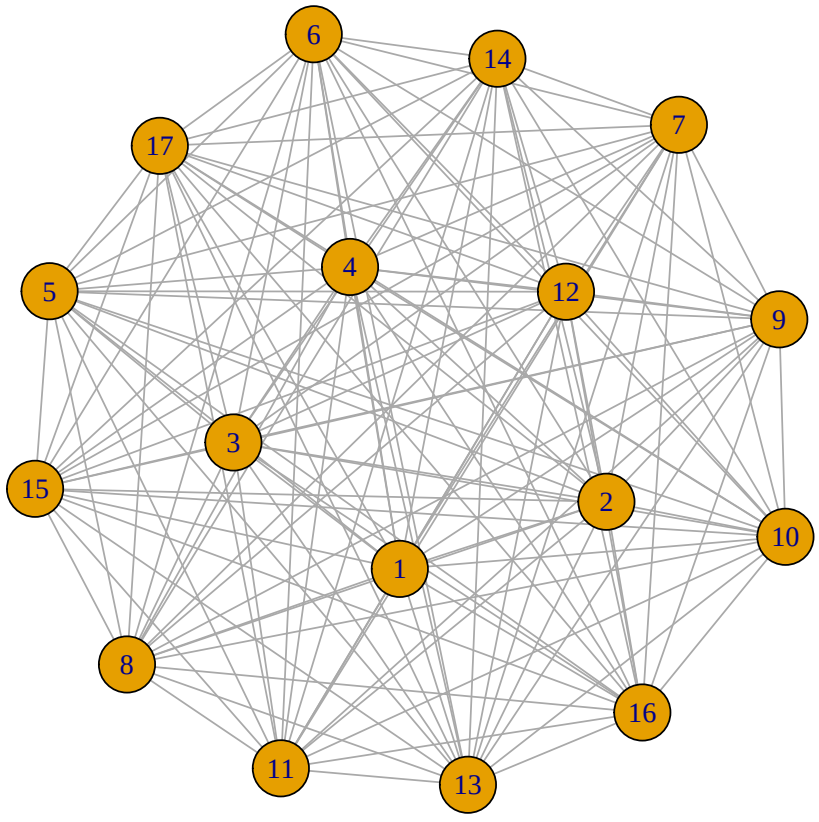
Niewielki graf pełny



Pokolorowany niewielki graf pełny

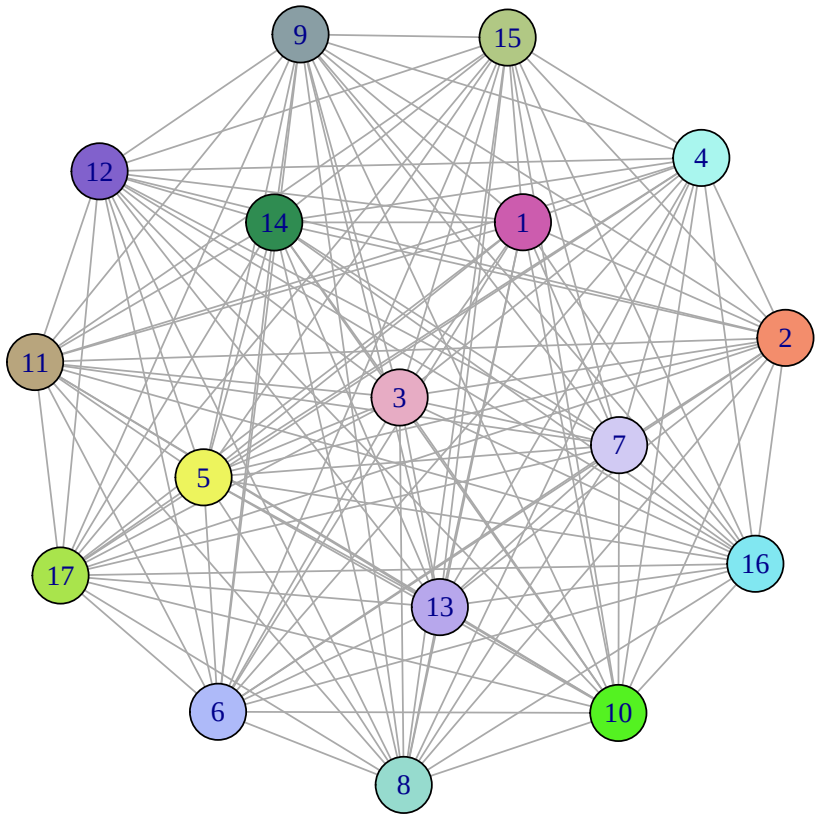


Duży graf pełny

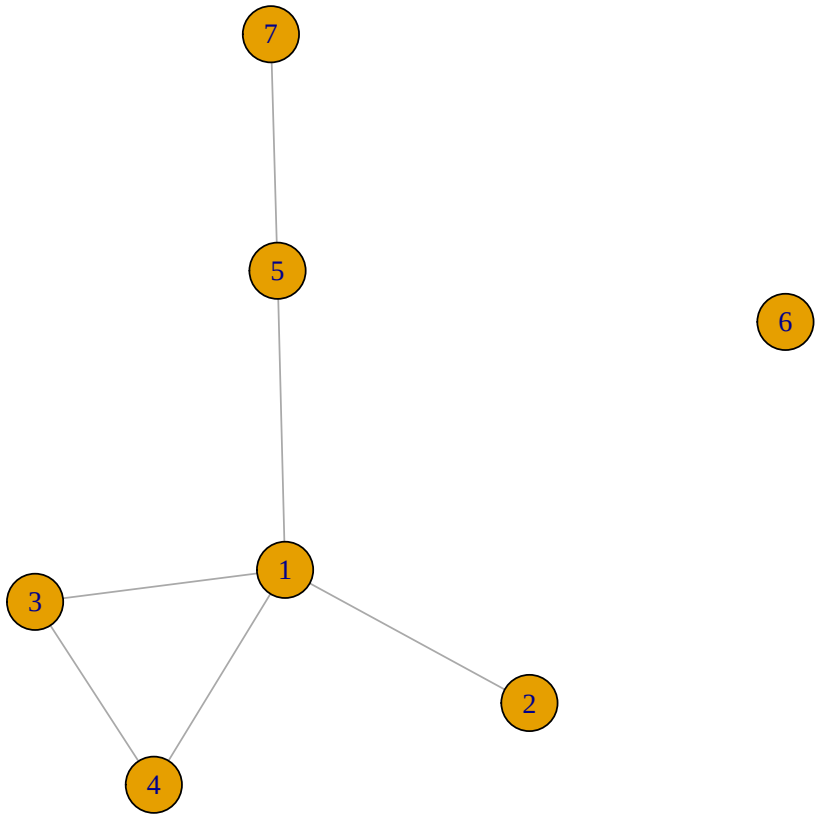


Pokolorowany duży graf pełny

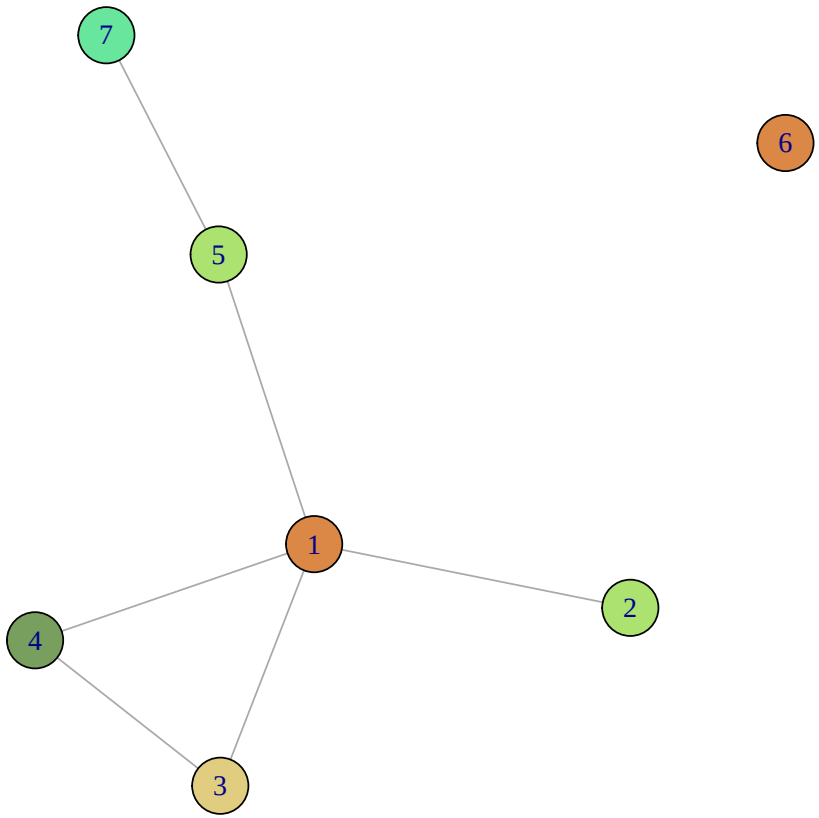
```
## [1] "Liczba różnych kolorów:"

## [1] 17
```



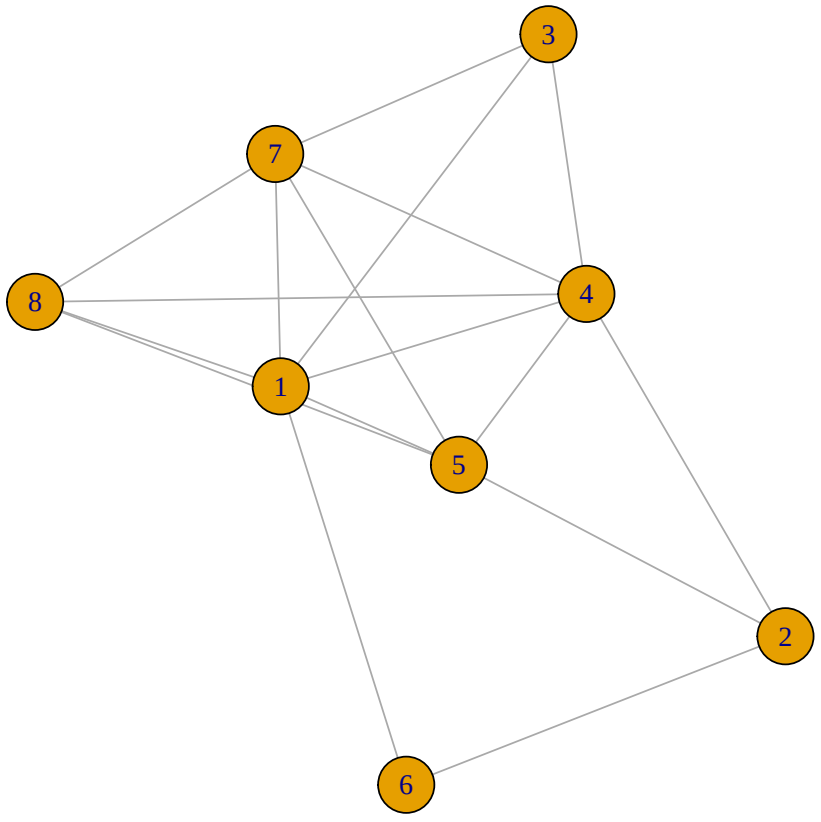
Graf z niewielką liczbą krawędzi



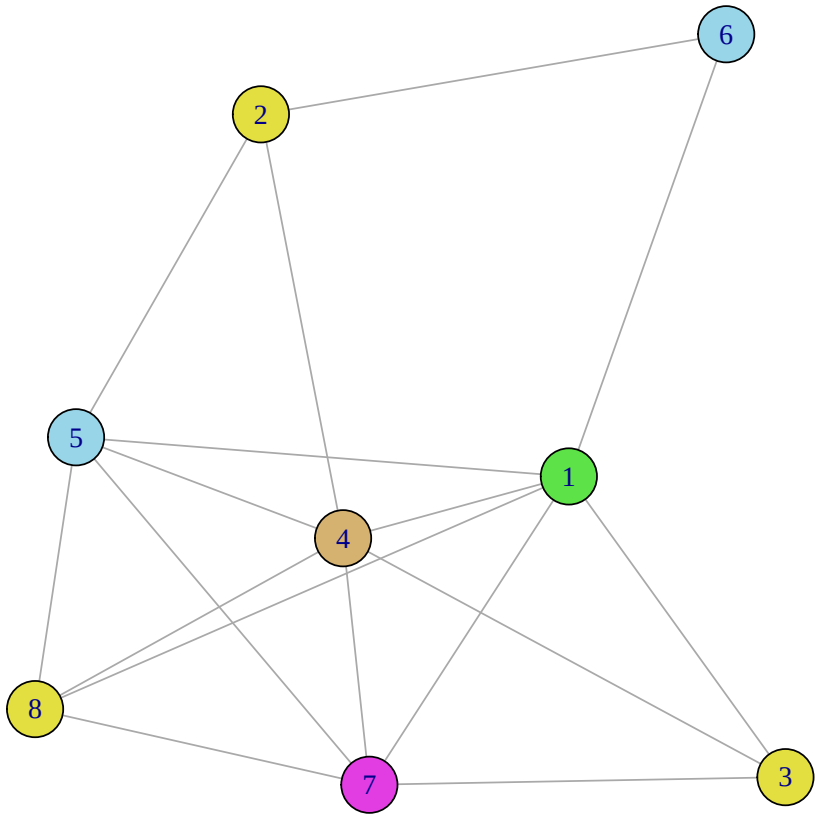
Pokolorowany graf z niewielką liczbą krawędzi



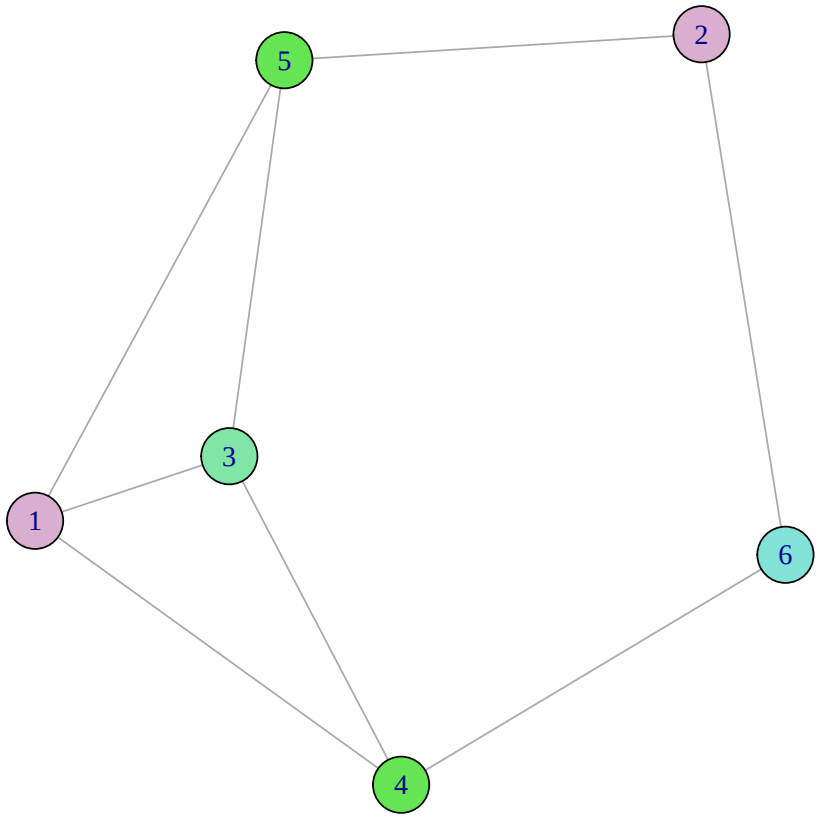
Graf z dużą liczbą krawędzi



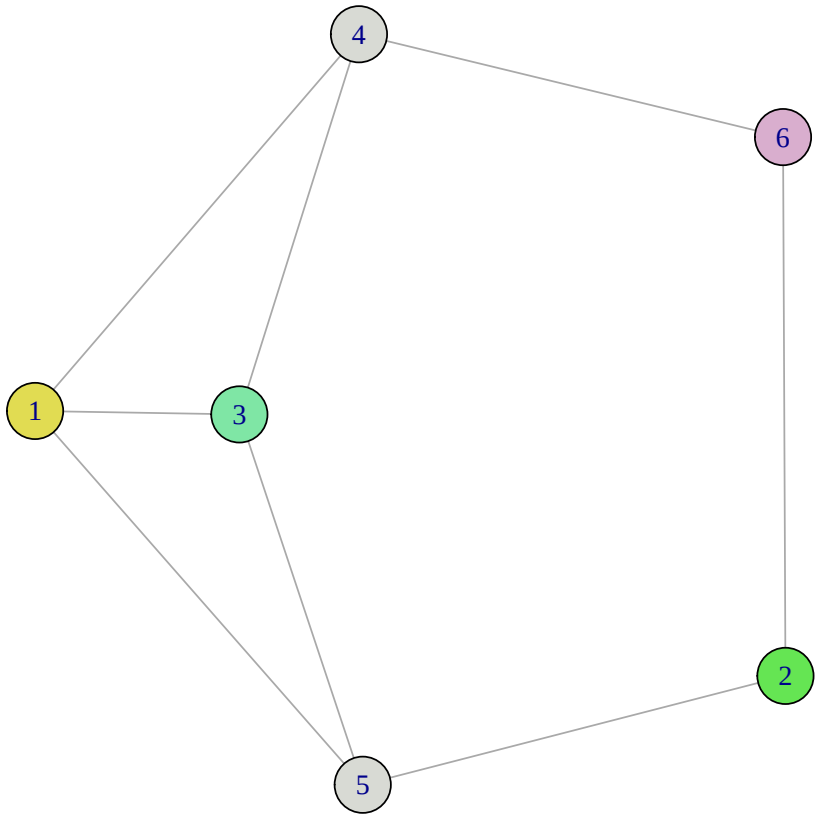
Pokolorwany graf z dużą liczbą krawędzi



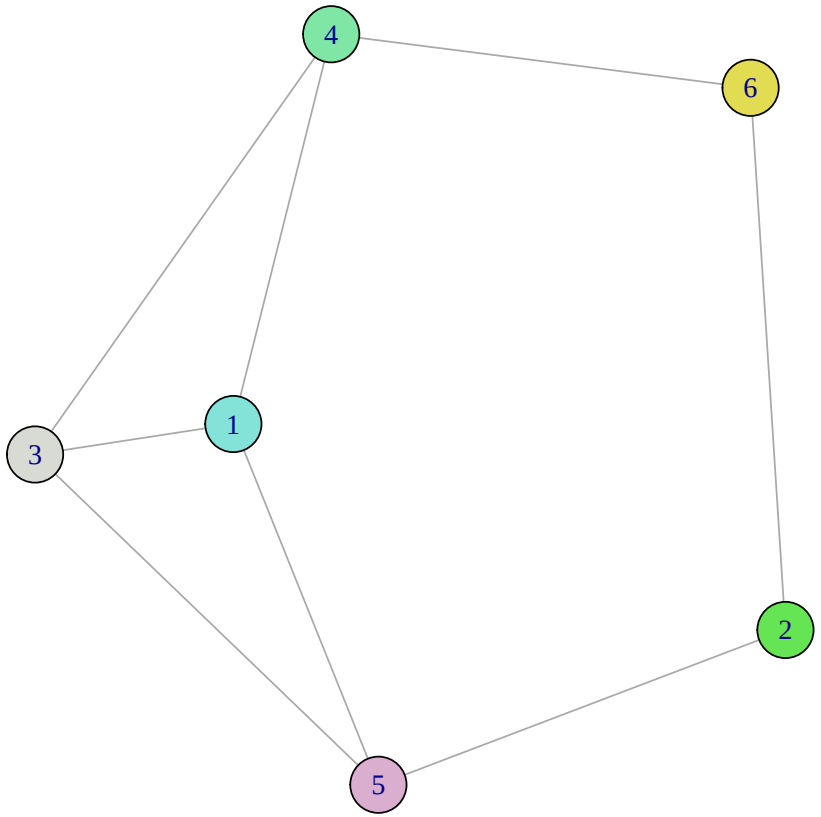
Różne kolorowania dla konkretnego grafu



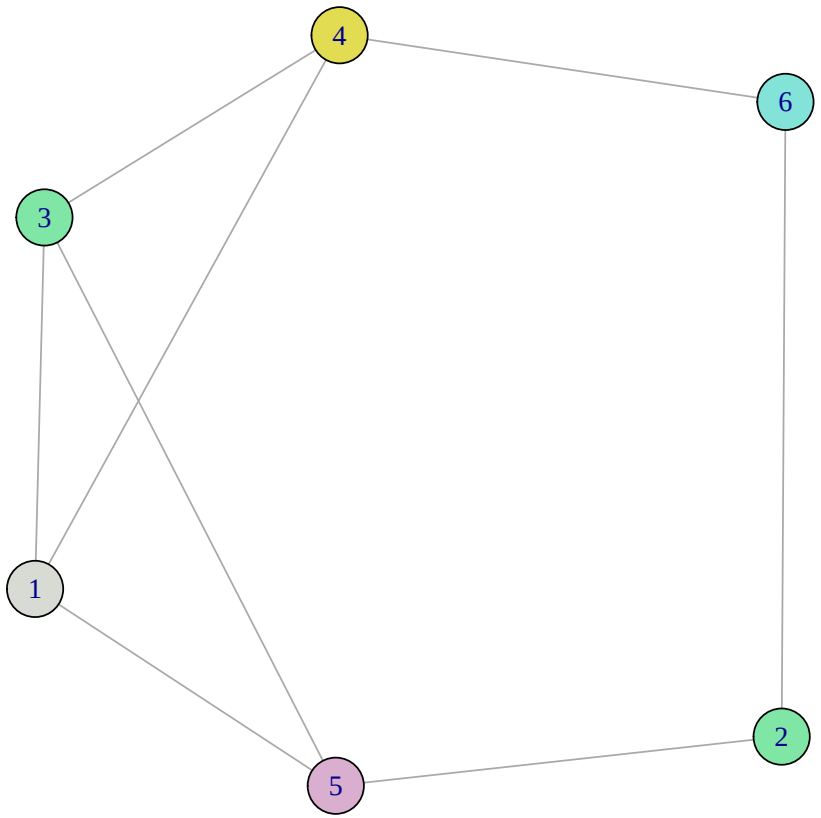
Różne kolorowania dla konkretnego grafu



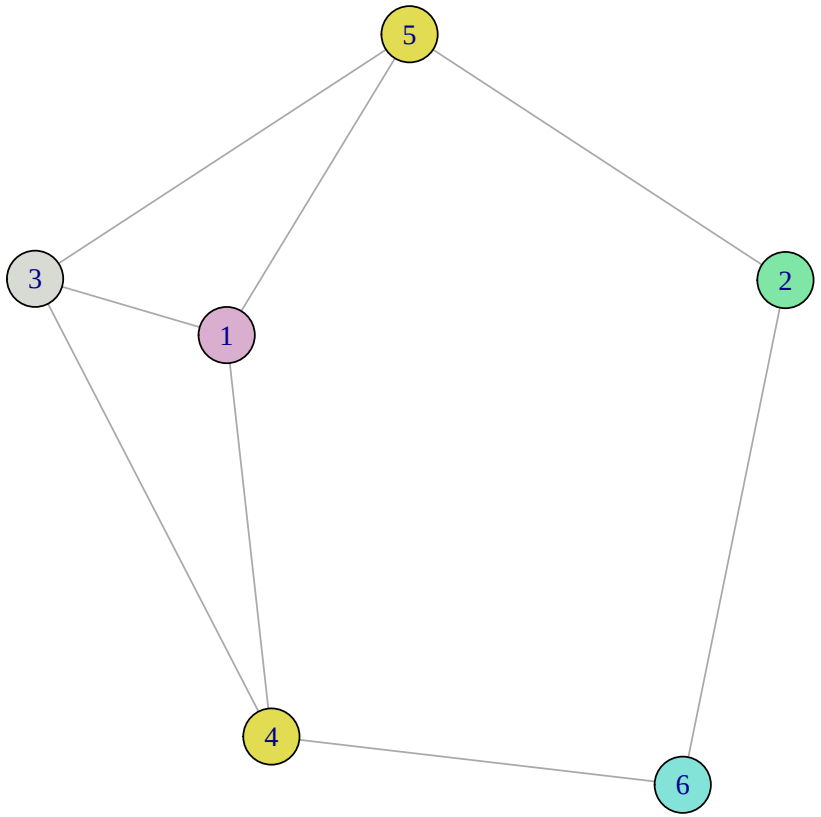
Różne kolorowania dla konkretnego grafu



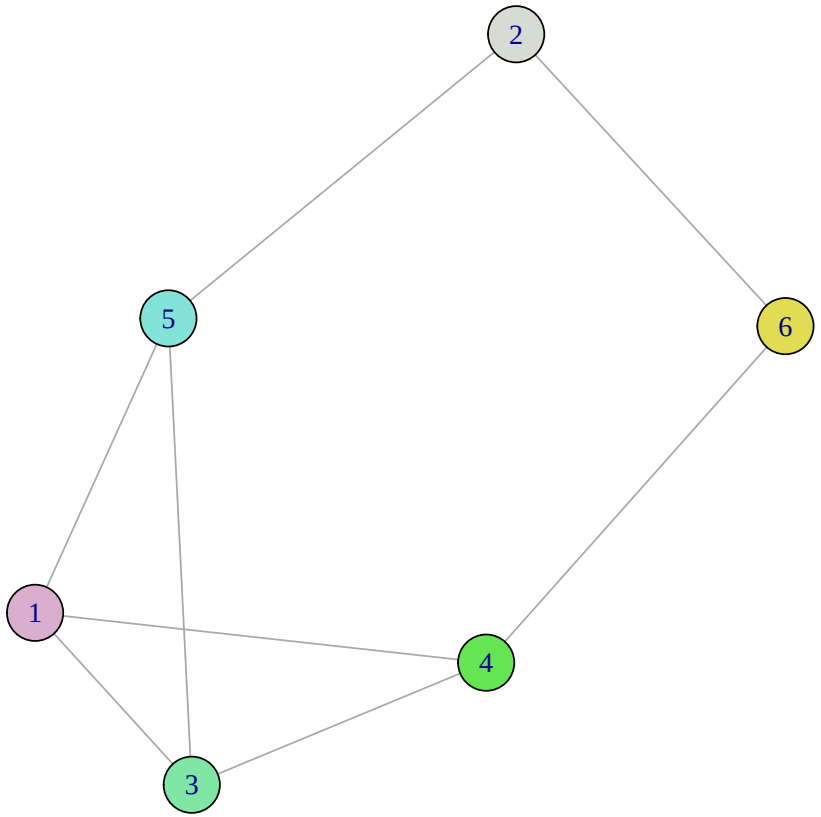
Różne kolorowania dla konkretnego grafu



Różne kolorowania dla konkretnego grafu



Różne kolorowania dla konkretnego grafu



Uzyskanie rozkładu stacjonarnego

Rozkład stacjonarny poprawnych kolorowań dla konkretnego grafu możemy uzyskać poprzez wykonanie dużą liczbę razy algorytmu i zliczenie częstości występowania każdego ze zwracanych stanów.

Podsumowanie

Powyżej zaprezentowany algorytm, bazujący na łańcuchach ograniczających, zwraca poprawne kolorowanie dla różnych typów grafów i dzięki niemu możemy uzyskać rozkład stacjonarny poprawnych kolorowań. Wyniki algorytmu empirycznie pokazują poprawność wcześniejszych rozważań teoretycznych.

Bibliografia

1. *Perfect Sampling Using Bounding Chains*, HUBER, 2004
2. *On Exact Simulation of Markov Random Fields Using Coupling from the Past*, HAGGSTROM, NELANDER, 1999

Koniec

Dziękuję za uwagę!