Examen Algoritmen en Datastructuren III

Naam : .....

Stellingen uit de les mogen natuurlijk altijd gebruikt worden zonder dat het bewijs opnieuw gegeven moet worden (behalve in gevallen waar het expliciet anders staat)!

1. **Vraag bij project** (Deze vraag wordt samen met het project en de practica gequoteerd op 4 punten.)

   (a) In het project werd gevraagd om een algoritme te implementeren voor het berekenen van het product van een $n \times n$ matrix A en een vector x met n elementen op een ring van p processoren $P_0, P_1, \ldots, P_{p-1}$. Ga ervan uit dat zowel de matrix A als de vector x reeds verdeeld is over de p processoren. Geef het algoritme in pseudocode en bespreek de tijdscomplexiteit. Wat is de versnelling over de voor de hand liggende sequentiële oplossing?

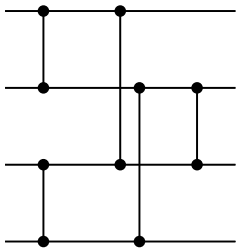
(b) Wat doet het volgende eenvoudige MPI-programma? Welke output verwacht je wanneer het programma gestart wordt met `mpirun -np 8`?

```
#include <stdio.h>
#include <mpi.h>

int main (int argc, char *argv[]) {
  MPI_Status status;
  int r,p,i,j;
  int n = 0;

  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &r);
  MPI_Comm_size(MPI_COMM_WORLD, &p);

  i = r - 1;
  j = r + 1;
  if (i != -1) {
    MPI_Recv(&n, 1, MPI_INT, i, 19, MPI_COMM_WORLD, &status);
    n = n + r;
  }
  if (j != p) {
    MPI_Send(&n, 1, MPI_INT, j, 19, MPI_COMM_WORLD);
  } else {
    printf("%d\n", n);
  }

  MPI_Finalize();
  return 0;
}
```

2. **Sorteernetwerken (2 pt.)**

   (a) Formuleer en bewijs het nul-een-principe voor sorteernetwerken.

(b) Gebruik het nul-een-principe om te bewijzen dat het onderstaande sorteernetwerk alle inputrijen van lengte 4 correct sorteert.



3. **String matching (2 pt.)**

   (a) Gegeven twee strings S en T. Gevraagd is om de string S om te zetten in de string T, waarbij volgende editeerbewerkingen toegelaten zijn:

   - het vervangen van een karakter door een ander;
   - het verwijderen van een karakter;
   - het tussenvoegen van een karakter.

   Geef een gretig algoritme voor het bepalen van een reeks editeerbewerkingen die S omzet in T.

(b) Bewijs dat dit algoritme niet gegarandeerd de reeks met het *kleinste* aantal editeerbewerkingen, nodig om S in T om te zetten, teruggeeft. Geef daartoe een tegenvoorbeeld.

4. **(2 pt)**

In het geval van extern sorteren hebben wij het bestand op een manier opgedeeld dat alle delen ongeveer even groot waren. Stel nu dat je bestanden $B_1, \ldots, B_n$ gevuld met een verschillend aantal getallen hebt die al gesorteerd zijn en die tot één groot bestand gemerged moeten worden. Daarbij bevat voor $1 \leq i \leq n$ bestand B_i precies l_i getallen. In elke stap worden 2 bestanden gemerged en de kost om een bestand met l_i getallen en een bestand met l_j getallen te mergen is $l_i + l_j$.

Wij zoeken nu de optimale volgorde om de bestanden te mergen.

a.) Geef een voorbeeld dat aantoont dat verschillende volgordes tot verschillende kosten kunnen leiden.

b.) Toon expliciet aan dat altijd $n-1$ mergeoperaties nodig zijn.

c.) Beschrijf een algoritme dat een volgorde met minimale kost bepaalt en bewijs dat jouw algoritme juist is.

5. **(1 pt)**

   Geef ofwel een voorbeeld van een tekst waar een code met vaste lengte (waar bv. elk codewoord 8 bits heeft) een kortere codering oplevert dan een Huffmancode ofwel bewijs dat een dergelijk voorbeeld niet bestaat.

6. **(3 pt)**

Jullie kennen het inpakprobleem al uit AD2: Gegeven een reeks van n gewichten $g_1, \ldots, g_n$ met $0 < g_i \leq 1$ voor $1 \leq i \leq n$. Het doel is de gewichten zo op vrachtwagens te plaatsen dat de som van de gewichten op elke vrachtwagen ten hoogste 1 is en er zo weinig mogelijk vrachtwagens gebruikt worden.

Ook dit probleem is NP-compleet en dus lijkt het een goed idee om voor grote waarden van n een metaheuristiek toe te passen.

- Geef een genetisch algoritme voor dit probleem. Geef alle definities en parameters die nodig zijn om het algoritme volledig te beschrijven. Het is niet nodig dat ze ook zo gekozen zijn dat het algoritme bijzonder goed werkt. Het algoritme moet alleen maar *in principe* werken en het optimum kunnen vinden (ook al zou dat veel te lang duren en te veel pogingen vragen).

  De bedoeling is gewoon om aan te tonen dat het principe verstaan is.

  Je kan ofwel ervoor zorgen dat crossover en mutatie alleen maar legale verdelingen van de pakketten als resultaat hebben ofwel ook ongeldige verdelingen toelaten en de voor de hand liggende doelfunctie zo wijzigen dat alleen maar legale verdelingen het optimum zijn.

7. **(2 pt)**

- Geef één voordeel van linear hashing in vergelijking met extensible hashing

- Geef één voordeel van extensible hashing in vergelijking met linear hashing

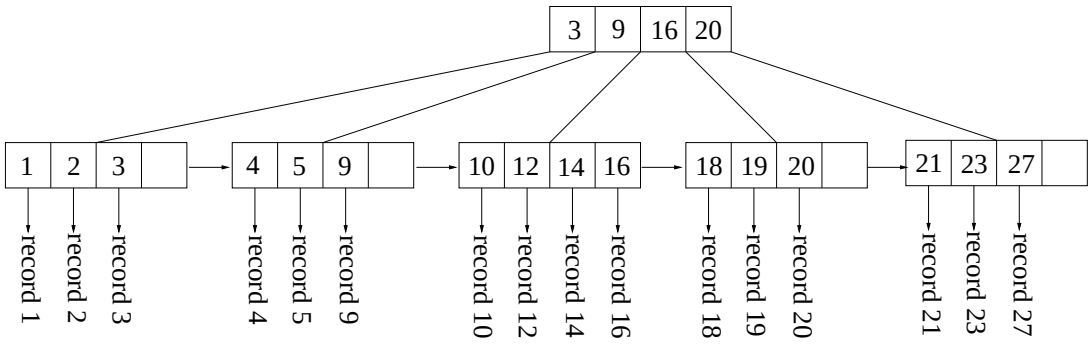
- Voeg de sleutels 7, 13, 44, 12, 1, 2 en 99 in deze volgorde toe aan een extensible hashing tabel waarbij je in het begin maar 1 bit van de hash-waarde gebruikt. Er mogen twee sleutels in één emmer zitten en wij stellen dat de hashfunctie maar 6-bit getallen schrijft (dus getallen van 0 t.e.m. 63). Het is hier voldoende de hash-waarden in de tabel te schrijven in plaats van de sleutels. Toon voldoende tussenstappen en details om te zien wat er gebeurt. De hash-waarden zijn:
  $h(7) = 53$ dus als binair getal zonder leidende nullen 110101
  $h(13) = 27$ dus binair 11011
  $h(44) = 56$ dus binair 111000
  $h(12) = 55$ dus binair 110111
  $h(1) = 16$ dus binair 10000
  $h(2) = 0$ dus binair 0
  $h(99) = 12$ dus binair 1100

8. **(3 pt)**

- In een B+-tree zitten sommige sleutels twee keer terwijl ze in een B-tree maar één keer zitten. Waarom kan een B+-tree toch een betere keuze zijn dan een B-tree?

- Stel dat x een sleutel in een interne top van een B+-tree is. Dan moet er ook een blad zijn dat x bevat. Beschrijf precies waar je dat blad kan vinden. Gebruik de definitie om te bewijzen dat jouw antwoord juist is.

- Voeg een record met sleutel 11 toe aan de volgende B+-tree met grootte 4. Toon voldoende tussenstappen om te kunnen zien wat er gebeurt.



9. **(1 pt)**

   Codeer de tekst `geen_eensteensmuur` met LZ77 waarbij je een sliding window van grootte 6 gebruikt. Toon voldoende tussenstappen om te zien wat er gebeurt. Je kan voor om het even welke manier kiezen om de posities en lengten te beschrijven maar als je een andere manier kiest dan de in de les gebruikte, moet die op voorhand precies beschreven worden!

NOG NIET OMDRAAIEN !