Fórum studentů MFF UK

Fórum pro všechny studenty matematicko-fyzikální fakulty UK, informatiky, fyziky i matematiky

Přejít na obsah

- Obsah fóra ‹ Informatika LS ‹ Výuka LS 2. ročník ‹ PRG005 Neprocedurální programování
- Změnit velikost textu
- Napsat e-mail
- Verze pro tisk

- FAQ
- Registrovat
- Přihlásit se

Zkouška 13. 6. 2016 (Dvořák, Hric)

Odeslat odpověď

Příspěvků: 2 • Stránka **1** z **1**

- Ohlásit tento příspěvek
- Odpovědět s citací

Zkouška 13. 6. 2016 (Dvořák, Hric)

od **nikdo** » 13. 6. 2016 21:18

1. část (80 minut, 4 příklady po 5 bodech, potřeba alespoň 4 body z Prologu i z Haskellu, celkem minimálně 12 bodů / 20)

- [Prolog] Napište predikát, který naplánuje pokud možno optimální (nutné použít nějakou jednoduchou heuristiku) rozvrh výroby na strojích. Na vstupu je seznam délek operací (např. [3,3,2,6,4]) a maximální čas běhu (např. 10). Operace je možné plánovat na paralelně běžící stroje, chceme, aby celkový počet potřebných strojů byl co nejmenší. Výstupem má být nějaké optimální rozložení operací pro jednotlivé stroje (např. [[3,3,2],[6,4]], což znamená, že použijeme dva stroje, první z nich vykoná operace trvající 3, 3 a 2 úseky, druhý operace trvající 6 a 4 časové úseky, obojí se vejde do limitu 10 časových úseků / stroj).

  Náznak možného řešení: Napište si predikát na naplánování co nejvíc operací pro jeden stroj. Rekurzivně plánujte stroje tak dlouho, dokud máte neprázdný seznam úloh. Jako heuristiku šlo použít třeba setřídění.

- [Prolog] Máte zadaný binární strom (klasická *t(vlevo, hodnota, vpravo)* notace). Roztřiďte vrcholy podle toho, kolikrát musíme jít doprava, než je objevíme. Například pro tento strom je výstupem [[A,B,F],[C,D,G],[E,H,I]].

  Náznak možného řešení: Je třeba prostě strom projít a rekurzivně si předávat počet kroků doprava a přitom stavět seznam klasicky appendama.

- [Haskell] Jsou zadány dva typy stromů - jeden, který nemá explicitně rozlišenou levo-pravou orientaci, a druhý, u něhož jsou synové rozlišeni na levé a pravé. Na vstupu dostanete strom, který má v každém vrcholu hodnotu a seznam synů. V každém vrcholu je dvojice (a,Int), kde číslo Int určuje, kolik synů má být vlevo. Úkolem je napsat funkci, která z takového stromu vyrobí strom, který má v každém vrcholu explicitně dva seznamy pro levé a pravé syny.

  Kód: Vybrat vše
  ```
  data Tree1 a = Node1 (a,Int) [Tree1 a]
  data Tree2 a = Node2 [Tree2 a] a [Tree2 a]
  ```

  Náznak možného řešení: Průchod stromem, na generování nového listu použít rekurzi na syny a na levo/pravo použít take, drop.

- [Haskell] Máte nějakou funkci, která nabývá jen dvou různých funkčních hodnot. Funkce přechází někde (nevíme kde) skokově z jedné funkční hodnoty na druhou. Na vstupu dostanete *c* a *d* určující ony dvě funkční hodnoty. Dále dostanete seznam (x,y) bodů, ve kterých jste funkci změřili s nějakou chybou. Napište funkci, která na výstupu tyto body rozdělí na levé a pravé (seznam dvou seznamů) podle toho, které body patří ještě k hodnotě *c*, a které už k hodnotě *d*. Pozor, je potřeba minimalizovat celkovou odchylku spočtenou jako součet $(f(x_i) - y_i)^2$ přes všechny body, kde f(x) je změřená hodnota (ze seznamu) a y skutečná hodnota z našeho odhadu.

  Náznak možného řešení: Minimalizaci odchylky šlo řešit prostě průchodem všech možností a vybráním té minimální. Při počítání odchylek zkoušíte posouvat hranici a přitom třídit body.

2. část (90 minut, Haskell/Prolog libovolně)
Ohodnocování vrcholů a hran s minimalizací odchylky, viz http://forum.matfyz.info/viewtopic.php?f=169&t=10536.

nikdo

Nahoru

- Ohlásit tento příspěvek
- Odpovědět s citací

Re: Zkouška 13. 6. 2016 (Dvořák, Hric)

od **jankasvk** » 17. 9. 2016 15:30

Pridávam svoje riešenia k malým úlohám.
Môže sa vyskytnúť neporozumenie úlohy, či nesprávne riešenie. Snáď ale niekomu môžu pomôcť keď si

nebude vedieť dať rady.

Kód: Vybrat vše

```
planuj(Max, Oper, Rozvrh) :-
    sort(Oper, ZorOper),
    reverse(ZorOper, RZ),
    rozdel2(Max, RZ, Rozvrh).

rozdel2(_, [], []) :- !.
rozdel2(Max, Zs, [X|Xs]) :-
    %prideluj ulohy kym stroj vladze
    rozdel(Max, Zs, [], X, Zvysok),
    %pridel zvysne ulohy
    rozdel2(Max, Zvysok, Xs).


% kolko este mozem vtesnat --- odkroji z listu tolko co sa zmesti do max
rozdel(_, [], Vys, Vys, []).
rozdel(Left, [X|Xs], Vys, Vys, [X|Xs]) :-
    X > Left.
rozdel(Left, [X|Xs], Ys, Vys, Zv) :-
    X =< Left,
    NewL is Left - X,
    rozdel(NewL, Xs, [X|Ys], Vys, Zv).
```

Kód: Vybrat vše

```
vpravo(nil, _, []).
vpravo(t(L, A, P), Hlbka, [Hlbka-A|NewList]) :-
    succ(Hlbka, NewH),
    vpravo(P, NewH, L1),
    vpravo(L, Hlbka, L2),
    append(L1, L2, NewList).

vyries(Strom, Vysledok) :-
    vpravo(Strom, 0, Vys),
    sort(Vys, Vys2),
    zg(Vys2, Vysledok).

% urcenie spravneho poctu mnoziniek a ich vytvorenie
zg(Z, New) :-
    last(Z, N-X),
    zg2(Z, 0, [], New),
    succ(N, N1),
    length(New, N1).

% spajanie vrcholov do mnozin
zg2([], N, Old, New) :-
    nth0(N, New, Old).
zg2([N-X|Xs], N, Old, New) :-
    zg2(Xs, N, [X|Old], New).
zg2([N-X|Xs], M, Old, New) :-
```

```
        N \= M,
        nth0(M, New, Old),
        zg2(Xs, N, [X], New).
```

Kód: Vybrat vše

```
data Tree1 a = Node1 (a, Int) [Tree1 a]
data Tree2 a = Node2 [Tree2 a] a [Tree2 a]
             deriving Show

vyvaz :: Tree1 a -> Tree2 a
vyvaz (Node1 (hod, 0) []) = Node2 [] hod []
vyvaz (Node1 (hod, poc) xs) =
    Node2 (map (vyvaz) (take poc xs)) hod (map (vyvaz) (drop poc xs))
```

Kód: Vybrat vše

```
vyries :: Float -> Float -> [(Float, Float)] ->  ([Float], [Float])
vyries c d merania = rozdel merania (skok c d merania)

skok :: Float -> Float -> [(Float, Float)] -> Float
skok c d merania = snd $ minimum [ (odchylka merania x c d, x) | (x, y) <- merania]

rozdel :: [(Float, Float)] -> Float -> ([Float], [Float])
rozdel merania bod = (map fst $ takeWhile (\(x, y) -> x < bod) merania, map fst $ dropWhile
(\(x,y) -> x < bod) merania)

odchylka :: [(Float, Float)] -> Float -> Float -> Float -> Float
odchylka [] _ _ _ = 0
odchylka ((x,y):xs) bod c d
    | x < bod = (y - c)^2 + zvysnaOdchylka
    | otherwise = (y - d)^2 + zvysnaOdchylka
    where zvysnaOdchylka = odchylka xs bod c d
```

jankasvk

Matfyz(ák|ačka) level I

Příspěvky: 13
Registrován: 31. 5. 2015 12:05
Typ studia: Informatika Bc.

Nahoru

Zobrazit příspěvky za předchozí: Všechny příspěvky Seřadit podle Čas odeslání Vzestupně Přejít

Odeslat odpověď

Příspěvků: 2 • Stránka **1** z **1**

Zpět na PRG005 Neprocedurální programování

Přejít na: PRG005 Neprocedurální programování Přejít

Kdo je online

Uživatelé procházející toto fórum: Žádní registrovaní uživatelé a 1 návštěvník

- Obsah fóra
- Tým • Smazat všechny cookies z fóra • Všechny časy jsou v UTC + 1 hodina