Algorytmy i struktury danych

Lista nr 5

Waga listy: 2

(termin oddania: 6 laboratorium - ostatnie własne zajęcia przed 7 czerwca)

Zadanie 1 (2pkt) Dla k elementowego ciągu bitowego x definiujemy wagę Hamminga $H(x)$ jako liczbę jedynek w tym ciągu (waga przyjmuje wartości od 0 do k). Niech $Z(x)$ będzie liczbą zer w ciągu x.

Dla danego k rozważmy graf skierowany (k-wymiarową hiperkostkę) o 2^k wierzchołkach, których etykietami są różne binarne ciągi długości k, a krawędziami łuki między ciągami różniącymi się na dokładnie jednej pozycji, z kierunkiem ku wierzchołkowi z większą wagą. (*Łatwo zauważyć, że każdy wierzchołek jest początkiem lub końcem dokładnie k krawędzi, a dodatkowo ciągi traktowane jako k-bitowe liczby numerują nam wierzchołki od* 0 *do* $2^k - 1$.)

Pojemność każdej krawędzi (u, v) losujemy z rozkładem jednostajnym ze zbioru $\{1, \ldots, 2^l\}$, gdzie $l = \max\{H(u), Z(u), H(v), Z(v)\}$.

Napisz program który zaimplementuje algorytm Edmonts'a-Karp'a i dla podanego parametru wejściowego $k \in \{1, \ldots, 16\}$, oznaczającego wymiar hiperkostki, wygeneruje opisany wyżej graf i obliczy maksymalny przepływ między wierzchołkami 0 i $2^k - 1$.

Program powinien przyjmować jako parametr wejściowy `--size k`.

Wynik powinien być wypisywany na standardowe wyjście, a na standardowym wyjściu błędów powinny być wypisywane w kolejności: czas działania całego programu oraz liczba ścieżek powiększających wyliczanych przez program.

Przeprowadź eksperymenty pozwalające oszacować średnią wielkość przepływu, liczbę ścieżek powiększających i czas działania programu dla wszystkich dopuszczalnych k. Wykonaj wykresy zależności tych wyników od k.

Zadanie 2 (1pkt) Dla danego k i i rozważamy dwudzielny graf losowy mający dwa rozłączne zbiory wierzchołków V_1 i V_2, każdy o mocy 2^k i krawędziach losowanych jednostajnie w ten sposób, że każdy wierzchołek z V_1 ma i sąsiadów z V_2.

Napisz program który dla podanych parametrów wejściowych $k \in \{1, \ldots, 16\}$ i $i \leqslant k$, wygeneruje opisany wyżej graf i obliczy wielkość maksymalnego skojarzenia.

Program powinien przyjmować jako parametry wejściowe `--size k` i `--degree i`.

Wynik powinien być wypisywany na standardowe wyjście, a na standardowym wyjściu błędów powinien być wypisany czas działania całego programu.

Dla każdego $k \in \{3, \ldots, 10\}$ i $i \in \{1, \ldots, k\}$ przeprowadź eksperymenty pozwalające oszacować wielkość maksymalnego skojarzenia i czas działania programu. Wykonaj dla każdego k wykresy wielkości maksymalnego skojarzenia w zależności od i. Oraz dla każdego i wykresy zależności czasu działania od k.

Zadanie 3 (1pkt) Przeczytaj dokumentację programu `glpk` i uzupełnij program z zadania 1 o generowanie pliku z modelem programowania liniowego dla problemu maksymalnego przepływu i grafu wygenerowanego w programie. Do parametrów programu dodaj opcjonalny parametr `--glpk nazwa`, który utworzy plik o podanej nazwie z modelem dla programu `glpk`.

Plik dla programu `glpk` powinien zawierać komentarze pozwalające zrozumieć zaprezentowany w nim model programowania liniowego.

Sprawdź czy `glpk` daje te same wielkości maksymalnego przepływu. Który program liczy rozwiązanie szybciej?

Zobacz notatki prof. P. Zielińskiego na temat modelowania w `glpk` *dostępne pod adresem https://cs.pwr.edu.pl/zielinski/lectures/om/glpk_notes.pdf*

Zadanie 4 (1pkt) Przeczytaj dokumentację programu `glpk` i uzupełnij program z zadania 2 o generowanie pliku z modelem programowania liniowego dla problemu maksymalnego skojarzenia i grafu wygenerowanego w programie. Do parametrów programu dodaj opcjonalny parametr `--glpk nazwa`, który utworzy plik o podanej nazwie z modelem dla programu `glpk`.

Plik dla programu `glpk` powinien zawierać komentarze pozwalające zrozumieć zaprezentowany w nim model programowania liniowego.

Sprawdź czy `glpk` daje te same wielkości maksymalnego skojarzenia. Który program liczy rozwiązanie szybciej?