

Egzamin licencjacki/inżynierski

26 czerwca 2024

Informacja dla zdających egzamin na kierunku informatyka

Z sześciu poniższych zestawów zadań (Matematyka I, Matematyka II, Metody programowania, Matematyka dyskretna, Algorytmy i struktury danych, Metody numeryczne) należy wybrać i przedstawić na osobnych kartkach rozwiązania trzech zestawów.

Informacja dla zdających egzamin na kierunku ISIM

Z sześciu poniższych zestawów zadań (Matematyka I, Metody programowania, Matematyka dyskretna, Algorytmy i struktury danych, Metody numeryczne, Języki formalne i złożoność obliczeniowa) należy wybrać i przedstawić na osobnych kartkach rozwiązania trzech zestawów.

Informacja dla wszystkich zdających

Za brakujące (do trzech) zestawy zostanie wystawiona ocena niedostateczna z urzędu. Egzamin uważa się za zaliczony, jeśli student rozwiąże z oceną dostateczną co najmniej 2 zestawy. Wtedy ocena z egzaminu jest średnią arytmetyczną ocen z trzech wybranych zestawów. Na rozwiązanie przeznaczona jest czas $3 \times 40 + 30 = 150$ minut. Po wyjściu z sali egzaminacyjnej w czasie egzaminu nie ma możliwości powrotu do tej sali i kontynuowania pisania egzaminu.

Matematyka I — Logika dla informatyków

Rozważmy niepusty zbiór uporządkowany $\langle X, \leq \rangle$. Udowodnij, że istnieje taka rodzina zbiorów $\mathcal{X} \subseteq \mathcal{P}(X)$, że porządki $\langle X, \leq \rangle$ oraz $\langle \mathcal{X}, \subseteq \rangle$ są izomorficzne.

Matematyka II — Algebra

Zadanie 1. (4 punkty)

Wykazać, że wektory $e_1, e_1 + e_2, e_1 + e_3 + e_3$ są niezależne w przestrzeni \mathbb{R}^3 .

Zadanie 2. (4 punkty)

Jaka jest wartość wyrażenia $7/10$ w grupie (mnożymy) \mathbb{Z}_{17} ?

Zadanie 3. (5 punktów)

$$A = \begin{bmatrix} 5 & 1 & -1 \\ 1 & 3 & -3 \\ -1 & -3 & 3 \end{bmatrix}.$$

Znaleźć wartości własne tej macierzy.

Progi punktowe: 4, 6, 8, 10, 12 punktów.

Metody Programowania

Poniższe zadania należy rozwiązać używając języka Racket lub Plait.

Zadanie 1 Zaproponuj reprezentację wyrażeń logicznych ze zmiennymi o następującej gramatyce:

$$e ::= b \mid \alpha \mid e_1 \vee e_2 \mid e_1 \wedge e_2 \mid \neg e$$

gdzie b reprezentuje stałe boolowskie (\top oraz \perp), natomiast α reprezentuje zmienne. Jeżeli Twoim metajęzykiem jest Racket, to napisz predykat sprawdzający czy jego argument jest poprawnym wyrażeniem logicznym.

Zadanie 2 Napisz procedurę `eval`, która dla danego wartościowania zmiennych (od Ciebie zależy reprezentacja wartościowania), zwraca wartość wyrażeń logicznych z zadania 1.

Zadanie 3 Język wyrażeń logicznych z zadania 1 rozszerzamy o kwantyfikatory:

$$e ::= \dots \mid \forall \alpha. e \mid \exists \alpha. e$$

Formuły napisane w tym języku to kwantyfikowane formuły logiczne (QBF). Formuła postaci $\forall \alpha. e$ jest prawdą wtedy i tylko wtedy, gdy formuła e po podstawieniu za α każdej ze stałych boolowskich (\top oraz \perp) jest prawdą. Semantykę kwantyfikatora egzystencjonalnego definiuje się analogicznie.

Rozbuduj implementację procedury `eval` z zadania 2, aby obliczała wartość formuł QBF.

Matematyka dyskretna

Niech $C = \text{NWW}(1, 2, 3, \dots, n)$ i niech \mathbb{P} będzie zbiorem liczb pierwszych. Pokaż wzór

$$\log C = \sum_{p \in \mathbb{P}} \lfloor \log_p n \rfloor \log p.$$

Metody numeryczne

Za rozwiązanie zadań można otrzymać łącznie 12 punktów. Otrzymanie 4 pkt. gwarantuje ocenę dostateczną, próg dla dst+ to 5.5 pkt., dla db – 7 pkt., dla db+ 8.5 pkt., a dla bdb – 10 pkt.

1. **4 punkty** Zaproponuj **efektywny** algorytm obliczania z dużą dokładnością wartości $a^{0.75}$ ($a > 0$) wykorzystując **jedynie** operacje arytmetyczne (+, −, ·, /). Jeśli zdecydujesz się na metodę iteracyjną, pamiętaj o odpowiednim warunku **STOPu**.
2. **4 punkty** Podaj postać Newtona wielomianu interpolacyjnego dla następujących danych:

$$\text{a)} \quad \begin{array}{c|c|c|c|c} x_k & -2 & -1 & 1 & 2 \\ y_k & 2 & 0 & -4 & -30 \end{array}, \quad \text{b)} \quad \begin{array}{c|c|c|c|c|c} x_k & -2 & -1 & 0 & 1 & 2 \\ y_k & 2 & 0 & 2 & -4 & -30 \end{array}.$$

Uwaga. Rozwiązując podpunkt **b)**, **odpowiednio wykorzystaj** wynik otrzymany w podpunkcie **a)** — **tylko wtedy** możesz dostać komplet punktów.

3. **4 punkty** Załóżmy, że istnieje rozkład LU macierzy **trójkątnej** $T := [t_{ij}] \in \mathbb{R}^{n \times n}$. **Wyprowadź wzory** na elementy macierzy trójkątnej dolnej $L := [l_{ij}] \in \mathbb{R}^{n \times n}$ oraz macierzy trójkątnej górnej $U := [u_{ij}] \in \mathbb{R}^{n \times n}$, dla których zachodzi $T = L \cdot U$. Następnie, **wykorzystując** rozkład LU macierzy T , **zaproponuj i uzasadnij** algorytm rozwiązywania układu równań $T \cdot x = b$, gdzie $b \in \mathbb{R}^n$. **Podaj** jego złożoność obliczeniową i pamięciową. W której z omówionych na wykładzie z analizy numerycznej metod pojawiają się układy równań trójkątnych?

Algorytmy i struktury danych

Za rozwiązanie obydwu zadań z tej części można otrzymać w sumie do 9 punktów. Skala ocen: poniżej 3 punktów — ocena niedostateczna (egzamin niezdany), 3 punkty dają ocenę dostateczną, 4 — dostateczną z plusem, 5 — dobrą, 6 — dobrą z plusem, 7 albo więcej punktów daje ocenę bardzo dobrą.

Zadanie 1: prefiksosufiks (4 punkty)

Dane jest n -elementowe słowo $P = (p_0, p_1, \dots, p_{n-1})$ zbudowane nad ustalonym alfabetem Σ . Należy wyznaczyć dla tego słowa długość jego najdłuższego prefiksosufiksu, czyli takiego prefiksu właściwego (niebędącego całym słowem), który jest jednocześnie sufiksem tego słowa. Dla słowa P należy więc wyznaczyć maksymalną wartość j taką, że $(p_0 \dots p_{j-1}) = (p_{n-j} \dots p_{n-1})$.

Opracuj dynamiczny algorytm, który efektywnie rozwiąże to zadanie. Postaraj się wykorzystać w algorytmie wyniki obliczone dla podproblemów, czyli maksymalne prefiksosufiksy dla wszystkich prefiksów tego słowa, (rozwiązania dla podproblemów P_1, P_2, \dots, P_{n-1} , gdzie P_k oznacza k -literowy prefiks $P_k = (p_0 \dots p_{k-1})$ dla $k = 1 \dots n - 1$).

Precyzyjnie opisz ideę rozwiązania a potem zapisz ją w pseudokodzie. Uzasadnij poprawność opisaney metody i oszacuj jej złożoność obliczeniową.

Uwaga: algorytm KMP korzysta ze stabilizowanych rozwiązań tego zadania dla wszystkich prefiksów dla zadanego wzorca P .

Zadanie 2: zastosowanie zbiorów rozłącznych do wyznaczania najbliższego wolnego sektora na dysku twardym (5 punktów)

Opisz budowę drzewiastej struktury danych dla *zbiorów rozłącznych*:

- Jakie zadania realizują operacje *union* i *find* w tej strukturze? Na czym polega łączenie według rozmiaru/rangi? Na czym polega kompresja ścieżki podczas wyszukiwania?
- Napisz w pseudokodzie implementację operacji *union* z uwzględnieniem rozmiaru/rangi oraz *find* (może być wersja rekurencyjna) z kompresją ścieżki.
- Jaka jest pesymistyczna złożoność czasowa każdej z tych operacji? Jaka jest złożoność czasowa wykonania ciągu m operacji *union* i *find* na zbiorach rozłącznych z łączeniem według rozmiarów/rang i kompresją ścieżek na n -elementowym uniwersum?

Następnie napisz jak zastosować strukturę dla zbiorów rozłącznych do rozwiązania następującego problemu. Określony dysk twardy jest podzielony na n sektorów ponumerowanych od 0 do $n - 1$. Początkowo wszystkie sektory są wolne. Zajęcie wolnego sektora jest realizowane za pomocą operacji *take(s)*, gdzie argument $s \in \{0, 1, \dots, n - 1\}$ to numer zajmowanego sektora. Jeśli sektor s jest już zajęty, to należy tylko zgłosić błąd; jeśli sektor s jest wolny, to należy go oznaczyć jako zajęty i wyznaczyć numer następnego wolnego sektora; gdy za zajętyym sektorem nie ma już żadnego wolnego, to operacja *take(s)* powinna zwrócić wartość n .

Opisz ideę rozwiązania tego problemu. Zapisz w pseudokodzie operację *take* i przeanalizuj jej złożoność czasową. Jaka będzie złożoność czasowa wykonania ciągu n operacji *take*, która spowoduje zajęcie wszystkich sektorów na dysku twardym? Oczywiście zajęcie wszystkich sektorów na dysku nie musi być realizowane po kolei, raczej będzie realizowane w kolejności określonej za pomocą jakiejś n -elementowej permutacji zbioru $\{0, 1, \dots, n - 1\}$. Na przykład

dla $n = 6$ zajmujemy sektory w następującej kolejności: 1, 5, 2, 4, 0, 3; dla takich danych operacja *take* zwróci odpowiednio wartości: 2, 6, 3, 6, 3, 6.

Języki formalne i złożoność obliczeniowa

Przypomnijmy, że deterministyczny automat ze stosem (DPDA) to krotka $(\Sigma, \Gamma, Q, q_0, \delta, F)$, gdzie

- Q to skończony zbiór stanów, $q_0 \in Q$ to stan początkowy, a F to zbiór stanów akceptujących,
- Σ to alfabet wejściowy a Γ to zbiór symboli stosowych,
- $\delta: Q \times \Sigma \times (\Gamma \cup \{\perp\}) \rightarrow Q \times \Gamma^*$ to funkcja przejścia.

DPDA akceptuje słowo w wtedy i tylko wtedy, gdy stan osiągnięty po przeczytaniu w jest akceptujący (a zawartość stosu może być dowolna). Rozmiar DPDA to liczba elementów Q plus suma długości słów zwracanych przez δ , t.j., $|Q| + \sum_{(q',w)=\delta(q,a,b)} |w|$.

Podaj przykład ciągu języków regularnych L_n takich, że L_n są rozpoznawane przez DPDA wielomianowego rozmiaru względem n , ale DFA rozpoznające L_n mają rozmiar wykładniczy względem n . Uzasadnij, że Twój przykład jest poprawny.