Dodane bądź uzupełnione:

- dodany Schemat raportu (na dysku google).

NOWY !!!

dr inż. Tomasz Kapłon

tomasz.kaplon@pwr.edu.pl

SDiZO PRJ (wydanie 2: 21.03)

Spis treści

Komunikacja

(spis treści)

1. Zajęcia odbywają się systemie zdalnym. Komunikacja w trakcie zajęć odbywa się przez **ZOOM.**

2. Konsultacje odbywają się przez ZOOM (możemy wypróbować również MS TEAMS). Terminy konsultacji (KNS) podane poniżej. W związku z tym, że konsultacje są zdalne, proszę umawiać się mailowo na konkretną godzinę w ramach podanych terminów.

	7:30 9:00	9:15 11:00	11:15 13:00	13:15 15:00	15:15 16:55	17:05 18:45
wt	SDiZO w N		SDiZO lab P/N	SDiZO prj P/N	KNS (15 – 17) P	
śr				KNS (13 – 16)		
cz				SDiZO ćw		SDiZO ćw

3. Na dysku Google udostępniony zostanie folder, w którym znajdować się będą zadania i informacje dla obu grup projektowych. W tym ten dokument. Link dostępowy przesłany będzie przez system edukacja.cl (jsos)

4. Również na dysku Google, każdemu indywidualnie udostępniony zostanie folder, w którym będzie umieszczał raporty (pdf) oraz kolejne wersje realizowanych programów. W tym samym folderze ja będę umieszczał pliki pdf z uwagami oraz ocenami.

Opis projektu

(spis treści)

Projekt wymaga realizacji czterech zadań. Każde z zadań składa się z dwóch części - implementacji algorytmów oraz eksperymentu badawczego polegającego na potwierdzeniu bądź zaprzeczeniu pewnej tezy odnośnie oszacowania złożoności obliczeniowej implementowanych algorytmów bądź operacji na strukturach danych. W skrócie. Należy napisać program (implementacja algorytmu bądź algorytmów) wykonujący pewne operacje na określonych strukturach danych i sprawdzić, czy w uzyskane w badaniu, odczytane z wykresów, zależności czasu od wielkości instancji (dla operacji bądź algorytmu) potwierdzają (pokrywają się co do kształtu) oszacowania teoretyczne.

Zadanie 1 (spis treści)

Zadanie polega na określeniu czasowej złożoności operacji: utwórz, wstaw, dodaj, usuń i wyszukaj, w czterech strukturach danych: tablicy (struktura statyczna), liście (struktura dynamiczna), kolejce FIFO i stosie (kolejce LIFO). Należy wygenerować zbiór danych - liczb całkowitych ze znakiem (patrz: Dane testowe). Tworzyć instancję o określonym rozmiarze - do struktury wprowadzać określoną liczbę danych. Mierzyć czas utworzenia struktury. Dla niewielkich rozmiarów instancji czasy mogą być trudno mierzalne - czas wykonania bliski zeru. Wartość pomiaru (różną od zera) można uzyskać wykonując operację tworzenia (wypełniania danymi) struktury wielokrotnie (np. 1000 razy), a następnie wyliczyć czas pojedynczej operacji dzieląc czas tysiąca wykonań przez ich liczbę. Wynik - średni czas operacji, należy zapisać, wraz z rozmiarem instancji, w pliku wyjściowym (w formacie csv). Program ma działać zgodnie z opisem podanym w sekcji Działanie programów. W raporcie, w sekcji wyniki (patrz: Schemat raportu), umieszczone mają być: lista nazw plików wyjściowych otrzymanych w trakcie badań (muszą znajdować się na dysku), wykresy zależności $t(n)$ oddzielne dla każdej operacji i każdej struktury jak również, wykresy $t(n)$ z parametrem, czyli wykresy zawierające porównanie każdej operacji na każdej strukturze.

Zadanie 2

(spis treści)

Zadanie polega na implementacji i określeniu złożoności czasowej i pamięciowej dwóch algorytmów.

W **wersji E**: algorytmu sortowania z grupy $O(n^2)$ - np. bąbelkowe, koktajlowe, przez wstawianie - oraz sortowania szybkiego w wersji rekurencyjnej.

W **wersji A**: sortowania szybkiego w wersji rekurencyjnej oraz algorytmu sortowania z grupy $O(n)$ - np. przez zliczanie, kubełkowe. Należy pokazać, na podstawie wyników badań, zgodność bądź niezgodność z oszacowaniami teoretycznymi czasowych złożoności, oczekiwanych i pesymistycznych, obu algorytmów. Należy również określić złożoność pamięciową dla obu algorytmów i porównać ją z oszacowaniem teoretycznym. (proszę pamiętać, że złożoność pamięciowa może zależeć od zastosowanych struktur danych oraz od sposobu reprezentacji danych).

Zadanie 3

(spis treści)

Zadanie polega na określeniu czasowej złożoności obliczeniowej operacji tworzenia struktury oraz operacji dodawania, usuwania i wyszukiwania elementów w strukturze.

W **wersji E**, badane struktury to drzewo binarne oraz, do wyboru, co najmniej jedna ze struktur drzewa binarnego posiadających określone własności: BST, drzewo AVL, drzewo czerwono-czarne, drzewo *splay*.

W **wersji A**, należy porównać ze sobą co najmniej dwie struktury spośród: BST, drzewo AVL, drzewo czerwono-czarne, drzewo *splay*. Dla każdej ze struktur należy określić złożoności czasowe wymienionych operacji. W raporcie, sekcji Analiza wyników, należy porównać, na podstawie uzyskanych wyników (w raporcie sekcja Wyniki), trudność implementacji (np. podatność na błędy, trudność testowania) oraz efektywność czasową operacji. Należy również odpowiedzieć na pytanie o przydatność (zastosowalność) badanych struktur w praktycznych realizacjach, czyli kiedy i którą strukturę należy rekomendować w zależności od wielkości przetwarzanych instancji, częstości ich przetwarzania oraz trudności w implementacji i serwisowaniu.

Zadanie 4

(spis treści)

Zadanie polega na określeniu przydatności, praktycznego zastosowania różnych reprezentacji grafowych: macierz sąsiedztwa, macierz incydencji, lista sąsiedztwa, pęk wyjściowy, w implementacjach algorytmów wyznaczania minimalnego drzewa spinającego (MST) lub wyznaczania najkrótszej ścieżki ze wspólnego węzła do pozostałych, ze względu na złożoność pamięciową oraz czasową wykorzystujących je algorytmów. Do badania wyznaczania MST należy użyć (wykonać implementację) jednego z trzech algorytmów: *Prima*, *Kruskala*, *Boruvki*. Do badania wyznaczania najkrótszych ścieżek, algorytmu *Dijkstry* bądź *Bellmana-Forda*. (Dane testowe podane w Dane testowe do zadania 4.) Dla algorytmów wyznaczania ścieżek proszę spreparować przykład bądź przykłady z cyklem ujemnym i sprawdzić poprawność implementacji. Algorytm *Dijkstry* powinien podać błędne długości ścieżek. Algorytm *Bellmana-Forda* powinien sygnalizować istnienie takiego cyklu.

Zbadane muszą być co najmniej dwie reprezentacje: macierz sąsiedztwa i lista sąsiedztwa. Liczba zbadanych reprezentacji wpływa na ocenę za zadanie. Minimum to max 4,0 Minimum plus macierz incydencji to 5,0. Wszystkie cztery to 6,0.

Podobnie jak w przypadku poprzednich zadań należy skorzystać z grafu w Danych testowych i badać złożoność czasową wybranego algorytmu dla różnych rozmiarów instancji. W efekcie, da każdej z badanych reprezentacji musi powstać wykres $t(n)$. (Taki wykres to minimum.) Można również (co powoduje wzrost wartości zadania), zbadać wpływ gęstości grafu na złożoność czasową i pamięciową algorytmu.

Pytania, na które warto odpowiedzieć

Która z reprezentacji jest odporna na zmiany gęstości grafu? Która jest najtrudniejsza w implementacji ze względu na możliwość popełnienia błędów oraz liczbę możliwych miejsc, w których błędy można popełnić? Która z reprezentacji jest najefektywniejsza pamięciowo, pamięciowo-czasowo i czasowo (w badanym zastosowaniu)?

Czy algorytm *Dijkstry* na pewno nie potrafi poprawnie określać najkrótszych ścieżek dla grafów z wagami ujemnymi?

Dane testowe (spis treści)

do zadania nr 1

Należy wygenerować zbiór zawierający liczby całkowite ze znakiem. Zbiór ma zawierać co najmniej milion liczb.

do zadania nr 2

Należy wygenerować zbiór zawierający liczby całkowite ze znakiem. Zbiór ma zawierać co najmniej milion liczb. Jeżeli badany będzie algorytm sortowania kubełkowego, należy wygenerować zbiór z liczbami rzeczywistymi ze znakiem. Rozmiar bez zmian.

do zadania nr 3

Należy wygenerować zbiór zawierający liczby całkowite ze znakiem. Zbiór ma zawierać co najmniej milion liczb.

do zadania nr 4

Do badania algorytmów wyznaczania MST oraz *Bellmana-Forda*, należy wygenerować graf zawierający liczby całkowite ze znakiem. Do badania algorytmu *Dijkstry* wygenerować graf tylko z liczbami całkowitymi dodatnimi.

Procedura badawcza (spis treści)

Działanie programów

(spis treści)

Program uruchamiany jest w oparciu o informacje zawarte w pliku inicjującym (.INI) zawierającym parametry (częściowo różne dla każdego z zadań), sterujące działaniem programu. Parametrami wspólnymi dla wszystkich zadań są: nazwa pliku wejściowego (z danymi), liczba powtórzeń wykonania programu dla każdej instancji, liczby elementów zbioru wejściowego określające wielkość kolejnych instancji, nazwa pliku wyjściowego (z wynikami). Plik zapisywany w formacie tekstowym.

przykładowa zawartość pliku .INI

```
dane_z1_1mln.csv 5
10 100 500 1000 1000000
wyniki_z1_1mln.csv
```

Plik wynikowy (tu: `wyniki_z1_1mln.csv`) ma zawierać powyższe informacje oraz, dla każdego badanego rozmiaru instancji wszystkie zmierzone czasy wykonania.

przykładowa zawartość pliku wynikowego .csv

```
dane_z1_1mln.csv 5
10 100 500 1000 1000000
wyniki_z1_1mln.csv
10 0,45 0,46 0,39 0,44 0,45
```

Schemat raportu

(spis treści)

[dodany; znajduje się na dysku Google]

Oceny

(spis treści)

Każde zadanie oceniane będzie w skali 2,0 do 5,5. Średnia arytmetyczna ocen oraz subiektywna ocena prezentacji działania programów, odpowiedzi na pytania oraz raportów, składać się będzie na ocenę końcową.

Opóźnienie złożenia (umieszczenia na dysku) zadania w określonym terminie (patrz: Terminy złożenia) skutkować będzie obniżeniem oceny z zadania o pół (0,5) stopnia oceny uzyskanej z zadania, za każdy rozpoczęty tydzień opóźnienia. Na przykład oddanie zadania 1. ocenionego na 3,0 po 23.03 oznaczać będzie ocenę 2,0 z zadania i w konsekwencji niezaliczenie kursu.

Terminy złożenia (spis treści)

zadanie	data
1	23.03.2021
2	27.04.2021
3	18.05.2021
4	15.06.2021

Opóźnienie złożenia (umieszczenia na dysku) zadania w określonym terminie (patrz: Terminy złożenia) skutkować będzie obniżeniem oceny z zadania o pół (0,5) stopnia oceny uzyskanej z zadania, za każdy rozpoczęty tydzień opóźnienia. Na przykład oddanie zadania 1. ocenionego na 3,0 po 23.03 oznaczać będzie ocenę 2,0 z zadania i w konsekwencji niezaliczenie kursu.

Uwagi do raportów (spis treści)

W punkcie "Procedura badawcza" proszę umieszczać:

- specyfikację sprzętu, na którym przeprowadzane są badania,
- opis metody pomiaru czasu uzyskiwania rozwiązania instancji; ma pojawić się opis metody pomiaru czasu oraz odpowiedni fragment kodu programu.

Wszelkie uwagi i wątpliwości proszę zgłaszać drogą mailową bądź w trakcie spotkań