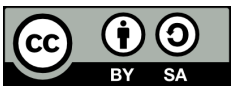
Rešene naloge pri predmetu
OPERACIJSKE RAZISKAVE

Študijsko gradivo za študente
2. letnika Finančne matematike

Janoš Vidali

Ljubljana, 2021

NASLOV: Rešene naloge pri predmetu OPERACIJSKE RAZISKAVE
PODNASLOV: Študijsko gradivo za študente 2. letnika Finančne matematike
AVTOR: Janoš Vidali
IZDAJA: 1. izdaja
ZALOŽNIK: samozaložba
KRAJ: Ljubljana
LETO IZIDA: 2021
AVTORSKE PRAVICE: Janoš Vidali
CENA: publikacija je brezplačna
NATIS: elektronsko gradivo, dostopno na naslovu
https://jaanos.github.io/or-zbirka/pdf/or-gradivo.pdf



Kataložni zapis o publikaciji (CIP) pripravili v Narodni in univerzitetni knjižnici v Ljubljani

COBISS.SI-ID 53559811

ISBN 978-961-07-0416-4 (pdf)

Kazalo

Predgovor

Na pobudo izr. prof. dr. Arjane Žitnik, ki od študijskega leta 2017/18 predava predmet Operacijske raziskave za študente 2. letnika Finančne matematike, sem kot asistent pri tem predmetu začel zbirati naloge iz vaj, kolokvijev in izpitov ter njihove rešitve. Pričujočo zbirko sestavljajo predvsem naloge, ki sem jih sam sestavil (ali dopolnil) od študijskega leta 2016/17 naprej, ko sem prevzel vodenje vaj pri tem predmetu. V pripravi je tudi obsežnejša zbirka, ki bo zajemala tudi naloge, ki so se pojavile na vajah, kolokvijih in izpitih v prejšnjih letih.

Na tem mestu bi se rad zahvalil prof. Arjani Žitnik za spodbudo, mojemu predhodniku na asistentskem mestu Davidu Gajserju, ki je sestavil nalogi 6.1 in 6.3, ter študentom, ki so prispevali nekatere rešitve, in sicer Evi Deželak (rešitve nalog 3.7, 5.1, 5.2, 5.3 in 5.10), Nejcu Ševerkarju (rešitvi nalog 5.5 in 5.9), Anji Trobec (rešitev naloge 6.7) in Janu Šifrerju (popravek rešitve naloge 5.5). Omenim naj še, da sta nalogi 4.1 in 5.3 vzeti iz vira [1], ki je služil tudi kot navdih za nekatere naloge iz razdelka 1.5.

1 Naloge

1.1 Zahtevnost algoritmov

Naloga 1.1. Vir: Vaje OR 21.2.2018

Naj bo $A[1\ldots n][1\ldots n]$ matrika (tj., seznam seznamov) dimenzij $n \times n$. Dan je spodnji program:

```
for i = 1,...,n do
    for j = i+1,...,n do
        A[i][j] ← A[j][i]
    end for
end for
```

(a) Kaj počne zgornji program?

(b) Oceni število korakov, ki jih opravi zgornji program, v odvisnosti od parametra n.

Naloga 1.2. Vir: Vaje OR 21.2.2018

Naj bo $\ell[1\ldots n]$ seznam, ki ima na začetku vse vrednosti nastavljene na 0. Dan je spodnji program:

```
i ← 1
while i ≤ n do
    ℓ[i] ← 1 − ℓ[i]
    if ℓ[i] = 1 then
        i ← 1
    else
        i ← i + 1
    end if
end while
```

(a) Kaj se dogaja, ko teče zgornji program?

(b) Oceni število korakov, ki jih opravi zgornji program, v odvisnosti od parametra n.

Naloga 1.3. Vir: Vaje OR 21.2.2018

Algoritem BUBBLESORT uredi vhodni seznam $\ell[1\ldots n]$ tako, da zamenjuje sosednje elemente v nepravem vrstnem redu:

```
function BUBBLESORT(ℓ[1...n])
    z ← n
    while z > 1 do
        y ← 1
        for i = 2,...,z do
            if ℓ[i−1] > ℓ[i] then
```

```
            ℓ[i−1], ℓ[i] ← ℓ[i], ℓ[i−1]
            y ← i
        end if
    end for
    z ← y − 1
  end while
end function
```

(a) Izvedi algoritem na seznamu $[7, 11, 16, 7, 5]$.

(b) Določi časovno zahtevnost algoritma.

Naloga 1.4. Vir: Vaje OR 12.10.2016

Algoritem MERGESORT uredi vhodni seznam tako, da ga najprej razdeli na dva dela, nato vsakega rekurzivno uredi, nazadnje pa zlije dobljena urejena seznama.

(a) S psevdokodo zapiši algoritem MERGESORT.

(b) Izvedi algoritem na seznamu $[7, 11, 16, 7, 5, 0, 14, 1, 19, 13]$.

(c) Določi časovno zahtevnost algoritma.

Naloga 1.5. Vir: Vaje OR 21.2.2018

Število n želimo razcepiti na dva netrivialna celoštevilska faktorja, kar storimo s sledečim algoritmom:

```
function RAZCEP(n)
  for i = 2, ..., ⌊√n⌋ do
    if n/i je celo število then
      return (i, n/i)
    end if
  end for
  return n je praštevilo
end function
```

Določi časovno zahtevnost algoritma. Ali je ta algoritem polinomski?

Naloga 1.6. Vir: Vaje OR 21.2.2018

Zapiši rekurziven algoritem, ki na vhod dobi celo število n in teče v času $O(\sqrt{n})$. Uporaba korenjenja ni dovoljena.

1.2 Celoštevilsko linearno programiranje

Naloga 2.1. Vir: Vaje OR 12.10.2016

Napiši linearni program, ki modelira iskanje največjega prirejanja v dvodelnem grafu.

Naloga 2.2. Vir: Izpit OR 15.12.2016

Na oddelku za matematiko je zaposlenih n asistentov, ki jim moramo dodeliti vaje pri m predmetih. Za asistenta i ($1 \leq i \leq n$) naj bosta a_i in b_i najmanjše in največje število ur, ki jih lahko izvaja, ter $N_i \subseteq \{1, 2, \ldots, m\}$ množica predmetov, ki jih ne želi izvajati. Za predmet j ($1 \leq j \leq m$) naj bo c_j število skupin za vaje pri predmetu, ter u_j število ur vaj na skupino. Poleg tega vemo, da sta asistenta p in q skregana, zato pri nobenem predmetu ne smeta oba izvajati vaj.

Predmete želimo asistentom dodeliti tako, da bomo ob upoštevanju njihovih želja minimizirali največje število različnih predmetov, ki smo jih dodelili posameznemu asistentu.

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Namig: napiši program s spremenljivko t, ki je dopusten natanko tedaj, ko vsak asistent dobi največ t različnih predmetov, in potem minimiziraj t.

Naloga 2.3. Vir: Izpit OR 31.1.2017

Imamo m opravil, ki jih želimo razporediti med n strojev. Vsak stroj lahko hkrati opravlja le eno opravilo, vsa opravila pa trajajo eno časovno enoto, neodvisno od stroja. Če stroj i ($1 \leq i \leq n$) uporabimo za vsaj eno opravilo, plačamo ceno c_i (cena ostane enaka, če na istem stroju naredimo več opravil). Skupni stroški ne smejo preseči količine C. Dani sta še množici parov P in S, pri čemer $(j,k) \in P$ pomeni, da mora biti opravilo j dokončano pred začetkom opravila k, $(j,k) \in S$ pa pomeni, da se opravili j in k ne smeta izvajati hkrati. Imamo še dodaten pogoj, ki zahteva, da je lahko v posamezni časovni enoti lahko aktivnih največ A strojev. Minimizirati želimo čas dokončanja zadnjega stroja.

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Naloga 2.4. Vir: Izpit OR 10.7.2017

Za hitrejše nalaganje posnetkov na YouTubu se uporabljajo krajevni strežniki, do katerih lahko uporabniki na določeni lokaciji hitreje dostopajo kakor do glavnega strežnika, ki vsebuje vse posnetke. Vendar pa imajo krajevni strežniki omejen prostor, zato je potrebno ugotoviti, kateri posnetki naj se naložijo na katere krajevne strežnike.

Denimo, da imamo poleg glavnega strežnika še k krajevnih strežnikov, m internetnih ponudnikov in n posnetkov. Naj bo c_h ($1 \leq h \leq k$) prostor v megabajtih, ki je na voljo na krajevnem strežniku h. Z indeksom 0 označimo glavni strežnik – lahko torej predpostavljaš $c_0 = \infty$. Nadalje naj bo s_j ($1 \leq j \leq n$) velikost posnetka

j, prav tako v megabajtih, ℓ_{hi} ($0 \le h \le k$, $1 \le i \le m$) latenca (zakasnitev pri prenosu v milisekundah) ponudnika i do strežnika h, in r_{ij} ($1 \le i \le m$, $1 \le j \le n$) število zahtevkov za posnetek j, ki jih pričakujejo od ponudnika i. Vsi parametri so cela števila.

Pri vsakem zahtevku bo posnetek poslan iz strežnika z najmanjšo latenco, ki vsebuje želeni posnetek. Lahko predpostavljaš, da ima za vsakega ponudnika glavni strežnik največjo latenco (torej $\ell_{0i} \ge \ell_{hi}$ za vsaka $1 \le h \le k$, $1 \le i \le m$). Določiti želimo, katere posnetke naj naložimo na posamezen krajevni strežnik, da minimiziramo vsoto pričakovanih latenc pri vseh zahtevkih. Posamezen posnetek lahko seveda naložimo tudi na več krajevnih strežnikov, ali pa na nobenega (v tem primeru bo poslan iz glavnega strežnika).

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Naloga 2.5. Vir: Izpit OR 29.8.2017

Distributer ima A zabojev avokadov in B zabojev banan, ki jih bo prodal n trgovcem. Trgovec i ($1 \le i \le n$) plača a_i evrov za zaboj avokadov in b_i evrov za zaboj banan, skupaj pa bo porabil največ c_i evrov. Distributer bo zaboje dostavil s tovornjaki, v katerih je lahko največ K zabojev (ne glede na vsebino). Če nekemu trgovcu dostavi t zabojev, bo torej opravljenih $\lceil t/K \rceil$ voženj. Vsaka vožnja do trgovca i (ne glede na to, koliko je poln tovornjak) ga stane d_i evrov. Poleg tega bo trgovec p kupil samo banane ali samo avokade, trgovec q pa bo kupil vsaj en zaboj avokadov, če bo tudi trgovec r kupil vsaj en zaboj avokadov. Distributer želi zaboje razdeliti med trgovce tako, da bo maksimiziral svoj dobiček – torej vsoto cen, ki jih plačajo trgovci, zmanjšano za stroške dostave. Lahko predpostavljaš, da so vse cene pozitivne.

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Naloga 2.6. Vir: Kolokvij OR 23.4.2018

Pri izvedbi projekta bo potrebno narediti n nalog, ki jih bomo dodelili delavcem. Vsako nalogo bo opravil natanko en delavec, vsak delavec pa lahko hkrati izvaja samo eno nalogo. Naj bo $t_i \in \mathbb{N}$ število časovnih enot, ki ga posamezen delavec potrebuje za dokončanje naloge i ($1 \le i \le n$). Vsaka naloga mora biti opravljena v enem kosu (če torej začnemo z nalogo i v času s, bo ta dokončana v času $s + t_i$, brez možnosti prekinitve in kasnejšega dokončanja). Celoten projekt mora biti dokončan v času T. Dana je še množica parov S, kjer $(i, j) \in S$ pomeni, da se naloga j ne sme začeti, preden se zaključi naloga i (lahko se pa j začne izvajati v trenutku, ko se i konča).

Delavce bomo najeli preko podjetja za posredovanje dela, to pa nam zaračuna fiksno ceno na najetega delavca (za ustrezna plačila delavcem bodo tako poskrbeli oni). Minimizirati želimo torej število najetih delavcev. Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Naloga 2.7. Vir: Izpit OR 11.6.2018

Avtobusno podjetje želi uvesti novo avtobusno linijo. Dan je neusmerjen enostaven graf $G = (V, E)$, katerega vozlišča predstavljajo postajališča, povezave pa ceste med njimi. Za vsako povezavo $uv \in E$ poznamo še čas $c_{uv} \in \mathbb{N}$, v katerem avtobus pride od u do v.

Dan je še seznam postajališč $p_1, p_2, \ldots p_n \in V$, ki jih linija mora obiskati v tem vrstnem redu – začeti mora v p_1 in končati v p_n, na poti med dvema postajališčema iz seznama pa lahko obišče tudi druga postajališča. Linija lahko vsako postajališče obišče največ enkrat (ko doseže končno postajališče, se vrne po isti poti do začetnega). Skupno trajanje vožnje (v eno smer) je lahko največ T. Podjetje želi določiti linijo tako, da bo obiskala čim več postaj.

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Namig: poskrbi, da linija nima ciklov.

Naloga 2.8. Vir: Izpit OR 5.7.2018

Nogometni klub pred novo sezono prenavlja svoj igralski kader. Naj bo A množica trenutnih igralcev kluba, B pa množica igralcev, za katere se v klubu zanimajo (A in B sta disjunktni množici). Za vsakega igralca $i \in A$ imajo s strani kluba j ($1 \leq j \leq n$) ponudbo v višini $p_{ij} \in \mathbb{N}$ (če se klub j ne zanima za igralca i, lahko predpostaviš $p_{ij} = 0$), za vsakega igralca $i \in B$ pa bodo morali plačati odkupnino $r_i \in \mathbb{N}$, če ga hočejo pripeljati v klub. Vsakega igralca $i \in A$ lahko seveda prodamo kvečjemu enemu klubu. Skupni stroški trgovanja (tj., razlika stroškov nakupov in dobička od odprodaj) ne smejo preseči S, število igralcev v klubu pa mora ostati enako kot pred trgovanjem.

Za vsakega igralca $i \in A \cup B$ poznamo njegov količnik kvalitete q_i, vsak pa pripada natanko eni izmed množic G (vratarji), D (branilci), M (vezni igralci) in F (napadalci). Število igralcev kluba v vsaki izmed teh množic se lahko spremeni (poveča ali zmanjša) za največ 1. Poleg tega lahko igralca $a, b \in A$ prodamo le istemu klubu – ali pa oba igralca ostaneta v klubu. Doseči želimo, da bo kvaliteta kluba čim večja – maksimizirati želimo torej vsoto količnikov q_i za igralce v klubu.

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Naloga 2.9. Vir: Kolokvij OR 19.4.2019

Na turnirju poleg tebe sodeluje še n igralcev, pri čemer se bo vsak igralec soočil z vsakim drugim igralcem v enem dvoboju. Vsak igralec na začetku dobi M žetonov, za katere se (po začetni fazi zbiranja informacij) vnaprej odloči, kako jih bo razdelil med dvoboje. V dvoboju zmaga igralec z večjim številom vloženih žetonov. Če oba igralca vložita enako število žetonov, je izid dvoboja izenačen. Zmagovalec dvoboja dobi 3 točke, poraženec 0 točk, v primeru izenačenega izida pa vsak dvobojevalec dobi 1 točko. Cilj vsakega igralca je zbrati čim večje število točk.

(a) Denimo, da za vsakega igralca izveš, kako namerava igrati. Naj bo torej c_i število žetonov, ki jih bo i-ti nasprotnik ($1 \leq i \leq n$) vložil v dvoboj proti tebi. Svojo strategijo želiš načrtovati tako, da bi dosegel čim večje število točk. Zapiši celoštevilski linearni program, ki modelira ta problem.

(b) Celoštevilskemu linearnemu programu iz prejšnje točke dodaj omejitve, ki modelirajo sledeče pogoje:

- Proti nobenemu od nasprotnikov iz množice A ne smeš doseči izenačenega izida.
- Izgubiš lahko kvečjemu proti dvema nasprotnikoma iz množice B.
- Če izgubiš proti nasprotniku u, moraš proti nasprotnikoma v in w doseči vsaj 4 točke.
- Več kot t žetonov lahko vložiš v največ k dvobojev.
- Izgubiti moraš vsaj ℓ dvobojev.

Naloga 2.10. Vir: Izpit OR 3.6.2019

V trgovini z oblačili sestavljajo ponudbo za poletno sezono. Odločijo se lahko za nakup kolekcij n različnih proizvajalcev, pri čemer za posamezen izvod kolekcije i-tega proizvajalca ($1 \leq i \leq n$) plačajo ceno c_i evrov, od nje pa si obetajo zaslužek d_i evrov. Kolekcijo i-tega proizvajalca lahko kupijo v največ k_i izvodih (vsak izvod je nedeljiva enota), z njeno prisotnostjo v ponudbi (ne glede na količino) pa si obetajo povečano vidnost in s tem dodaten zaslužek e_i evrov (npr. iz prodaje stalne ponudbe in starih zalog).

Za nakup kolekcij imajo na voljo C evrov, iz marketinških razlogov pa želijo kupiti kolekcije največ K različnih proizvajalcev. Poleg tega proizvajalec p zahteva, da če kupijo kak izvod njegove kolekcije, je morajo kupiti v vsaj toliko izvodih kot kolekciji proizvajalcev q in r skupaj (če pa ne kupijo nobenega izvoda kolekcije proizvajalca p, te omejitve ni). V trgovini želijo maksimizirati pričakovani dobiček (tj., razliko zaslužka od prodaje s stroški nakupa).

Zapiši celoštevilski linearni program, ki modelira zgoraj opisani problem.

Naloga 2.11. Vir: Izpit OR 27.8.2019

Glasbeni festival bo trajal m dni, vsak dan pa bo nastopilo k glasbenih skupin. Organizator izbira nastopajoče izmed n glasbenih skupin (kjer je $n > km$). Na voljo ima M evrov za plačilo glasbenikom, pri čemer i-ta skupina ($1 \leq i \leq n$) računa m_i evrov za nastop, pripeljala pa bo a_{ij} obiskovalcev, če bo nastopila j-ta po vrsti v dnevu ($1 \leq j \leq k$). Vsaka skupina lahko seveda nastopi samo enkrat, prav tako naenkrat nastopa samo ena skupina. V posameznem dnevu se skupine zvrstijo glede na njihovo ceno (tj., zadnja nastopa skupina, ki največ računa), organizator pa želi zagotoviti, da bo vsak dan festivala prišlo vsaj toliko ljudi kot prejšnji dan.

Skupina r pogojuje svoj nastop s tem, da skupina s ne nastopa na festivalu. Skupina t bo nastopila samo kot glavna skupina (tj., zadnja v posameznem dnevu), razen če nastopi neposredno pred skupino u (lahko predpostaviš, da velja $m_t \leq m_u$) – ali pa sploh ne bo nastopila. Skupini v in w ne smeta nastopati na isti dan, saj se njuni oboževalci med seboj ne marajo.

Organizator želi določiti spored tako, da bo na festival prišlo čim več ljudi. Zapiši celoštevilski linearni program, ki modelira opisani problem.

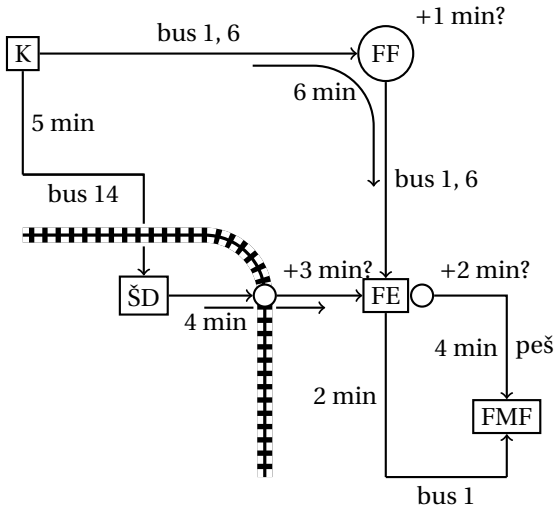
Naloga 2.12. Vir: Izpit OR 22.6.2020

Iz kriznih žarišč COVID-19 po svetu se na Kitajsko vrača n strokovnjakov, ki jih želijo oblasti razporediti po m karantenskih centrih s kapacitetami $k_1, k_2, \ldots, k_m$, pri čemer velja $\sum_{h=1}^{m} k_h \geq n$. S pomočjo naprednih tehnologij so za vsak par strokovnjakov $i, j \in \{1, 2, \ldots, n\}$ izračunali verjetnost $p_{ij} \in [0, 1]$, da je prišlo do prenosa virusa med i in j (lahko predpostaviš, da za vsaka i, j velja $p_{ij} = p_{ji}$ ter $p_{ii} = 0$). Poiskati želijo tako razporeditev, da bo vsota vrednosti p_{ij} vseh parov (i, j), ki bodo nastanjeni v istem karantenskem centru, čim manjša.

Pri tem imajo nekaj omejitev. Identificirali so množico A strokovnjakov, ki so se nahajali na območjih, kjer je prišlo do mutacije virusa – ti ne smejo biti nastanjeni skupaj s strokovnjaki izven množice A (lahko so pa člani množice A nastanjeni po različnih centrih). Poleg tega so sestavili množico B parov strokovnjakov v bližnjem sorodstvu – za vsak par $(i, j) \in B$ velja, da mora biti nastanjen v istem centru.

Zapiši celoštevilski linearni program, ki modelira opisani problem.

Namig: uporabi spremenljivke, ki za par oseb povedo, ali se nastanita v nekem centru.



Slika 1: Shema možnih poti za nalogo 3.1.

1.3 Teorija odločanja

Naloga 3.1. Vir: Izpit OR 15.12.2016

Mudi se ti na izpit, a ravno v trenutku, ko prideš na postajo Konzorcij, odpelje avtobus številka 1. Na prikazovalniku se izpiše, da bo naslednji avtobus številka 1 prispel čez 10 minut, naslednji avtobus številka 6 čez 6 minut, naslednji avtobus številka 14 pa čez 2 minuti.

Avtobusa 1 in 6 ob ugodnih semaforjih potrebujeta 6 minut do postaje pri FE, pri čemer se lahko čas vožnje zaradi rdeče luči na semaforju pri FF podaljša za 1 minuto. Verjetnosti, da bo rdečo luč imel avtobus 1, da bo rdečo luč imel avtobus 6, ter da bosta oba avtobusa imela zeleno luč, so enake $1/3$ (zaradi majhnega razmaka se ne more zgoditi, da bi oba avtobusa naletela na rdečo luč). Avtobus številka 1 nadaljuje pot do postaje pri FMF, za kar potrebuje še 2 minuti.

Avtobus številka 14 potrebuje 5 minut do postaje pri študentskih domovih, od tam pa greš peš do postaje pri FE, za kar potrebuješ še 4 minute. Pri tem prečkaš železnico – če mimo pripelje vlak (kar se zgodi z verjetnostjo 0.05), se čas hoje podaljša za 3 minute. Ko prideš na postajo pri FE (ne glede na to, ali si prišel z avtobusom 6 ali 14), te čakajo še 4 minute hoje do FMF, vendar moraš najprej prečkati Tržaško cesto. Če je na semaforju rdeča luč (kar se zgodi z verjetnostjo 0.9, neodvisno od drugih dogodkov), se lahko odločiš, da 2 minuti počakaš na zeleno luč in potem nadaljuješ peš, ali pa da greš nazaj do postaje in počakaš na avtobus številka 1 (ki bo, tako kot prej, vozil še 2 minuti do FMF).

Kakšne bodo tvoje odločitve, da bo pričakovano trajanje poti do FMF čim krajše? Nariši odločitveno drevo in odločitve sprejmi na podlagi izračunanih verjetnosti! Glej sliko 1 za shemo možnih poti.

Naloga 3.2. Vir: Izpit OR 31.1.2017

Dve podjetji bosta predstavili konkurenčna izdelka. Možnost imaš kupiti delnice prvega podjetja za ceno 10 000€ ali delnice drugega podjetja po ceni 5 000€, lahko pa se seveda odločiš tudi, da delnic ne kupiš. Ocenjuješ, da bo z verjetnostjo 0.4 uspelo prvo podjetje, z verjetnostjo 0.1 bo uspelo drugo podjetje, z verjetnostjo 0.5 pa ne bo uspelo nobeno izmed njiju (ne more se zgoditi, da bi obe uspeli). Ob uspehu prvega podjetja se bo vrednost njihovih delnic potrojila, ob uspehu drugega podjetja pa se bo vrednost njihovih delnic popeterila – če si lastiš delnice uspešnega podjetja, jih torej lahko prodaš, pri čemer bo torej dobiček enak dvakratniku oziroma štirikratniku vloženega zneska. Delnic neuspešnega podjetja ne bo želel nihče kupiti, tako da je v tem primeru vloženi znesek izgubljen.

Za mnenje lahko povprašaš tržnega izvedenca, ki bo po opravljeni raziskavi povedal, katero od dveh podjetij ima večje možnosti za uspeh. Če bo uspešno prvo podjetje, bo to pravilno napovedal z verjetnostjo 0.8, če pa bo uspešno drugo podjetje, bo to pravilno napovedal z verjetnostjo 0.7. V primeru, ko podjetji ne bosta uspeli, bo z verjetnostjo 0.4 večje možnosti pripisal prvemu, z verjetnostjo 0.6 pa drugemu podjetju. Za svoje mnenje izvedenec računa 1 000€.

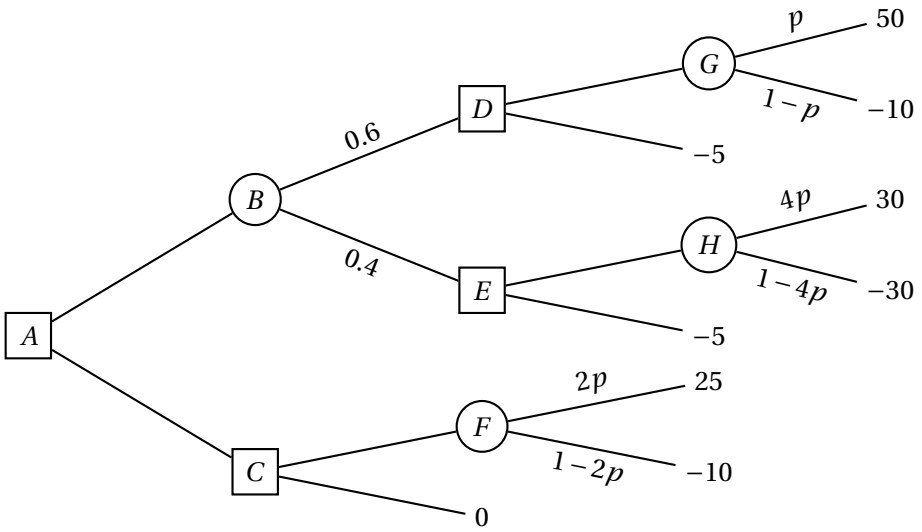
Kakšne bodo tvoje odločitve, da bo tvoj pričakovani dobiček po odprodaji delnic čim večji? Nariši odločitveno drevo in odločitve sprejmi na podlagi izračunanih verjetnosti. Pričakovani dobiček tudi izračunaj.

Naloga 3.3. Vir: Izpit OR 10.7.2017

Proti računalniškemu programu igraš Texas hold 'em poker. Pravila igre tukaj niso pomembna. Ker imaš dostop do kode programa, poznaš logiko, po kateri se ravna. V trenutni igri si vložil 30 žetonov, enako tudi nasprotnik. Nasprotnik z verjetnostjo 0.6 meni, da so odprte karte ugodnejše zate, z verjetnostjo 0.4 pa, da so ugodnejše zanj (sam si ne ustvariš nobenega mnenja). V prvem primeru je verjetnost, da so dejansko tvoje karte boljše, enaka 0.8, v drugem pa le 0.1.

Nasprotnik se bo sedaj odločil, ali naj vloži še 10 žetonov. Sam se lahko nato odločiš, ali boš vložil 0, 10 ali 20 žetonov (skupni vložek bo torej 30, 40 ali 50 žetonov). Če je tvoj vložek manjši od nasprotnikovega, je igra izgubljena in izgubiš do sedaj vloženo. Če je tvoj vložek enak nasprotnikovemu, z nasprotnikom pogledata karte in tako določita zmagovalca. Če je tvoj vložek višji od nasprotnikovega, ima ta možnost odstopiti (tako pridobiš nasprotnikov vložek), ali pa izenačiti, nakar se zmagovalec določi na podlagi kart. Če zmagaš, pridobiš nasprotnikov vložek, če izgubiš, pa izgubiš svojega.

V spodnji tabeli so zbrane verjetnosti dogodkov v odvisnosti od nasprotnikovega mnenja glede kart. Verjetnosti navedenih dogodkov pri istem mnenju so med seboj neodvisne.



Slika 2: Odločitveno drevo za nalogo 3.4.

dogodek \ nasprotnikovo mnenje	ugodnejše karte zate	za nasprotnika
dejansko imaš boljše karte	0.8	0.1
nasprotnik vloži 10 žetonov po razkritju karte	0.3	0.8
nasprotnik izenači skupni vložek 40 žetonov	0.2	0.7
nasprotnik izenači skupni vložek 50 žetonov	0.1	0.8

Na primer:

Pr[nasprotnik izenači skupni vložek 40 žetonov |

| nasprotnikovo mnenje je “ugodnejše karte zate”] = 0.2.

Kakšne bodo tvoje odločitve, da bo tvoj pričakovani dobiček po koncu igre čim večji? Nariši odločitveno drevo in odločitve sprejmi na podlagi izračunanih verjetnosti. Pričakovani dobiček tudi izračunaj.

Naloga 3.4. Vir: Izpit OR 29.8.2017

Dano je odločitveno drevo s slike 2, pri čemer velja $0 \leq p \leq 1/4$. Pričakovano vrednost želimo maksimizirati. Poišči optimalne odločitve in pričakovano vrednost v odvisnosti od vrednosti parametra p.

Naloga 3.5. Vir: Kolokvij OR 23.4.2018

Mladi podjetnik je razvil inovativen izdelek in se odloča za nadaljnje korake pri njegovem trženju. Naenkrat lahko naroči izdelavo 500 izdelkov po ceni 10 000€ ali 1 000 izdelkov po ceni 18 000€, lahko se pa odloči tudi, da izdelave ne naroči. Podjetnik ocenjuje, da je izdelek tržno zanimiv z verjetnostjo 0.8. Odloča se, ali naj posamezen izdelek prodaja po ceni 50€ ali 60€, pri čemer so v spodnji tabeli zbrana pričakovana števila prodanih kosov v odvisnosti od teh pogojev.

	cena 50€	cena 60€
tržno zanimiv	650	550
tržno nezanimiv	250	100

Predpostavi, da se bodo prodali vsi izdelani kosi, če je število teh manjše od pričakovane prodaje pri danih pogojih.

(a) Kako naj se podjetnik odloči, da bo pričakovani zaslužek čim večji? Nariši odločitveno drevo in ga uporabi pri sprejemanju odločitev.

(b) V zadnjem trenutku je podjetnik izvedel, da bo Podjetniški pospeševalnik objavil razpis za nagrado za najboljši izdelek. Razpisni pogoji zahtevajo, da se za potrebo kontrole kvalitete izdela vsaj 1 000 izdelkov, od katerih komisija izbere 20 za dejansko kontrolo, z ostalimi pa lahko prijavitelj prosto razpolaga. Če podjetnik zmaga, bo dobil nagrado v višini k, pri čemer $k \in [1\,000€, 5\,000€]$ še ni znan, poleg tega pa si obeta tudi 20% povečanje pričakovanega števila prodanih kosov (v vseh zgoraj omenjenih pogojih). Če ne zmaga, se pričakovanja ne spremenijo. Podjetnik ocenjuje, da je verjetnost zmage enaka 0.6, če je izdelek tržno zanimiv, in 0.1, če izdelek ni tržno zanimiv.

Naj se podjetnik prijavi na razpis? Nariši odločitveno drevo in odločitve sprejmi v odvisnosti od parametra k!

Naloga 3.6. Vir: Izpit OR 11.6.2018

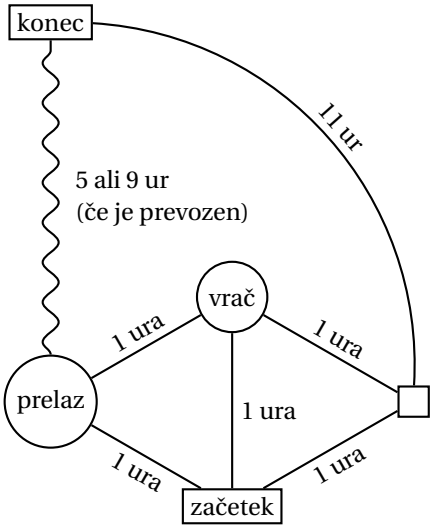
Podajamo se na pot v mesto na drugi strani gorovja. Lahko se odločimo za pot po cesti okoli gorovja, za kar bomo porabili 12 ur. Vendar pa nas najkrajša pot vodi čez prelaz, ki je eno uro stran od začetne lokacije. Žal pa so razmere v gorah nepredvidljive: ocenjujemo, da bo z verjetnostjo 0.2 prelaz čist in ga bomo lahko prevozili v 5 urah, z verjetnostjo 0.1 bo delno zasnežen in ga bomo prevozili v 9 urah, z verjetnostjo 0.7 pa bo neprevozen, zaradi česar se bomo morali vrniti na začetek in iti okoli gorovja (skupno trajanje poti bo v tem primeru torej 14 ur).

Edini, ki nam lahko kakorkoli pomaga pri oceni vremenskih razmer na prelazu, je lokalni vrač, ki pa živi v dolini pod goro in ne uporablja sodobne tehnologije, s katero bi ga lahko kontaktirali. Rade volje pa nam bo povedal, ali na gori vlada mir, če se na poti do prelaza ustavimo pri njem. Pogojne verjetnosti njegovega odgovora v odvisnosti od razmer na prelazu so podane v spodnji tabeli.

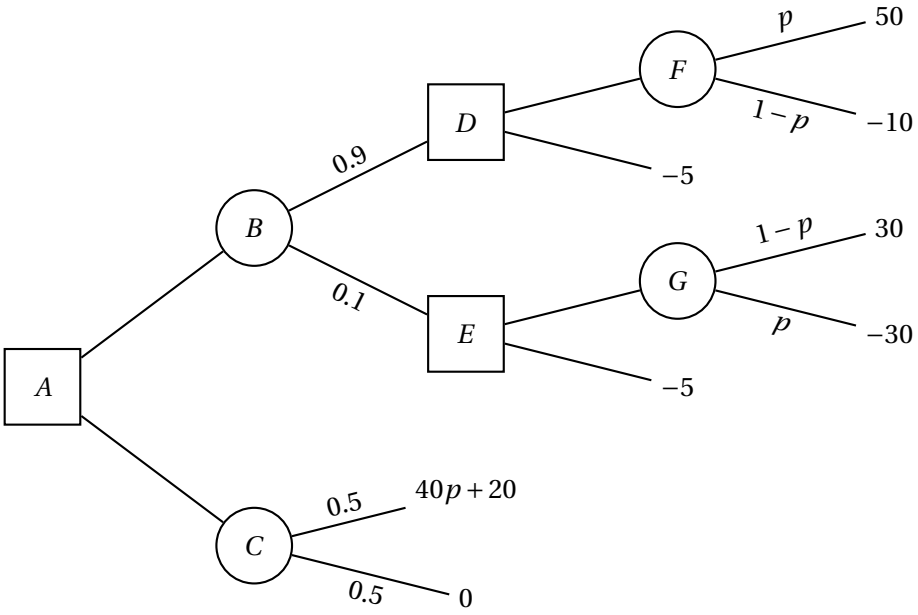
P(vračev odgovor | razmere na prelazu)	čist	delno zasnežen	neprevozen
na gori vlada mir	0.9	0.5	0.1
na gori divja vojna	0.1	0.5	0.9

Do vrača imamo eno uro vožnje, do prelaza pa potem še eno uro. Če se po obisku vrača odločimo za pot okoli gorovja, bomo za nadaljnjo pot porabili 12 ur.

Kako se bomo odločili? Nariši odločitveno drevo in ga uporabi pri sprejemanju odločitev. Izračunaj tudi pričakovano trajanje poti. Glej sliko 3 za shemo možnih poti.



Slika 3: Shema možnih poti za nalogo 3.6.



Slika 4: Odločitveno drevo za nalogo 3.7.

Naloga 3.7. Vir: Izpit OR 28.8.2018

Dano je odločitveno drevo s slike 4, pri čemer velja $0 \leq p \leq 1$. Pričakovano vrednost želimo maksimizirati. Poišči optimalne odločitve in pričakovano vrednost v odvisnosti od vrednosti parametra p.

Naloga 3.8. Vir: Kolokvij OR 19.4.2019

V manjšem podjetju so razvili nov okus sadnega soka. Proizvedli so ga že 100 000 litrov, ko so prejeli ponudbo multinacionalke, da odkupijo vso razpoložljivo količino po ceni 1€ na liter. Če se ne odločijo za odprodajo, bodo sok sami pakirali in ponudili na trgu po ceni 3€ na liter. Zagon pakiranja stane 1 000€, pakiranje enega litra soka pa 0.5€. V podjetju ocenjujejo, da bo z verjetnostjo 0.7 produkt uspešen in ga bodo vsega prodali, z verjetnostjo 0.3 pa bo neuspešen in ga bodo prodali le 10 000 litrov.

Pri podjetju imajo na voljo ravno dovolj časa, da pred odločitvijo pakirajo in na regionalnem trgu ponudijo 1 000 litrov soka po akcijski ceni 2.5€ na liter. Ocenjujejo, da bi v primeru uspeha produkta z verjetnostjo 0.8 tudi akcija uspela in bi tako razprodali akcijske zaloge, v primeru neuspeha produkta pa bi se to zgodilo le z verjetnostjo 0.1. Če akcija ne bi uspela, bi prodali le 100 litrov soka. Če se po akcijski ponudbi odločijo za samostojno prodajo, bodo seveda morali še enkrat plačati stroške zagona pakiranja, v nasprotnem primeru pa bo multinacionalka odkupila preostalih 99 000 litrov še nepakiranega soka.

Opomba: količina, prodana v akcijski ponudbi, je vključena v končno prodano količino. Če torej v akciji prodajo vseh 1 000 litrov soka po ceni 2.5€ na liter, ga bodo v primeru neuspeha produkta v redni prodaji prodali le še 9 000 litrov po ceni 3€ na liter, v primeru uspeha pa še 99 000 litrov.

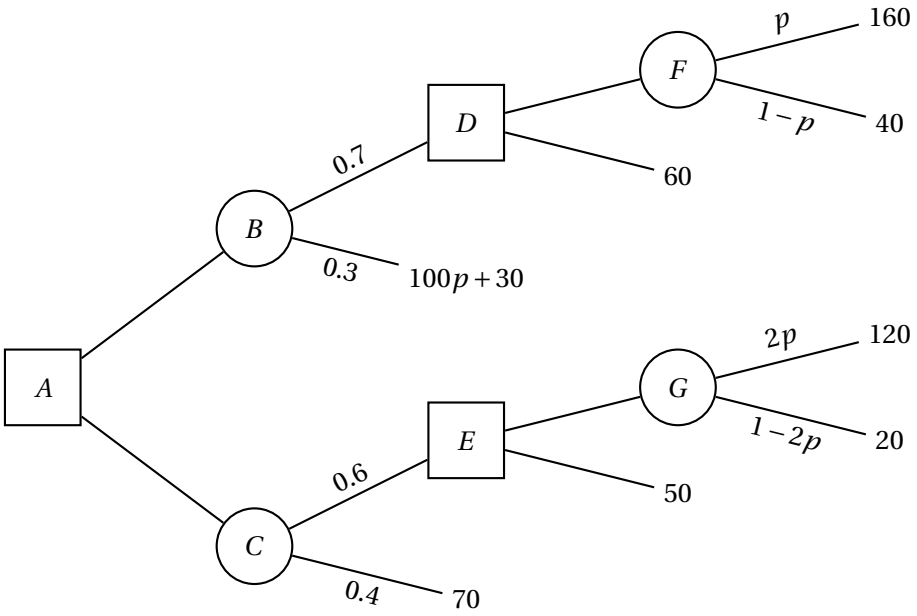
Kako naj se pri podjetju odločijo, da bo pričakovani zaslužek čim večji? Nariši odločitveno drevo in ga uporabi pri sprejemanju odločitev.

Naloga 3.9. Vir: Izpit OR 3.6.2019

Podjetje je od konkurenta v finančnih težavah odkupilo tovarno čokolade, ki že nekaj časa ni bila v uporabi. Ker že prejemajo nova naročila, želijo v naslednjih 8 tednih proizvesti čim večjo količino čokolade. Ocenjujejo, da so z verjetnostjo 0.4 stroji v dobrem stanju in lahko proizvedejo 10 000 kg čokolade na teden, z verjetnostjo 0.6 pa so v slabem stanju in lahko proizvedejo le 1 000 kg čokolade na teden.

Odločijo se lahko, da pred zagonom proizvodnje na strojih izvedejo servis. Izvedba servisa lahko poteka gladko in konča v enem tednu, ali pa se pojavijo težave in tako servis traja tri tedne (za proizvodnjo tako ostane le še 7 oziroma 5 tednov). Če so stroji v dobrem stanju, bo servis z verjetnostjo 0.9 potekal gladko, z verjetnostjo 0.1 pa se bodo pojavile težave – v obeh primerih pa bodo stroji po servisu ostali v dobrem stanju. Če so stroji v slabem stanju, bo servis z verjetnostjo 0.2 potekal gladko in jih tedaj z verjetnostjo 0.7 spravil v dobro stanje, z verjetnostjo 0.8 pa se bodo pojavile težave – tedaj je verjetnost, da bodo stroji po servisu v dobrem stanju, enaka 0.5. Po zagonu proizvodnje te zaradi previsokih stroškov ni mogoče predčasno prekiniti; stroški servisa pa niso pomembni, saj ga bodo morali izvesti kasneje, če se zanj ne odločijo takoj.

Kako naj se v podjetju odločijo? Nariši odločitveno drevo in ga uporabi pri sprejemanju odločitev. Izračunaj tudi pričakovano količino proizvedene čoko-



Slika 5: Odločitveno drevo za nalogo 3.10.

lade.

Naloga 3.10. Vir: Izpit OR 19.6.2019

Dano je odločitveno drevo s slike 5, pri čemer velja $0 \leq p \leq 1/2$. Pričakovano vrednost želimo minimizirati. Poišči optimalne odločitve in pričakovano vrednost v odvisnosti od vrednosti parametra p.

Naloga 3.11. Vir: Izpit OR 4.6.2020

Na trajektu stoji 200 avtomobilov, vozovnica pa stane 35€. Pri družbi, ki upravlja trajekt, se lahko odločijo prodati 200, 201, 202 ali 203 vozovnice. Naj bo p_i verjetnost, da i avtomobilov z vozovnicami ne pride do trajekta (tj., p_0 je verjetnost, da vsi pridejo):

i	0	1	2	3
p_i	0.3	0.4	0.2	0.1

Z vsakim avtomobilom, ki pride do trajekta in se ne more vkrcati, ima družba 60€ stroškov (skupaj torej 25€ izgube).

(a) Koliko vozovnic naj proda družba, da bo imela čim večji dobiček?

(b) Na trajektu je na voljo še eno mesto, ki pa ni najbolj varno[1] – avtomobil, ki se pelje na njem, bo z verjetnostjo 0.001 (neodvisno od ostalih dejavnikov) med

[1] Seveda bo to mesto ostalo prazno, če bo to mogoče.

potjo padel v morje. V tem primeru ima družba dodatnih 40 000€ stroškov. Ali se družbi izplača uporabiti to mesto? Koliko vozovnic naj tedaj proda?

1.4 Dinamično programiranje

Naloga 4.1. Vir: [1, Exercise 6.14]

Imamo pravokoten kos blaga dimenzij $m \times n$, kjer sta m in n pozitivni celi števili, ter seznam k izdelkov, pri čemer potrebujemo za izdelek h pravokoten kos blaga dimenzij $a_h \times b_h$ (a_h, b_h sta pozitivni celi števili), ki ga prodamo za ceno $c_h > 0$. Imamo stroj, ki lahko poljuben kos blaga razreže na dva dela bodisi vodoravno, bodisi navpično. Začetni kos blaga želimo razrezati tako, da bomo lahko naredili izdelke, ki nam bodo prinašali čim večji dobiček. Pri tem smemo izdelati poljubno število kosov posameznega izdelka. Kose blaga lahko seveda tudi obračamo (tj., za izdelek h lahko narežemo kos velikosti $a_h \times b_h$ ali $b_h \times a_h$).

Zapiši rekurzivne enačbe za reševanje danega problema. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev.

Naloga 4.2. Vir: Izpit OR 15.12.2016

Oceni časovno zahtevnost algoritma, ki sledi iz rekurzivnih enačb za nalogo 4.1. Reši problem za podatke $m = 5, n = 3, k = 4$, $(a_h)_{h=1}^k = (2,3,1,2)$, $(b_h)_{h=1}^k = (2,1,4,3)$ in $(c_h)_{h=1}^k = (6,3,5,7)$.

Naloga 4.3. Vir: Izpit OR 31.1.2017

Dano je zaporedje n realnih števil $a_1, a_2, \ldots, a_n$. Želimo poiskati strnjeno podzaporedje z največjim produktom – t.j., taka indeksa i, j ($1 \le i \le j \le n$), da je produkt $a_i a_{i+1} \cdots a_{j-1} a_j$ čim večji.

(a) Zapiši rekurzivne enačbe za reševanje danega problema. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev.
Namig: posebej obravnavaj pozitivne in negativne delne produkte.

(b) Oceni časovno zahtevnost algoritma, ki sledi iz zgoraj zapisanih enačb.

(c) S svojim algoritmom reši problem za zaporedje

$$0.9,\ -2,\ -0.6,\ -0.5,\ -2,\ 5,\ 0.1,\ 3,\ 0.5,\ -3\,.$$

Naloga 4.4. Vir: Izpit OR 10.7.2017

V podjetju imajo na voljo m milijonov evrov sredstev, ki jih bodo vložili v razvoj nove aplikacije. Denar bodo porazdelili med tri skupine. Naj bodo x_1, x_2 in x_3 količine denarja (v milijonih evrov), ki jih bodo dodelili razvijalcem, oblikovalcem in marketingu. Vrednosti x_1, x_2, x_3 niso nujno cela števila. Razvijalci morajo dobiti vsaj a_1 milijonov evrov, potencial, ki ga ustvarijo, pa je $p_1 = n_1 + k_1 x_1$. Oblikovalci morajo dobiti vsaj a_2 milijonov evrov, potencial, ki ga ustvarijo, pa je

$p_2 = n_2 + k_2 x_2$. Marketing mora dobiti vsaj a_3 milijonov evrov, ustvari pa faktor $p_3 = n_3 + k_3 x_3$. Pričakovani dobiček v milijonih evrov se izračuna po formuli $d = (p_1 + p_2) p_3$. V podjetju bi radi sredstva porazdelili med skupine tako, da bo pričakovani dobiček čim večji.

(a) Zapiši rekurzivne enačbe za reševanje danega problema.

(b) Z zgoraj zapisanimi enačbami reši problem pri podatkih $m = 15$, $a_1 = 4$, $n_1 = 3$, $k_1 = 1.5$, $a_2 = 3$, $n_2 = 4$, $k_2 = 2$, $a_3 = 2$, $n_3 = 0.4$ in $k_3 = 0.3$.

Naloga 4.5. Vir: Izpit OR 29.8.2017

V veliki multinacionalni korporaciji želijo, da bi se zakonodaja spremenila v njihov prid. V ta namen so najeli m lobistov, ki se bodo pogajali z n političnimi strankami, da pridobijo njihovo podporo pri spremembi zakonodaje. Vsak lobist se bo pogajal s samo eno stranko; k vsaki stranki lahko pošljejo več lobistov. Naj bo p_{ij} ($0 \le i \le m$, $1 \le j \le n$) verjetnost, da pridobijo podporo stranke j, če se z njo pogaja i lobistov (lahko predpostaviš $p_{i-1,j} \le p_{ij}$ za vsaka i, j). Verjetnosti za različne stranke so med seboj neodvisne. Maksimizirati želijo verjetnost, da bodo lobisti pridobili podporo vseh n političnih strank.

(a) Zapiši rekurzivne enačbe za reševanje danega problema.

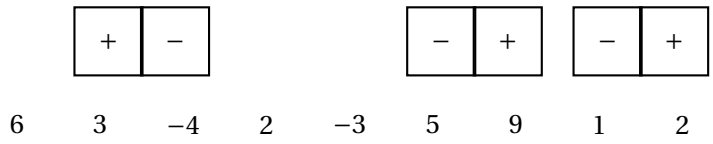
(b) Naj bo $m = 6$ in $n = 3$. K vsaki stranki želijo poslati vsaj enega lobista (tj., $p_{0j} = 0$ za vsak j), vrednosti p_{ij} za $i \ge 1$ pa so podane v spodnji tabeli.

p_{ij}	1	2	3
1	0.2	0.4	0.3
2	0.5	0.5	0.4
3	0.7	0.5	0.8
4	0.8	0.6	0.9

Za dane podatke reši problem z zgoraj zapisanimi enačbami.

Naloga 4.6. Vir: Kolokvij OR 23.4.2018

Imamo zaporedje n polj, pri čemer je na i-tem polju zapisano število a_i. Na voljo imamo še $\lfloor n/2 \rfloor$ domin, z vsako od katerih lahko pokrijemo dve sosednji polji. Vsaka domina je sestavljena iz dveh delov: na enem je znak +, na drugem pa znak −. Posamezno polje lahko pokrijemo z le eno domino; če sta pokriti dve sosednji polji, morata biti pokriti z različnima znakoma (bodisi z iste, bodisi z druge domine). Iščemo tako postavitev domin, ki maksimizira vsoto pokritih števil, pomnoženih z znakom na delu domine, ki pokriva število. Pri tem ni potrebno, da uporabimo vse domine. Primer je podan na sliki 6.



Slika 6: Primer dopustnega (ne nujno optimalnega) pokritja za nalogo 4.6. Vsota tega pokritja je $3-(-4)-5+9-1+2=12$. Če bi eno od zadnjih dveh domin obrnili (zamenjala bi se znaka), dobljeno pokritje ne bi bilo dopustno, saj bi dve zaporedni polji bili pokriti z enakima znakoma.

(a) Zapiši rekurzivne enačbe za reševanje danega problema. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev.
Namig: posebej obravnavaj dva primera glede na postavitev zadnje domine.

(b) Oceni časovno zahtevnost algoritma, ki sledi iz zgoraj zapisanih enačb.

(c) S svojim algoritmom poišči optimalno pokritje za primer s slike 6.

Naloga 4.7. Vir: Izpit OR 5.7.2018

Vlagatelj ima na voljo 50 milijonov evrov sredstev, ki jih lahko porabi za donosno, a tvegano naložbo. Ocenjuje, da bi se mu ob uspehu naložbe vložek povrnil petkratno, verjetnost uspeha pa ocenjuje na 0.6. Zaradi tveganja se lahko odloči za zavarovanje naložbe, pri čemer ima ponudbi dveh zavarovalnic, ki mu proti plačilu ustrezne premije ponujata povračilo dela vložka, če bo naložba neuspešna. Vlagatelj lahko del sredstev obdrži tudi zase (tj., ga ne porabi za naložbo ali premijo).

Naj bodo torej x_1, x_2, x_3 vrednosti v milijonih evrov, ki zaporedoma predstavljajo količine, ki jih vlagatelj obrži zase, porabi za naložbo, in plača za zavarovalniško premijo. Pričakovana vrednost naložbene strategije vlagatelja (tj., količina denarja, ki jo ima na koncu) je potem

$$x_1 + x_2(0.6 \cdot 5 + 0.4\,q(x_3)),$$

kjer $q(x_3)$ predstavlja delež vložka, ki ga glede na vloženo premijo zavarovalnica povrne ob neuspehu naložbe.

Vlagatelj ima dve ponudbi konkurenčnih zavarovalnic. Zavarovalnica Zvezna d.z.z. za premijo v višini x_3 milijonov evrov ponuja povračilo deleža $0.15x_3$ celotne naložbe v primeru neuspeha, pri čemer je največja možna premija 4 milijone evrov. Zavarovalnica Diskretna d.d.z. pa ponuja le tri možne premije:

premija	delež povračila ob neuspešni naložbi
1 milijon evrov	0.1
2 milijona evrov	0.35
3 milijoni evrov	0.5

Pogodbo smemo skleniti samo pri eni zavarovalnici.

(a) Zapiši definicijo funkcije $q(x)$ skupaj z izbiro najugodnejše zavarovalnice pri vsakem x.

(b) Zapiši rekurzivne formule za določitev strategije vlaganja, ki nam bo prinesla največji pričakovani dobiček.

(c) S pomočjo zgornjih rekurzivnih enačb ugotovi, kako naj ravna vlagatelj, da bo imel čim večji dobiček.

Naloga 4.8. Vir: Izpit OR 28.8.2018

Pri direkciji za ceste načrtujejo nov avtocestni odsek dolžine M kilometrov. Ob cesti želijo zgraditi počivališča tako, da je razdalja med dvema zaporednima počivališčema največ K kilometrov. Prav tako mora biti prvo počivališče največ K kilometrov od začetka, zadnje pa največ K kilometrov od konca avtocestnega odseka. Naj bodo $x_1 < x_2 < \cdots < x_n$ možne lokacije počivališč (v kilometrih od začetka avtocestnega odseka), in c_i $(1 \leq i \leq n)$ cena izgradnje počivališča na lokaciji x_i. Postavitev počivališč želijo izbrati tako, da bo skupna cena izgradnje čim manjša.

(a) Zapiši rekurzivne enačbe za reševanje danega problema. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev.

(b) Oceni časovno zahtevnost algoritma, ki sledi iz zgoraj zapisanih enačb.

(c) S pomočjo rekurzivnih enačb reši zgornji problem za podatke

$$M = 100, \qquad (x_i)_{i=1}^{8} = (5, 12, 22, 34, 49, 65, 83, 91),$$
$$K = 30, \qquad (c_i)_{i=1}^{8} = (18, 11, 21, 16, 23, 15, 19, 13).$$

Naloga 4.9. Vir: Kolokvij OR 19.4.2019

Gradimo avtocesto skozi puščavo in želimo zagotoviti, da bo v celoti pokrita z mobilnim signalom. Cesta je ravna in dolga n milj ($n \in \mathbb{Z}$), na vsako miljo pa imamo možnost postaviti bazno postajo z dosegom 1 miljo. Cena postavitve bazne postaje na i-ti milji je podana s parametrom a_i $(0 \leq i \leq n)$. Predpostaviš lahko, da so vse cene pozitivne. Pri obstoječi infrastrukturi je pokrita le začetna točka avtoceste (tj., milja 0). Poiskati želimo torej čim cenejšo postavitev baznih postaj, da je vsaka točka avtoceste pokrita s signalom.

Primer: če postavimo postaji na milji 0 in 3, potem je interval $(1,2)$ nepokrit. Če pa npr. postajo namesto na milji 0 postavimo na milji 1, smo tako v celoti pokrili interval $[0,4]$.

(a) Zapiši rekurzivne enačbe za reševanje danega problema. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev.

(b) Oceni časovno zahtevnost algoritma, ki sledi iz zgoraj zapisanih enačb.

(c) S svojim algoritmom poišči optimalno postavitev postaj za $n = 10$ ter

$$(a_i)_{i=0}^{10} = (4, 6, 1, 10, 14, 21, 15, 6, 10, 3, 2).$$

(d) Denimo, da lahko postavimo tudi večje bazne postaje z dosegom 2 milji, pri čemer je cena postavitve take postaje na i-ti milji podana s parametrom b_i ($0 \leq i \leq n$). Zapiši rekurzivne enačbe, ki bodo upoštevale tudi to možnost.

(e) Problem iz točke (d) reši s podatki iz točke (c) in

$$(b_i)_{i=0}^{10} = (10, 12, 3, 18, 24, 25, 20, 11, 16, 7, 4).$$

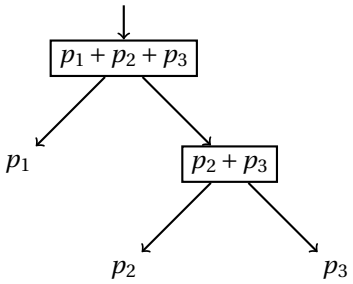
Naloga 4.10. Vir: Izpit OR 19.6.2019

Za novo postavljeno obrtno cono želimo napeljati vodovodno omrežje. Imamo n delavnic v ravni vrsti, za i-to delavnico pa imamo podano porabo vode p_i. Iz javnega vodovoda bodo napeljali cev, preko katere bo priteklo dovolj vode za vseh n delavnic, mi pa želimo postaviti razdelilnike, ki bodo poskrbeli za ustrezno razdelitev vode med delavnice. Vsak razdelilnik ima eno vhodno cev in dve izhodni, vsaka od njiju pa bo pripeljala vodo do zaporednih delavnic (po potrebi preko nadaljnjih razdelilnikov). Cena postavitve razdelilnika je sorazmerna porabi delavnic, ki jim služi. Razdelilnike želimo postaviti tako, da bo skupna cena čim manjša.

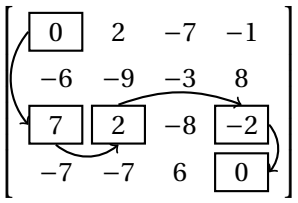
Primer: na sliki 7 je prikazana možna postavitev razdelilnikov za tri delavnice. Odločimo se lahko, ali bomo najprej razdelili vodo delavnici 1 in delavnicama 2, 3, ali pa bomo vodo peljali naprej do delavnic 1, 2 in do delavnice 3 (ne moremo pa deliti na delavnici 1, 3 in delavnico 2). Če se odločimo za prvi primer, potem do delavnice 1 ne potrebujemo drugih razdelilnikov, še enega pa moramo postaviti, da razdelimo vodo do delavnic 2 in 3. Cena postavitve prvega razdelilnika je tako $p_1 + p_2 + p_3$ (ne glede na odločitev), za drugega pa je v tem primeru cena $p_2 + p_3$.

(a) Naj bo $c_i = \sum_{h=1}^{i} p_h$ ($0 \leq i \leq n$). Zapiši rekurzivne enačbe za čim učinkovitejši izračun teh vrednosti.

(b) Zapiši rekurzivne enačbe za reševanje danega problema. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev. Kakšna je časovna zahtevnost algoritma?
Namig: pomagaj si z vrednostmi c_i iz prejšnje točke.



Slika 7: Primer postavitve razdelilnikov za nalogo 4.10.



Slika 8: Primer matrike dimenzij 4×4 za nalogo 4.11 skupaj z dopustno (ne nujno optimalno!) rešitvijo. Vsota obiskanih mest je v tem primeru $0+7+2+(-2)+0=7$.

(c) S pomočjo rekurzivnih enačb reši zgornji problem za $n = 6$ in

$$(p_i)_{i=1}^{6} = (4, 19, 17, 7, 5, 9).$$

Naloga 4.11. Vir: Izpit OR 4.6.2020

Dana je matrika A dimenzij $m \times n$. Iščemo tako zaporedje skokov po matriki, pri čemer začnemo levo zgoraj in končamo desno spodaj, v vsakem koraku pa skočimo za vsaj eno mesto desno ali dol, pri katerem je vsota obiskanih mest čim večja. Drugače povedano, iščemo tako zaporedje indeksov $(i_1, j_1), (i_2, j_2), \ldots (i_k, j_k)$ (za nek $k \geq 1$), za katere velja $(i_1, j_1) = (1, 1)$, $(i_k, j_k) = (m, n)$ in $(i_h < i_{h+1} \wedge j_h = j_{h+1}) \vee (i_h = i_{h+1} \wedge j_h < j_{h+1})$ za vse h $(2 \leq h \leq k)$, ki maksimizira vsoto $\sum_{h=1}^{k} A_{i_h, j_h}$. Primer je podan na sliki 8.

(a) Zapiši začetne pogoje in rekurzivne enačbe za reševanje opisanega problema ter določi, v kakšnem vrstnem redu računamo spremenljivke in kako dobimo maksimalno vsoto. Kakšna je časovna zahtevnost algoritma, ki sledi iz rekurzivnih enačb?

(b) Reši problem za matriko s slike 8 z uporabo rekurzivnih enačb.

Naloga 4.12. Vir: Izpit OR 27.8.2019

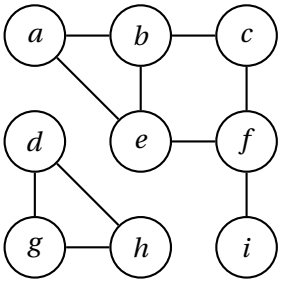
Vlagatelj ima na voljo m kapitala, del katerega bo vložil v eno izmed dveh konkurenčnih podjetij, ki razvijata podobna produkta (v obe ne more vložiti), preostanek pa bo namenil za marketinško kampanjo, ki bo promovirala produkt izbranega podjetja. Naj bodo x_1, x_2 in x_3 količine denarja (te so lahko poljubna nenegativna realna števila), ki jih bo vložil v prvo oziroma drugo podjetje in v marketing, ter določimo

$$p_i = \begin{cases} n_i + k_i x_i, & \text{če } x_i \geq a_i, \text{ in} \\ 0 & \text{sicer} \end{cases} \qquad (i = 1,2,3)$$

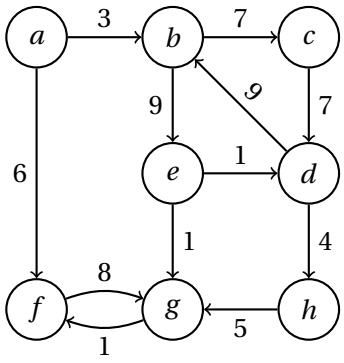
kot faktor, ki ga ustvari vsako izmed njih. Pričakovani dobiček se izračuna po formuli $d = (p_1 + p_2)p_3$, pri čemer bo eden od p_1 in p_2 enak 0. Vlagatelj bi rad sredstva porazdelil tako, da bo pričakovani dobiček čim večji.

(a) Zapiši enačbe za reševanje danega problema. Lahko predpostaviš, da velja $p_i \geq 0$ za vse vrednosti x_i $(i = 1,2,3)$.

(b) Z zgoraj zapisanimi enačbami reši problem pri podatkih $m = 10$, $a_1 = 4$, $n_1 = 8$, $k_1 = 4$, $a_2 = 5$, $n_2 = 12$, $k_2 = 2$, $a_3 = 3$, $n_3 = 4$ in $k_3 = 1$.



Slika 9: Graf za nalogi 5.1 in 5.3.



Slika 10: Graf za nalogo 5.2.

1.5 Algoritmi na grafih

Naloga 5.1. Vir: Vaje OR 30.11.2016

Na grafu s slike 9 izvedi iskanje v širino. V primerih, ko imaš več enakovrednih izbir, upoštevaj abecedni vrstni red. Za vsako povezavo določi, ali se nahaja v drevesu iskanja v širino.

Naloga 5.2. Vir: Vaje OR 7.12.2016

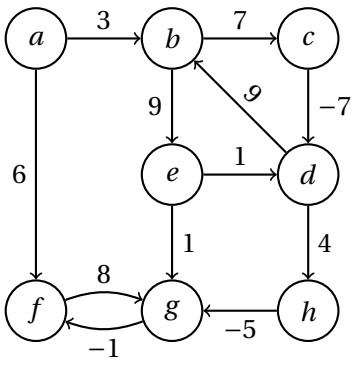
S pomočjo Dijkstrovega algoritma določi razdalje od vozlišča a do ostalih vozlišč v grafu s slike 10.

Naloga 5.3. Vir: [1, Exercise 3.1]

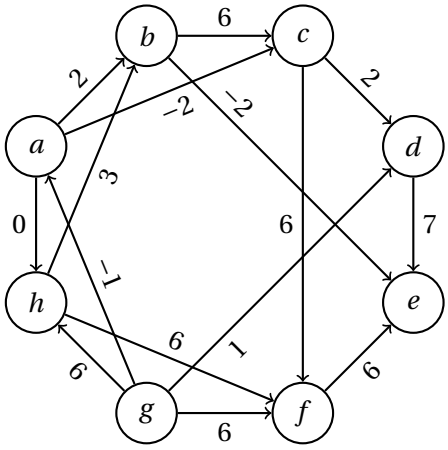
Na grafu s slike 9 izvedi iskanje v globino. V primerih, ko imaš več enakovrednih izbir, upoštevaj abecedni vrstni red. Za vsako povezavo določi, ali se nahaja v drevesu iskanja v globino.

Naloga 5.4. Vir: Vaje OR 21.5.2018

S pomočjo Bellman-Fordovega algoritma določi razdalje od vozlišča a do ostalih vozlišč v grafu s slike 11.



Slika 11: Graf za nalogi 5.4 in 5.17.



Slika 12: Graf za nalogo 5.5.

Naloga 5.5. Vir: Vaje OR 7.12.2016

Dan je usmerjen acikličen graf s slike 12.

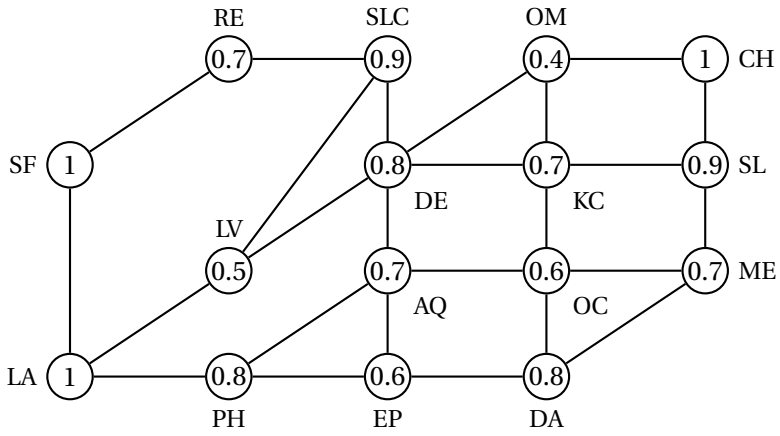
(a) Poišči topološko ureditev vozlišč zgornjega grafa.

(b) Poišči najkrajšo pot od vozlišča g do vozlišča e.

(c) Poišči najdaljšo pot od vozlišča g do vozlišča e.

Naloga 5.6. Vir: Izpit OR 15.12.2016

Lovec na zaklade se z bogatim ulovom vrača iz Kalifornije nazaj domov v Chicago, pri čemer mora seveda prečkati Divji zahod. Potoval bo s kočijo, pri čemer bo vsak dan potoval med dvema mestoma in nato prespal. Zaradi varnosti se bo držal samo državnih cest, ki so varne. Toda mesta, kjer bo prespal, niso povsem varna. Za vsako mesto pozna verjetnosti, da ga tam ne bodo oropali (te so med



Slika 13: Graf za nalogo 5.6.

seboj neodvisne). Tako bi želel načrtovati najvarnejšo pot domov – torej pot z največjo verjetnostjo, da ga pri nobenem postanku ne bodo oropali.

(a) Mesta in ceste med njimi lahko predstavimo z vozlišči in povezavami v neusmerjenem grafu G, verjetnosti pa kot teže vozlišč. Opiši, kako lahko za dani graf G z uteženimi vozlišči učinkovito poiščemo ustrezno pot med danima vozliščema s in t z uporabo variante Dijkstrovega algoritma, ter utemelji njegovo ustreznost. Lahko predpostaviš, da sta teži začetnega in končnega vozlišča enaki 1.

(b) Reši problem za graf s slike 13, pri čemer naj se pot začne v LA in konča v CH. Zadostovalo bo, če verjetnosti računaš na 3 decimalke natančno.

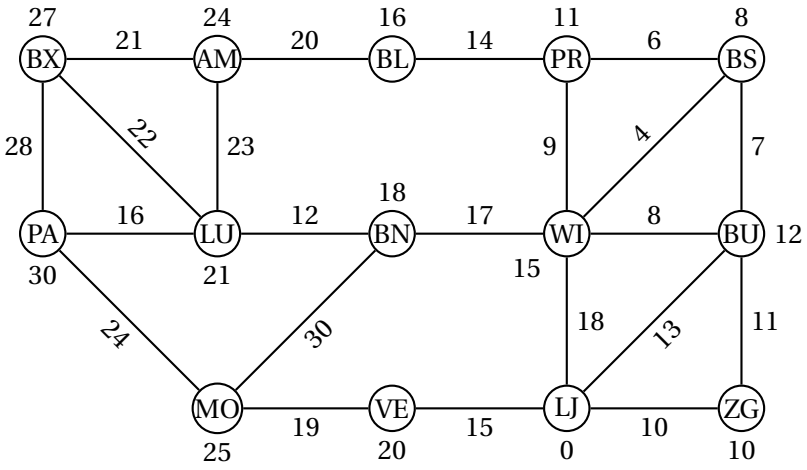
Naloga 5.7. Vir: Izpit OR 10.7.2017

Z električnim vozilom se odpravljamo na počitnice. Vozilo moramo vsako noč napolniti, zato smo si pripravili seznam krajev in cestnih povezav med njimi, ki jih lahko prevozimo v enem dnevu. Poiskati želimo pot od začetne točke do destinacije, ki bo imela čim manjše število postankov (tj., bo trajala čim manj dni).

(a) Predstavi problem v jeziku grafov in predlagaj algoritem za njegovo reševanje.

(b) Na koncu počitnic razmišljamo o poti nazaj. Spet bi radi naredili čim manj postankov, a se pri tem ne želimo ustaviti v nobenem kraju, kjer smo se ustavili na poti naprej. Dopolni zgornji algoritem, da bo našel še ustrezno pot nazaj.

(c) S pomočjo zgornjih algoritmov poišči najkrajšo pot od LJ do AM in najkrajšo pot nazaj v grafu s slike 14, ki ne gre čez kraje iz prejšnje poti.



Slika 14: Graf za nalogi 5.7 (brez uteži) in 5.10.

Naloga 5.8. Vir: Kolokvij OR 11.6.2018

Dan je povezan neusmerjen enostaven graf $G = (V, E)$ (tj., brez zank in večkratnih povezav). *Prerezno vozlišče* v grafu G je tako vozlišče $u \in V$, da graf $G - u$ (tj., graf G brez vozlišča u in povezav s krajiščem v u) ni več povezan. Poiskati želimo seznam prereznih vozlišč grafa G.

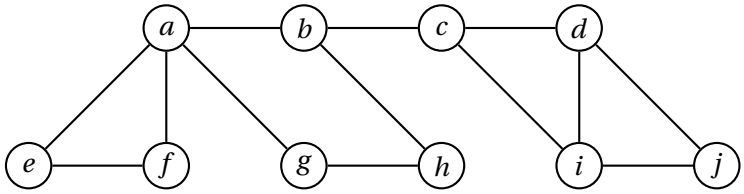
Pri iskanju si bomo pomagali s preiskovanjem v globino. Ob prvem obisku vozlišča u s predhodnikom v se tako pokliče funkcija PREVISIT(u, v), ob njegovem zadnjem obisku pa funkcija POSTVISIT(u, v). Če je u koren preiskovalnega drevesa, potem ima v vrednost NULL. Predpostavi, da imaš v obeh funkcijah dostop do seznama *izhod*, kamor bo treba dodati najdena presečna vozlišča. Prav tako imata lahko obe funkciji dostop do drugih pomožnih spremenljivk.

Naj bo ℓ_u globina vozlišča u v drevesu iskanja v globino (tj., razdalja od korena do u v drevesu iskanja v globino). Za vsako vozlišče u definiramo vrednost p_u kot najmanjšo globino vozlišč, ki so v grafu G sosedna (ali enaka) vozlišču u ali njegovim potomcem v drevesu iskanja v globino.

(a) Za graf na sliki 15 nariši drevo iskanja v globino (v njem označi tudi povratne povezave, npr. s črtkano črto) in določi njegova prerezna vozlišča. Upoštevaj abecedni vrstni red obiskovanja vozlišč. Za vsako vozlišče u določi še vrednosti ℓ_u in p_u.
Namig: vrednosti p_u najprej določi za vozlišča z večjo globino.

(b) Napiši rekurzivno formulo za vrednost p_u.
Namig: loči med sosedi v vozlišča u v grafu G (pišeš lahko $u \sim v$) in njegovimi neposrednimi nasledniki w v preiskovalnem drevesu ($u \to w$).

(c) Natančno opiši funkcijo PREVISIT(u, v) (z besedami ali psevdokodo), ki poskrbi za izračun vrednosti ℓ_u.



Slika 15: Graf za nalogo 5.8.

(d) Natančno opiši funkcijo POSTVISIT(u, v) (z besedami ali psevdokodo), ki naj za vozlišče u izračuna vrednost p_u in ugotovi, ali je u prerezno vozlišče, in ga v tem primeru doda v *izhod*. Predpostavi, da imaš globine vozlišč že poračunane.
Namig: obravnavaj dve možnosti – ko je u koren drevesa, in ko u ni koren drevesa. Kako v vsakem od teh primerov ugotoviš, ali je u prerezno vozlišče?

(e) Oceni časovno zahtevnost celotnega algoritma.

Naloga 5.9. Vir: Izpit OR 5.7.2018

Dan je utežen usmerjen acikličen graf s slike 16.

(a) Poišči topološko ureditev grafa s slike 16.

(b) Poišči najcenejšo pot od vozlišča s do vozlišča t v grafu s slike 16.

(c) Naj bo $G = (V, E)$ usmerjen acikličen graf z nenegativno uteženimi povezavami ter $s, t \in V$ njegovi vozlišči. Algoritem $\mathscr{A}$ se po grafu G sprehaja po naslednjem pravilu: začne v vozlišču s, v vsakem koraku pa se iz vozlišča u premakne v njegovega izhodnega soseda v z verjetnostjo

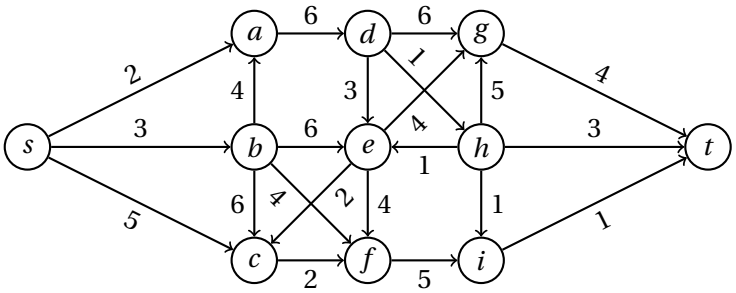
$$p_{uv} = \frac{\ell_{uv}}{\sum_{u \to w} \ell_{uw}},$$

kjer je ℓ_{uv} teža povezave od u do v. Algoritem $\mathscr{A}$ se ustavi, ko doseže vozlišče t.

Natančno opiši (z besedami ali psevdokodo), kako bi v času $O(m)$ (kjer je $m = |V| + |E|$) za vsako vozlišče $u \in V$ določil verjetnost q_u, da algoritem $\mathscr{A}$ obišče vozlišče u. Verjetnosti za graf s slike 16 ni potrebno računati.

Naloga 5.10. Vir: Izpit OR 28.8.2018

Odpravljamo se na pot, ki bo trajala več dni. Pripravili smo si seznam krajev in povezav med njimi, ki jih lahko prevozimo v enem dnevu. Za vsako povezavo poznamo stroške prevoza, prav tako pa za vsak kraj poznamo še stroške nočitev. Poiskati želimo čim cenejšo pot od začetne točke do destinacije (tj., skupna cena prevozov in nočitev naj bo čim manjša).



Slika 16: Graf za nalogo 5.9.

(a) Predstavi problem v jeziku grafov in predlagaj čim bolj učinkovit algoritem za njegovo reševanje.

(b) S pomočjo zgornjega algoritma poišči najcenejšo pot od LJ do BX v grafu s slike 14. Na povezavah so napisani stroški prevozov med krajema (veljajo za obe smeri), pri vozliščih pa stroški prenočitve v kraju.

Naloga 5.11. Vir: Izpit OR 29.8.2017

Peter zaključuje študij na Fakulteti za alternativno znanost. Opravil je že vse obvezne predmete, za pristop k zaključnemu izpitu pa mora opraviti še nekaj izbirnih predmetov. To bi rad storil čim hitreje. V tabeli 1 so našteti izbirni predmeti skupaj s trajanjem (v tednih, od pristopa do uspešnega opravljanja) in predmeti, h katerim lahko pristopi po uspešnem opravljanju. K predmetom a, c in f lahko pristopi že takoj, za pristop k ostalim predmetom pa zadostuje, če je opravljen eden od pogojev.

(a) Topološko uredi ustrezni graf in ga nariši.

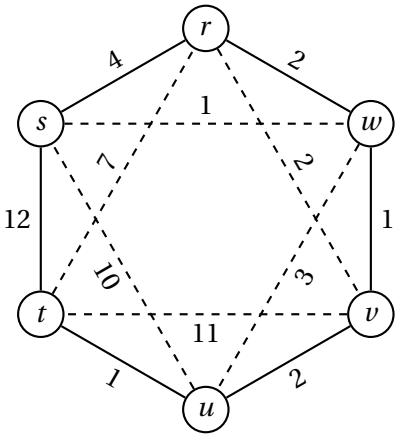
(b) Katere predmete naj Peter opravi, da bo lahko čim prej pristopil k zaključnemu izpitu? Koliko časa bo za to potreboval? Natančno opiši postopek iskanja odgovora.

Naloga 5.12. Vir: Kolokvij OR 3.6.2019

Dan je neusmerjen utežen graf $G = (V, E)$ z nenegativnimi cenami povezav L_e ($e \in E$). Naj bosta A in B disjunktni množici povezav, tako da velja $E = A \cup B$. Želimo najti najcenejšo *alternirajočo* pot med danima vozliščema $s, t \in V$ – torej takšno, v kateri se povezave iz A in iz B izmenjujejo (ni pomembno, ali začnemo oziroma končamo s povezavo iz množice A ali B). Posamezno vozlišče se lahko v alternirajoči poti pojavi tudi večkrat.

Oznaka	Predmet	Trajanje	Zadosten pogoj za
a	Alternativna zgodovina	5	i, j
b	Astrološki praktikum	7	d, g
c	Diskretna numerologija	9	b
d	Filozofija magije	4	z
e	Kvantno pravo	8	b, h
f	Postmoderna ekonomija	4	e
g	Telepatija in telekineza	5	z
h	Teorija antigravitacije	4	g
i	Teorije zarote	5	h
j	Ufologija II	10	k
k	Uvod v kriptozoologijo	6	z
z	Zaključni izpit	/	/

Tabela 1: Podatki za nalogo 5.11.



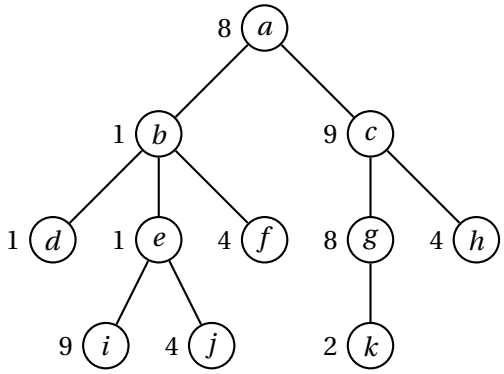
Slika 17: Graf za nalogo 5.12(b).

(a) Predlagaj čim učinkovitejši algoritem za reševanje danega problema. Kakšna je njegova časovna zahtevnost?
Namig: grafu G priredi usmerjen graf G', v katerem bodo vse poti od s do t ustrezale alternirajočim potem v G. Po potrebi lahko vozlišča tudi podvojiš.

(b) S svojim algoritmom poišči najcenejšo alternirajočo pot od s do t v grafu s slike 17. Povezave iz množice A so označene s polno, povezave iz množice B pa s črtkano črto. Iz rešitve naj bo jasno, kako poteka izvajanje algoritma.

Naloga 5.13. Vir: Kolokvij OR 3.6.2019

Neodvisna množica vozlišč grafa $G = (V, E)$ je taka množica $S \subseteq V$, da sta poljubni vozlišči iz množice S nesosedni v G, torej $uv \notin E$ za vsaka $u, v \in S$.



Slika 18: Drevo za nalogo 5.13(e).

Dano je drevo $T = (V, E)$ in uteži vozlišč c_v ($v \in V$). V drevesu T želimo najti *najtežjo neodvisno množico* – torej tako množico vozlišč $S \subseteq V$, ki maksimizira vsoto njihovih uteži, torej vrednost $\sum_{u \in S} c_u$.

(a) Denimo, da je drevo T podano kot neusmerjen graf, predstavljen s seznami sosedov. Razloži, kako lahko sestaviš slovar *pred*, ki za vsako vozlišče $v \in V$ določa njegovega prednika, če za koren izbereš vozlišče $r \in V$. Koren r je lahko izbran poljubno, zanj pa velja $pred[r] = \text{NULL}$. Kako iz seznamov sosedov in slovarja *pred* ugotovimo, katera vozlišča so neposredni nasledniki danega vozlišča v drevesu?

(b) Napiši rekurzivne enačbe za reševanje problema najtežje neodvisne množice v drevesu T. Razloži, kaj predstavljajo spremenljivke, v kakšnem vrstnem redu jih računamo, ter kako dobimo optimalno rešitev. Predpostaviš lahko, da imaš poleg seznamov sosedov in uteži vozlišč drevesa T na voljo tudi slovar *pred* kot v točki (a).
Namig: za vsako vozlišče uporabi dve spremenljivki – eno za primer, ko je vozlišče izbrano, in eno za primer, ko ni.

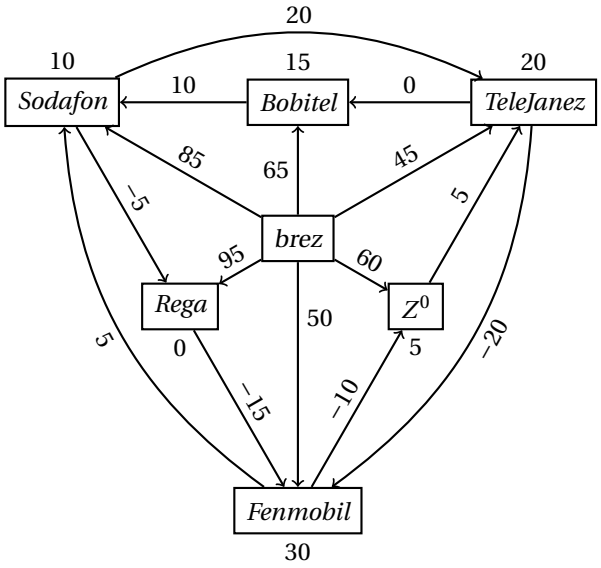
(c) Natančno opiši postopek (z besedami ali psevdokodo), ki iz zgoraj izračunanih vrednosti sestavi najtežjo neodvisno množico v T.

(d) Oceni časovno zahtevnost algoritma iz točk (b) in (c).

(e) S svojim algoritmom poišči najtežjo neodvisno množico na drevesu s slike 18. Iz rešitve naj bo jasno, kako poteka izvajanje algoritma.

Naloga 5.14. Vir: Izpit OR 19.6.2019

Trg mobilne telefonije je zelo konkurenčen, zato si mobilni operaterji na vse kriplje prizadevajo pridobiti stranke svojih konkurentov. Tako zelo, da lahko v nekaterih primerih stranka s prehodom h konkurentu celo zasluži. Da pa vendarle



Slika 19: Graf za nalogo 5.14.

operaterji sami ne bi imeli prevelike izgube, stranke zadržujejo z različnimi vezavami, poleg tega pa novim strankam (tistim, ki še niso imele mobilnega telefona) krepko zaračunajo priklop.

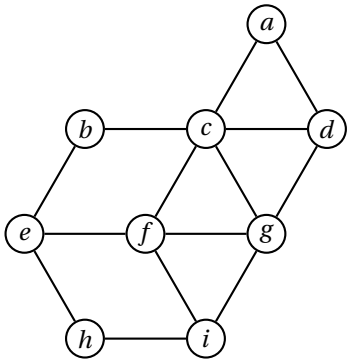
Stanje trga zberemo v utežen usmerjen graf $G = (V, E)$, kjer vozlišča predstavljajo operaterje, utež L_{uv} povezave $uv \in E$ pomeni strošek prehoda od operaterja u k operaterju v, utež c_v vozlišča $v \in V$ pa pomeni strošek, ki nastane tekom trajanja vezave pri operaterju v (ko preidemo k operaterju v, imamo torej c_v dodatnih stroškov). Če povezave uv ni v grafu, potem neposreden prehod od u k v ni mogoč. Uteži vozlišč so nenegativne, uteži povezav pa so lahko tudi negativne (tj., če ob prehodu zaslužimo). Poiskati želimo najcenejše zaporedje prehodov med operaterji (tj., tako, pri katerem imamo najmanj stroškov), če začnemo pri operaterju s in končamo pri operaterju t.

(a) Predlagaj čim bolj učinkovit algoritem za reševanje zgornjega problema. Kakšna je njegova časovna zahtevnost? Lahko predpostaviš, da velja $c_s = c_t = 0$.

(b) Denimo, da trenutno še nimamo mobilnega telefona. Naši izračuni kažejo, da je dolgoročno najugodnejša ponudba operaterja *Rega*, tako da bi radi s čim manjšimi stroški postali njihov naročnik. S pomočjo algoritma iz točke (a) poišči ustrezno zaporedje prehodov na grafu s slike 19.

Naloga 5.15. Vir: Izpit OR 4.6.2020

Dana sta neutežen neusmerjen graf $G = (V, E)$ z n vozlišči in m povezavami ter vozlišče $s \in V$. Za vsako vozlišče $v \in V$ nas zanima, koliko najkrajših poti od s do



Slika 20: Graf za nalogo 5.15.

v je v grafu G (tj., koliko je takšnih poti od s do v, katerih dolžina je enaka razdalji med s in v v grafu G).

(a) Čim bolj natančno opiši algoritem, ki reši zgornji problem v času $O(m)$.

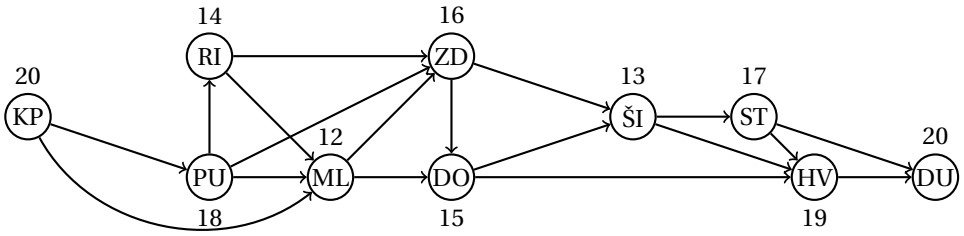
(b) Uporabi svoj algoritem, da za vsako vozlišče v grafu s slike 20 poiščeš število najkrajših poti od vozlišča i. Natančno zabeleži, kaj se zgodi v vsakem koraku algoritma.

Naloga 5.16. Vir: Izpit OR 22.6.2020

Večja skupina prijateljev želi preživeti počitnice na jadrnici. Ker pa je na sami jadrnici le manjše število ležišč, so se odločili, da bodo prenočevali na kopnem vzdolž poti (vključno z začetnim in končnim krajem). Sestavili so usmerjen aciklični graf $G = (V, E)$ z n vozlišči in m povezavami, v katerem vozlišča predstavljajo kraje, kjer se lahko ustavijo, vozlišči $u, v \in V$ pa sta povezani z usmerjeno povezavo $uv \in E$, če lahko v enem dnevu plujejo od kraja u do kraja v. Začetna in končna točka poti sta predstavljeni z vozliščema $s, t \in V$. Poleg tega za vsako vozlišče $u \in V$ poznajo še število k_u, ki predstavlja, koliko oseb lahko prespi v kraju u. Sestaviti želijo tako pot od s do t, da bo lahko na jadrnici potovalo čim več prijateljev (tj., maksimizirati želijo minimalno vrednost k_u vseh obiskanih krajev u na poti). Dolžina poti pri tem ni pomembna.

(a) Natančno opiši čim učinkovitejši algoritem (z besedami ali psevdokodo) za reševanje zgornjega problema in določi njegovo časovno zahtevnost.

(b) Uporabi svoj algoritem, da v grafu $G = (V, E)$ s slike 21 poiščeš tako pot od KP do DU, da bo lahko na jadrnici potovalo čim večje število prijateljev. Vrednosti k_u ($u \in V$) so zapisane ob vsakem vozlišču. Natančno zabeleži, kaj se zgodi v vsakem koraku algoritma.



Slika 21: Graf za nalogo 5.16(b).

Naloga 5.17. Vir: Vaje OR 17.5.2021

S pomočjo Floyd-Warshallovega algoritma poišči najkrajše poti med vsemi pari vozlišč v grafu s slike 11.

faza	opis	trajanje	pogoj	min. trajanje	cena za dan manj
a	gradnja kleti	10 dni	/	7 dni	200
b	gradnja pritličja	6 dni	a	5 dni	100
c	gradnja prvega nadstropja	7 dni	b, f	5 dni	150
d	gradnja strehe	8 dni	c, e	6 dni	160
e	gradnja desnega podpornega stebra	13 dni	a	9 dni	250
f	gradnja glavnega podpornega stebra	14 dni	/	11 dni	240
g	gradnja baročnega stolpa pred hišo	30 dni	/	25 dni	300

Tabela 2: Podatki za nalogi 6.1 (prvi štirje stolpci) in 6.2.

1.6 CPM/PERT

Naloga 6.1. Vir: Kolokvij OR 9.5.2013

Gradbinec in samooklicani arhitekt Brezzobec se je odločil, da bo postavil zelo posebno hišo. Gradnja bo imela sedem glavnih faz, ki so opisane v tabeli 2.

(a) Kdaj je lahko hiša najhitreje zgrajena? Katere faze so kritične?

(b) Koliko je kritičnih poti in katere so?

(c) Katero opravilo je najmanj kritično? Najmanj kritično je opravilo, katerega trajanje lahko največ podaljšamo, ne da bi vplivali na trajanje gradnje.

(d) Brezzobčev brat je ponudil pomoč pri največ eni fazi gradnje. Slovi po tem, da pri fazi, pri kateri pomaga, zmanjša čas izvajanja za 10%. Pri kateri fazi naj pomaga, da bo čas gradnje čim krajši?

Naloga 6.2. Vir: Vaje OR 28.5.2018

Brezzobčev bratranec ima podjetje, ki lahko pomaga pri gradnji, vendar za vsak dan krajšanja posamezne faze zahteva ustrezno plačilo (glej zadnja dva stolpca tabele 2). Brezzobca zanima način, kako bi s čim manjšimi stroški čas gradnje zmanjšal na 27 dni. Zapiši linearni program za ta problem.

Naloga 6.3. Vir: Izpit OR 19.5.2015

Dinamika priprave dveh palačink z dvema kuharjema je opisana v tabeli 3.

(a) Topološko uredi ustrezni graf in ga nariši.

(b) Določi kritična opravila in kritično pot ter trajanje priprave.

(c) Katero opravilo je najmanj kritično?

	aktivnost	trajanje	predhodna opravila
a	nakup moke, jajc in mleka	5 min	c
b	rezanje sira	3 min	/
c	vožnja do trgovine	5 min	/
d	čiščenje mešalnika	2 min	e
e	mešanje sestavin	5 min	a
f	pečenje prve palačinke	2 min	e
g	mazanje prve palačinke z marmelado	1 min	f, j
h	pečenje palačinke (s sirom)	3 min	b, f
i	pomivanje posode	8 min	g, h
j	odpiranje marmelade	1 min	/

Tabela 3: Podatki za naloge 6.3, 6.4 in 6.5.

Naloga 6.4. Vir: Vaje OR 18.5.2020

Določi skupne, proste, varnostne in neodvisne rezerve opravil iz naloge 6.3.

Naloga 6.5. Vir: Vaje OR 14.12.2016

Določi razpored opravil iz naloge 6.3, pri čemer en kuhar prevzame opravila na kritični poti, drugi pa naj čim kasneje začne in čim prej konča.

Naloga 6.6. Vir: Izpit OR 31.1.2017

Izdelati želimo terminski plan za izdelavo spletne aplikacije. V tabeli 4 so zbrana opravila pri izdelavi.

(a) Topološko uredi ustrezni graf in ga nariši.

(b) Določi kritična opravila in kritično pot ter čas izdelave.

(c) Katero opravilo je najmanj kritično? Najmanj kritično je opravilo, katerega trajanje lahko najbolj podaljšamo, ne da bi vplivali na trajanje izdelave.

Naloga 6.7. Vir: Kolokvij OR 11.6.2018

Izdelati želimo terminski plan za organizacijo konference. V tabeli 5 so zbrana opravila pri organizaciji.

(a) Topološko uredi ustrezni graf in ga nariši. Za trajanja opravil vzemi pričakovana trajanja po modelu PERT.

(b) Določi pričakovano kritično pot in čas izdelave.

(c) Katero opravilo je (ob zgornjih predpostavkah) najmanj kritično? Najmanj kritično je opravilo, katerega trajanje lahko najbolj podaljšamo, ne da bi vplivali na celotno trajanje izvedbe.

opravilo	opis	trajanje	pogoji
a	natančna opredelitev funkcionalnosti	15 dni	/
b	programiranje uporabniškega vmesnika	40 dni	k
c	programiranje skrbniškega vmesnika	25 dni	a, m
d	programiranje strežniškega dela	30 dni	a, m
e	integracija uporabniškega vmesnika s strežnikom	20 dni	b, d
f	alfa testiranje	20 dni	c, e
g	beta testiranje	30 dni	f, h
h	pridobivanje testnih uporabnikov	45 dni	a
i	vnos zadnjih popravkov	10 dni	g
j	izdelava uporabniške dokumentacije	35 dni	b
k	dizajniranje uporabniškega vmesnika	15 dni	a
ℓ	nabava računalniške opreme	20 dni	/
m	postavitev strežnikov	10 dni	ℓ

Tabela 4: Podatki za nalogo 6.6.

	Opravilo	Pogoji	Minimalno trajanje	Najbolj verjetno trajanje	Maksimalno trajanje
a	Izbira lokacije	/	10 dni	13 dni	22 dni
b	Rezervacija sob za goste	f	13 dni	22 dni	25 dni
c	Dogovarjanje za cene hotelskih sob	a	3 dni	6 dni	9 dni
d	Naročilo hrane in pijače	a	6 dni	15 dni	21 dni
e	Priprava letakov	c, j	5 dni	8 dni	11 dni
f	Pošiljanje letakov	e	4 dni	4 dni	4 dni
g	Priprava zbornika s povzetki	d, j	22 dni	28 dni	31 dni
h	Določitev glavnega govorca	/	5 dni	8 dni	14 dni
i	Planiranje poti za glavnega govorca	a, h	11 dni	14 dni	17 dni
j	Določitev ostalih govorcev	h	12 dni	15 dni	21 dni
k	Planiranje poti za ostale govorce	a, j	9 dni	12 dni	18 dni

Tabela 5: Podatki za nalogo 6.7.

(d) Določi variance trajanj opravil in oceni verjetnost, da bo izvedba trajala manj kot 55 dni.

Naloga 6.8. Vir: Izpit OR 28.8.2018

Pri izdelavi letala imamo faze, opisane v tabeli 6.

(a) S pomočjo topološke ureditve ustreznega grafa določi kritična opravila in čas izdelave. Uporabi podatke iz stolpca “trajanje”.

(b) Katero opravilo je najmanj kritično? Najmanj kritično jo opravilo, katerega trajanje lahko najbolj podaljšamo, ne da bi vplivali na trajanje izdelave.

opravilo	opis	trajanje	pogoji	najmanjše trajanje	cena za dan manj
a	izgradnja nosu	40 dni	/	36 dni	1 000
b	izgradnja trupa s krili	50 dni	/	48 dni	1 500
c	izgradnja repa	35 dni	/	31 dni	800
d	vgraditev pilotske kabine	30 dni	a	28 dni	1 200
e	opremljanje potniške kabine	18 dni	f, g	16 dni	500
f	povezovanje nosu s trupom	10 dni	a, b	9 dni	1 100
g	povezovanje repa s trupom	12 dni	b, c	10 dni	1 300
h	vgraditev motorjev	20 dni	b	18 dni	1 400

Tabela 6: Podatki za nalogo 6.8.

(c) Naročnik bi rad, da letalo izdelamo v 75 dneh. Posamezna opravila lahko skrajšamo tako, da zanje zadolžimo več delavcev. Seveda prinese tako krajšanje dodatne stroške, poleg tega pa za vsako opravilo poznamo najmanjše možno trajanje (glej zadnja dva stolpca v zgornji tabeli). Zapiši linearni program, s katerim modeliramo iskanje razporeda opravil, ki nam prinese čim manjše stroške. Linearnega programa ne rešuj.

Naloga 6.9. Vir: Kolokvij OR 3.6.2019

Izdelati želimo terminski plan za izgradnjo manjšega stanovanjskega naselja. V tabeli 7 so zbrana opravila pri gradnji.

(a) Topološko uredi ustrezni graf in ga nariši. Za trajanja opravil vzemi pričakovana trajanja po modelu PERT.

(b) Določi pričakovano kritično pot in čas izdelave.

(c) Katero opravilo je (ob zgornjih predpostavkah) najmanj kritično? Najmanj kritično je opravilo, katerega trajanje lahko najbolj podaljšamo, ne da bi vplivali na celotno trajanje izvedbe.

(d) Določi variance trajanj opravil in oceni verjetnost, da bo gradnja trajala manj kot 180 dni.

Naloga 6.10. Vir: Izpit OR 19.6.2019

Gradbeno podjetje gradi avtocestni odsek na težavnem terenu. Identificirali so faze gradnje, ki so zbrane v tabeli 8.

(a) S pomočjo topološke ureditve ustreznega grafa določi kritična opravila in čas izdelave. Uporabi podatke iz stolpca “trajanje”.

(b) Katero opravilo je najmanj kritično? Najmanj kritično jo opravilo, katerega trajanje lahko najbolj podaljšamo, ne da bi vplivali na trajanje izdelave.

	Opravilo	Pogoji	Minimalno trajanje	Najbolj verjetno trajanje	Maksimalno trajanje
a	pridobitev gradbenega dovoljenja	/	14 dni	14 dni	23 dni
b	pridobitev uporabnega dovoljenja	f, g	18 dni	21 dni	30 dni
c	izgradnja komunalne infrastrukture	a	40 dni	45 dni	59 dni
d	postavitev temeljev	a, i	28 dni	37 dni	46 dni
e	izgradnja podpornega zidu	a	7 dni	8 dni	12 dni
f	izgradnja severne stolpnice	c, d	63 dni	78 dni	99 dni
g	izgradnja južne stolpnice	d, e	75 dni	84 dni	99 dni
h	izgradnja otroškega igrišča	a, i	20 dni	29 dni	29 dni
i	odstranitev rastja	/	19 dni	22 dni	25 dni

Tabela 7: Podatki za nalogo 6.9.

opravilo	opis	trajanje	pogoji	najmanjše trajanje	cena za dan manj
a	izgradnja južnega priključka	45 dni	/	40 dni	300
b	izgradnja severnega priključka	20 dni	/	15 dni	200
c	vrtanje tunelske cevi	90 dni	a, b	70 dni	700
d	postavitev zahodnega stebra viadukta	25 dni	a	22 dni	1 100
e	postavitev vzhodnega stebra viadukta	30 dni	c	26 dni	1 500
f	gradnja cestišča na viaduktu	35 dni	d, e	28 dni	900
g	ureditev napeljave v tunelu	80 dni	c	65 dni	400
h	asfaltiranje viadukta	10 dni	f	8 dni	150

Tabela 8: Podatki za nalogo 6.10.

(c) Naročnik bi rad, da podjetje odsek zgradi v 200 dneh. Posamezna opravila lahko skrajšajo tako, da zanje zadolžijo več delavcev. Seveda prinese tako krajšanje dodatne stroške, poleg tega pa za vsako opravilo poznamo najmanjše možno trajanje (glej zadnja dva stolpca v tabeli 8). Zapiši linearni program, s katerim modeliramo iskanje razporeda opravil, ki nam prinese čim manjše stroške. Linearnega programa ne rešuj.

1.7 Upravljanje zalog

Naloga 7.1. Vir: Vaje OR 6.6.2018

Marta izdeluje nakit iz školjk v delavnici, ki jo najema v bližini ljubljanske tržnice, in ga prodaja po vnaprej dogovorjeni ceni. Povpraševanje je 10 kosov na teden, stroški skladiščenja so 0.2€ za kos na teden. Zagonski stroški izdelovanja nakita so 150€. Marta lahko na teden izdela 12.5 kosa nakita. Dovoli si, da pride do primanjkljaja, pri čemer jo ta stane 0.8€ za kos nakita na teden. Kako naj Marta organizira proizvodnjo in skladiščenje, da bo imela čim manj stroškov?

Naloga 7.2. Vir: Kolokvij OR 11.6.2018

Vulkanizer v svoji delavnici izdeluje in prodaja avtomobilske pnevmatike. Glede na trenutno povpraševanje vsak teden proda 60 pnevmatik, v istem času pa jih lahko izdela 90. Cena zagona proizvodnje je 180€, cena skladiščenja posamezne pnevmatike pa je 0.5€ na teden.

(a) Denimo, da primanjkljaja ne dovolimo. Izračunaj dolžino cikla proizvodnje in prodaje pnevmatik, pri kateri so stroški najmanjši. Kako veliko skladišče mora vulkanizer imeti? Izračunaj tudi enotske stroške.

(b) Kako dolgo naj pri zgornji rešitvi traja proizvodnja? Koliko pnevmatik naj vulkanizer naredi v vsakem ciklu?

(c) Vulkanizer je ocenil, da bi ga primanjkljaj ene pnevmatike stal 2€ na teden. Kakšna naj bosta dolžina cikla in velikost skladišča, da bodo stroški čim manjši? Izračunaj tudi enotske stroške in določi največji primanjkljaj.

Naloga 7.3. Vir: Izpit OR 5.7.2018

V trgovini s pohištvom vsak mesec prodajo 20 sedežnih garnitur. Z vsakim naročilom imajo 750€ stroškov, pri tem pa vse naročeno blago dobijo hkrati. Skladiščenje posamezne sedežne garniture jih stane 10€ na mesec, primanjkljaj za en kos pa jih stane 50€ na mesec.

(a) Kako pogosto naj v trgovini naročajo sedežne garniture, da bodo stroški čim manjši? Kako veliko skladišče morajo imeti? Izračunaj tudi enotske stroške.

(b) Izračunaj največji dovoljeni primanjkljaj. Koliko sedežnih garnitur naj vsakič naročijo?

(c) V trgovini dobijo ponudbo za uporabo drugega skladišča, v katerem imajo za posamezno sedežno garnituro le 6€ stroškov na mesec, vendar lahko hrani največ 40 sedežnih garnitur. Kako naj organizirajo naročanje, če namesto prvotnega uporabljajo to skladišče? Ali se jim ga splača uporabiti?
Namig: izpelji formulo za enotske stroške in poišči optimalen interval naročanja pri fiksni velikosti skladišča.

Naloga 7.4. Vir: Izpit OR 22.6.2020

V šiviljski delavnici so začeli proizvajati zaščitne maske. Vsak dan lahko izdelajo 400 mask, prodajo pa jih 300. Cena zagona proizvodnje je 240€, cena skladiščenja ene maske za en dan pa je 0.1€.

(a) Denimo, da primanjkljaj ni dovoljen. Kako pogosto naj zaženejo proizvodnjo in koliko časa naj ta teče? Koliko mask naj izdelajo v posameznem ciklu ter kako veliko skladišče potrebujejo?

(b) Obravnavajmo še primer, ko dovolimo primanjkljaj, ki nas stane 0.4€ na dan za eno masko. Izračunaj podatke iz prejšnje točke še za ta primer. Kolikšen je največji dovoljeni primanjklaj?

2 Rešitve

2.1 Zahtevnost algoritmov

Naloga 1.1.

(a) Program prepiše vnose v matriki A nad diagonalo na ustrezno mesto pod diagonalo tako, da je po izvedbi programa matrika A simetrična.

(b) Kot korak bomo upoštevali posamezno izvedbo notranje zanke **for**, kjer kopiramo vrednost v matriki na drugo mesto – ob predpostavki, da je velikost vnosov omejena (npr. 32-bitna cela števila), bo trajanje take operacije omejeno s konstanto. Preštejmo število takih korakov:

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} 1 = \sum_{i=1}^{n} (n-i) = \sum_{i=0}^{n-1} i = \frac{n(n-1)}{2}$$

Lahko bi seveda upoštevali še korake, ki so potrebni za vzdrževanje števcev zank (inicializacija števca, povečevanje števca, preverjanje konca zanke), a bi spet dobili kvadratni polinom v n. Tako lahko rečemo, da je število korakov omejeno z $O(n^2)$.

Naloga 1.2.

(a) V vsakem obhodu zanke **while** se vrednost $\ell[i]$ spremeni iz 0 v 1 ali obratno. Če se vrednost spremeni na 1, se i nastavi na 1, sicer se pa poveča za 1.

Izpišimo si vrednosti v seznamu ℓ in spremenljivke i ob koncu vsakega obhoda zanke **while** tekom izvajanja algoritma, recimo za $n = 4$:

obhod	$\ell[4\ldots1]$	i	obhod	$\ell[4\ldots1]$	i
1	0001	1	16	1001	1
2	0000	2	17	1000	2
3	0010	1	18	1010	1
4	0011	1	19	1011	1
5	0010	2	20	1010	2
6	0000	3	21	1000	3
7	0100	1	22	1100	1
8	0101	1	23	1101	1
9	0100	2	24	1100	2
10	0110	1	25	1110	1
11	0111	1	26	1111	1
12	0110	2	27	1110	2
13	0100	3	28	1100	3
14	0000	4	29	1000	4
15	1000	1	30	0000	5

Če pogledamo samo tiste obhode, na koncu katerih velja $i = 1$, opazimo, da vrednosti v seznamu ℓ predstavljajo dvojiške zapise števil od 1 do $2^n - 1$ v ostalih pa se najmanj pomembna 1 zamenja z 0. Algoritem torej simulira dvojiški števec z n mesti.

(b) Algoritem obišče vseh $2^n - 1$ števil, poleg tega pa mora vsakič poskrbeti za zamenjavo vseh enic za najmanj pomembno ničlo. Ob upoštevanju, da obstaja 2^{n-i} števil z najmanj pomembno ničlo na i-tem mestu, za vrednost $2^n - 1$ pa je potrebno nadomestiti vseh n mest, je skupno število korakov enako

$$n + \sum_{i=1}^{n} (i \cdot 2^{n-i}) = \sum_{j=1}^{n} \left(1 + \sum_{i=j}^{n} 2^{n-i}\right) =$$

$$= \sum_{j=1}^{n} \left(1 + \sum_{i=0}^{n-j} 2^{i}\right) = \sum_{j=1}^{n} 2^{n-j+1} = 2^{n+1} - 2.$$

Časovna zahtevnost algoritma je torej $O(2^n)$.

Naloga 1.3.

(a) Izpišimo vrednosti spremenljivk ob koncu vsakega obhoda zanke **for** oziroma **while**, ko se prejšnja konča.

obhod **while**	i	y	z	$\ell[1\ldots5]$
1	2	1	5	[7,11,16,7,5]
1	3	1	5	[7,11,16,7,5]
1	4	4	5	[7,11,7,16,5]
1	5	5	5	[7,11,7,5,16]
1		5	4	[7,11,7,5,16]
2	2	1	4	[7,11,7,5,16]
2	3	3	4	[7,7,11,5,16]
2	4	4	4	[7,7,5,11,16]
2		4	3	[7,7,5,11,16]
3	2	1	3	[7,7,5,11,16]
3	3	3	3	[7,5,7,11,16]
3		3	2	[7,5,7,11,16]
4	2	2	2	[5,7,7,11,16]
4		2	1	[5,7,7,11,16]

(b) Naj bodo $z_1, z_2, \ldots, z_k$ vrednosti, ki jih zavzame spremenljivka z ob vsakem vstopu v zanko **while**. Očitno velja $z_i - 1 \geq z_{i+1}$ za vsak i, tako da velja $k \leq n-1$. Največje število korakov je torej

$$\sum_{z=2}^{n} (z-1) = \frac{n(n-1)}{2}.$$

Tako število korakov dosežemo, če je seznam ℓ na začetku urejen padajoče – tako vsakič pride do zamenjave v zadnjem koraku zanke **for**, zato se z vsakič zmanjša za 1. Časovna zahtevnost algoritma je torej $O(n^2)$.

Naloga 1.4.

(a)
```
function MergeSort(ℓ[1...n])
    if n ≤ 1 then
        return ℓ
    end if
    m ← ⌈n/2⌉
    ℓ1 ← MergeSort(ℓ[1...m])
    ℓ2 ← MergeSort(ℓ[m+1...n])
    i, j ← 1, 1
    ℓ' ← []
    while i ≤ m ∧ j ≤ n − m do
        if ℓ1[i] ≤ ℓ2[j] then
            ℓ'.append(ℓ1[i])
            i ← i + 1
        else
            ℓ'.append(ℓ2[j])
            j ← j + 1
        end if
    end while
    pripni ℓ1[i...m] na konec ℓ'
    pripni ℓ2[j...n − m] na konec ℓ'
    return ℓ'
end function
```

(b) Izvajanje algoritma je prikazano na sliki 22. Nad črtkano črto je prikazano rekurzivno razbijanje seznamov, pod njo pa zlivanje dobljenih urejenih podseznamov.

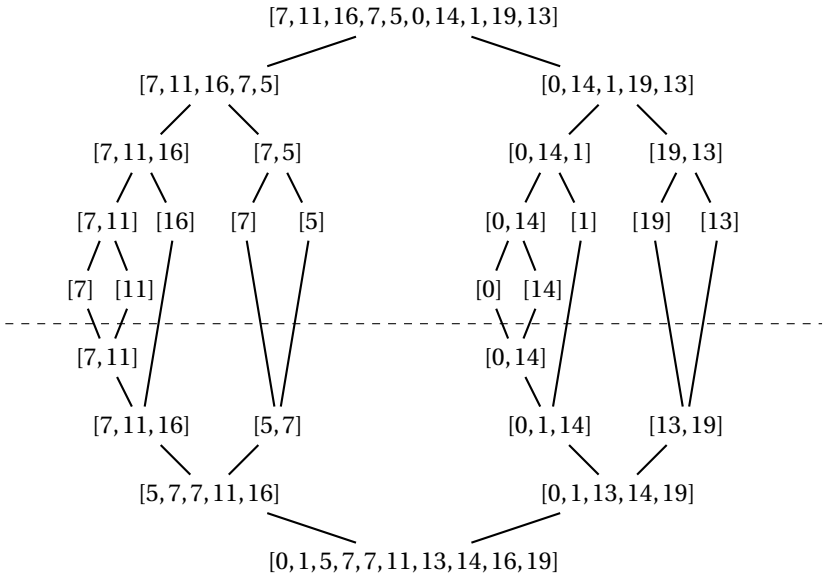
(c) Funkcija obsega dva rekurzivna klica na seznamih polovične dolžine ter združevanje obeh dobljenih seznamov v enega. Naj bo $T(n)$ čas izvajanja algoritma pri vhodnem seznamu dolžine n. Ker združevanje poteka v linearnem času, velja rekurzivna zveza

$$T(n) = O(n) + 2T\left(\frac{n}{2}\right).$$

Po krovnem izreku lahko izpeljemo, da je časovna zahtevnost algoritma $O(n \log n)$.

Naloga 1.5.

Algoritem teče v času $O(\sqrt{n})$. Ker je vhod algoritma število n, ki je zapisano kot zaporedje $\ell = O(\log n)$ bitov, vidimo, da algoritem teče v času $O(2^{\ell/2})$ in torej ni polinomski v dolžini vhoda.



Slika 22: Diagram izvajanja algoritma za nalogo 1.4(b).

Naloga 1.6.
Po krovnem izreku ima časovno zahtevnost $O(\sqrt{n})$ algoritem, katerega čas izvajanja $T(n)$ je opisan z rekurzivno zvezo

$$T(n) = O(1) + 2T\left(\frac{n}{4}\right).$$

Zapišimo tak algoritem:

function KORENSKI($\ell[1\ldots n]$)
  if $n \geq 4$ **then**
    $m \leftarrow \lfloor \frac{n}{4} \rfloor$
    KORENSKI($\ell[1\ldots m]$)
    KORENSKI($\ell[n-m+1\ldots n]$)
  end if
end function

2.2 Celoštevilsko linearno programiranje

Naloga 2.1.
Naj bo $G = (V, E)$ dvodelen graf. Lahko torej zapišemo $V = A \cup B$ tako, da sta množici A in B disjunktni ter za vsako povezavo $uv \in E$ velja $u \in A$, $v \in B$. Množica $M \subset E$ je *prirejanje*, če nobeni dve povezavi iz M nimata skupnega krajišča. Iščemo največje prirejanje $M_{\max}$.

Za vsako povezavo $uv \in E$ bomo uvedli spremenljivko x_{uv}, katere vrednost interpretiramo kot

$$x_{uv} = \begin{cases} 1, & \text{povezava } uv \text{ je v množici } M_{\max}, \text{ in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\begin{aligned} & \max \sum_{uv \in E} x_{uv} \quad \text{p.p.} \\ \forall uv \in E: \ & 0 \le x_{uv} \le 1, \quad x_{uv} \in \mathbb{Z} \\ \forall u \in A: \ & \sum_{uv \in E} x_{uv} \le 1 \\ \forall v \in B: \ & \sum_{uv \in E} x_{uv} \le 1 \end{aligned}$$

Naloga 2.2.
Za i-tega asistenta ($1 \le i \le n$) in j-ti predmet ($1 \le j \le m$) bomo uvedli spremenljivki x_{ij} in y_{ij}, kjer je x_{ij} število skupin, ki jih i-temu asistentu dodelimo pri j-tem predmetu, vrednost y_{ij} pa interpretiramo kot

$$y_{ij} = \begin{cases} 1, & i\text{-ti asistent bo izvajal vaje pri } j\text{-tem predmetu, in} \\ 0 & \text{sicer.} \end{cases}$$

Poleg tega bomo uvedli še spremenljivko t, s katero omejimo največje število različnih predmetov pri posameznem asistentu.

Zapišimo celoštevilski linearni program.

$$\begin{aligned} & \min t \quad \text{p.p.} \\ \forall i \in \{1,\dots,n\}\ \forall j \in \{1,\dots,m\}: \ & x_{ij} \ge 0, \quad x_{ij} \in \mathbb{Z} \\ \forall i \in \{1,\dots,n\}\ \forall j \in \{1,\dots,m\}: \ & 0 \le y_{ij} \le 1, \quad y_{ij} \in \mathbb{Z} \end{aligned}$$

$x_{ij} = 0$ natanko tedaj, ko $y_{ij} = 0$:

$$\forall i \in \{1,\dots,n\}\ \forall j \in \{1,\dots,m\}: \quad y_{ij} \le x_{ij} \le c_j y_{ij}$$

Omejimo število ur za vsakega asistenta:

$$\forall i \in \{1,\dots,n\}: a_i \le \sum_{j=1}^{m} u_j x_{ij} \le b_i$$

Neželenih predmetov ne dodelimo:

$$\forall i \in \{1,\dots,n\}: \qquad \sum_{j \in N_i} y_{ij} = 0$$

Za vsak predmet dodelimo potrebno število skupin:

$$\forall j \in \{1,\dots,m\}: \qquad \sum_{i=1}^{n} x_{ij} = c_j$$

Nobenega predmeta ne dodelimo asistentoma p in q:

$$\forall j \in \{1,\dots,m\}: \qquad y_{pj} + y_{qj} \leq 1$$

Število različnih predmetov na asistenta omejimo s t:

$$\forall i \in \{1,\dots,n\}: \qquad \sum_{j=1}^{n} y_{ij} \leq t$$

Naloga 2.3.
Za stroj i $(1 \leq i \leq n)$ bomo uvedli spremenljivko x_i, dodatno za opravilo j in časovno enoto h $(1 \leq h, j \leq m)$ pa še spremenljivko y_{ijh}. Njihove vrednosti interpretiramo kot

$$x_i = \begin{cases} 1, & \text{če stroj } i \text{ izvede vsaj eno opravilo, in} \\ 0 & \text{sicer;} \end{cases}$$

$$\text{ter} \quad y_{ijh} = \begin{cases} 1, & \text{če stroj } i \text{ izvede opravilo } j \text{ v časovni enoti } h, \text{ in} \\ 0 & \text{sicer.} \end{cases}$$

Dodatno bomo uvedli še spremenljivko t, katere vrednost bo enaka časovni enoti dokončanja zadnjega stroja.

Zapišimo celoštevilski linearni program.

$$\min t \qquad \text{p.p.}$$

$$\forall i \in \{1,\dots,n\}: \qquad 0 \leq x_i \leq 1, \qquad x_i \in \mathbb{Z}$$

$$\forall i \in \{1,\dots,n\}\ \forall j,h \in \{1,\dots,m\}: \qquad 0 \leq y_{ijh} \leq 1, \qquad y_{ijh} \in \mathbb{Z}$$

Vsako opravilo se izvede natanko enkrat:

$$\forall j \in \{1,\dots,m\}: \qquad \sum_{i=1}^{n} \sum_{h=1}^{m} y_{ijh} = 1$$

Vsako stroj lahko hkrati opravlja največ eno opravilo:

$$\forall i \in \{1,\dots,n\}\ \forall h \in \{1,\dots,m\}: \qquad \sum_{j=1}^{m} y_{ijh} \leq 1$$

Omejitev stroškov obratovanja:

$$\forall i \in \{1,\dots,n\}: \qquad \sum_{j=1}^{m} \sum_{h=1}^{m} y_{ijh} \leq m x_i$$

$$\sum_{i=1}^{n} c_i x_i \leq C$$

Vrstni red opravil:

$$\forall (j,k) \in P : \sum_{i=1}^{n} \sum_{h=1}^{m} h(y_{ikh} - y_{ijh}) \geq 1$$

Hkratnost opravil:

$$\forall (j,k) \in S \; \forall h \in \{1,\ldots,m\} : \qquad \sum_{i=1}^{n} (y_{ijh} + y_{ikh}) \leq 1$$

Omejitev hkrati aktivnih strojev:

$$\forall h \in \{1,\ldots,m\} : \qquad \sum_{i=1}^{n} \sum_{j=1}^{m} y_{ijh} \leq A$$

Omejitev časa dokončanja:

$$\forall i \in \{1,\ldots,n\} \; \forall j,h \in \{1,\ldots,m\} : \qquad h y_{ijh} \leq t$$

Naloga 2.4.
Za vsak strežnik h ($0 \leq h \leq k$), ponudnika i ($1 \leq i \leq m$) in posnetek j ($1 \leq j \leq n$) bomo uvedli spremenljivko x_{hij}, za strežnik h ($1 \leq h \leq k$) in posnetek j ($1 \leq j \leq n$) pa dodatno še spremenljivko y_{hj}. Njihove vrednosti interpretiramo kot

$$x_{hij} = \begin{cases} 1, & \text{ponudniku } i \text{ pošiljamo posnetek } j \text{ iz strežnika } h, \text{ in} \\ 0 & \text{sicer;} \end{cases}$$

$$\text{ter} \quad y_{hj} = \begin{cases} 1, & \text{posnetek } j \text{ naložimo na strežnik } h, \text{ in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\min \sum_{h=0}^{k} \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \ell_{hi} x_{hij} \quad \text{p.p.}$$

$$\forall h \in \{0,\ldots,k\} \; \forall i \in \{1,\ldots,m\} \; \forall j \in \{1,\ldots,n\} : \qquad 0 \leq x_{hij} \leq 1, \quad x_{hij} \in \mathbb{Z}$$

$$\forall h \in \{1,\ldots,k\} \; \forall j \in \{1,\ldots,n\} : \qquad 0 \leq y_{hj} \leq 1, \quad y_{hj} \in \mathbb{Z}$$

Omejitev prostora:

$$\forall h \in \{1,\ldots,k\} : \qquad \sum_{j=1}^{n} s_j y_{hj} \leq c_h$$

Za dostavo mora biti posnetek prisoten na strežniku:

$$\forall h \in \{0,\ldots,k\} \; \forall i \in \{1,\ldots,m\} \; \forall j \in \{1,\ldots,n\} : \qquad x_{hij} \leq y_{hj}$$

Strežnika za posnetek ne uporabimo, če se nahaja na strežniku z manjšo latenco:

$$\forall h,t \in \{0,\ldots,k\} \; \forall i \in \{1,\ldots,m\} \; \forall j \in \{1,\ldots,n\} : \qquad (\ell_{hi} - \ell_{ti})(x_{hij} + y_{tj}) \leq 1$$

Vsak posnetek pošiljamo ponudniku iz natanko enega strežnika:

$$\forall h \in \{0,\ldots,k\} \; \forall j \in \{1,\ldots,n\} : \qquad \sum_{i=1}^{m} x_{hij} = 1$$

Naloga 2.5.
Za i-tega trgovca ($1 \le i \le n$) bomo uvedli spremenljivke x_i, y_i in z_i, pri čemer sta x_i in y_i število zabojev avokadov oziroma banan, ki jih distributer proda i-temu trgovcu, in z_i število voženj tovornjakov, ki bodo sadje dostavili do i-tega trgovca. Poleg tega bomo uvedli še spremenljivki u in v, ki ju interpretiramo kot

$$u = \begin{cases} 1, & \text{trgovec } p \text{ kupi avokade, in} \\ 0 & \text{sicer;} \end{cases}$$

$$\text{ter} \quad v = \begin{cases} 1, & \text{trgovec } p \text{ kupi banane, in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\begin{aligned} \max \sum_{i=1}^{n} (a_i x_i + b_i y_i - d_i z_i) \qquad & \text{p.p.} \\ \forall i \in \{1,\dots,n\}: \qquad & x_i \ge 0, \quad x_i \in \mathbb{Z} \\ \forall i \in \{1,\dots,n\}: \qquad & y_i \ge 0, \quad y_i \in \mathbb{Z} \\ \forall i \in \{1,\dots,n\}: \qquad & z_i \ge 0, \quad z_i \in \mathbb{Z} \\ & 0 \le u \le 1, \quad u \in \mathbb{Z} \\ & 0 \le v \le 1, \quad v \in \mathbb{Z} \end{aligned}$$

Skupno število zabojev:

$$\sum_{i=1}^{n} x_i \le A$$
$$\sum_{i=1}^{n} y_i \le B$$

Omejitev porabe trgovca:

$$\forall i \in \{1,\dots,n\}: a_i x_i + b_i y_i \le c_i$$

Število tovornjakov:

$$\forall i \in \{1,\dots,n\}: \qquad x_i + y_i \le K z_i$$

Trgovec p ne kupi obojega:

$$x_p \le Au$$
$$y_p \le Bv$$
$$u + v \le 1$$

Trgovec q kupi avokade, če jih tudi r:

$$x_r \le A x_q$$

Naloga 2.6.
Očitno bomo potrebovali največ n delavcev, zato bomo za h-tega delavca ($1 \le h \le n$), i-to nalogo ($1 \le i \le n$) in k-to časovno enoto ($1 \le k \le T$) uvedli spremenljivko x_{hik}, katere vrednost interpretiramo kot

$$x_{hik} = \begin{cases} 1; & h\text{-ti delavec začne izvajati } i\text{-to nalogo v } k\text{-ti časovni enoti, in} \\ 0 & \text{sicer.} \end{cases}$$

Poleg tega bomo uvedli še spremenljivko r, ki šteje število delavcev.

Zapišimo celoštevilski linearni program, pri čemer uporabimo oznako $T_i = T - t_i + 1$ za zadnji možen čas začetka i-te naloge.

$$\begin{aligned} & & \min r & & \text{p.p.} \\ \forall h,i \in \{1,\dots,n\}\ \forall k \in \{1,\dots,T\}: & & 0 \le x_{hik} \le 1, & & x_{hik} \in \mathbb{Z} \\ & & r \ge 0, & & r \in \mathbb{Z} \end{aligned}$$

Vsako nalogo opravi natanko en delavec in jo pravočasno konča:

$$\forall i \in \{1,\dots,n\}: \qquad \sum_{h=1}^{n} \sum_{k=1}^{T_i} x_{hik} = 1$$

Vsak delavec lahko hkrati izvaja samo eno nalogo:

$$\forall h,i \in \{1,\dots,n\}\ \forall k \in \{1,\dots,T_i\}: (t_i - 2)x_{hik} + \sum_{j=1}^{n} \sum_{\ell=k}^{k+t_i-1} x_{hj\ell} \le t_i - 1$$

Vrstni red opravljanja nalog:

$$\forall (i,j) \in S\ \forall k \in \{1,\dots,T_i\}: \qquad \sum_{h=1}^{n} \left(x_{hik} - \sum_{\ell=k+t_i}^{T_j} x_{hj\ell} \right) \le 0$$

Štetje delavcev:

$$\forall h,i \in \{1,\dots,n\}\ \forall k \in \{1,\dots,T\}: \qquad h x_{hik} \le r$$

Naloga 2.7.
Naj bo m število vozlišč grafa G. Očitno bo linija obiskala največ m postaj, zato bomo za postajališče $u \in V$ in $i \in \{1,\dots,m\}$ uvedli spremenljivko x_{ui}, za vsako povezavo $uv \in E$ pa še spremenljivko y_{uv}. Njihove vrednosti interpretiramo kot

$$x_{ui} = \begin{cases} 1; & \text{postajališče } u \text{ obiščemo } i\text{-to po vrsti, in} \\ 0 & \text{sicer;} \end{cases}$$

$$\text{ter} \quad y_{uv} = \begin{cases} 1; & \text{postajališči } u \text{ in } v \text{ obiščemo zaporedoma, in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\max \sum_{u\in V}\sum_{i=1}^{n} x_{ui} \qquad \text{p.p.}$$

$$\forall u \in V\ \forall i \in \{1,\dots,m\}: \qquad 0 \le x_{ui} \le 1, \qquad x_{ui} = \mathbb{Z}$$

$$\forall uv \in E: \qquad 0 \le y_{uv} \le 1, \qquad y_{uv} = \mathbb{Z}$$

Vsako postajališče obiščemo največ enkrat:

$$\forall u \in V: \qquad \sum_{i=1}^{m} x_{ui} \le 1$$

Naenkrat obiščemo največ eno postajališče:

$$\forall i \in \{1,\dots,m\}: \qquad \sum_{u\in V} x_{ui} \le 1$$

Indeksov ne izpuščamo:

$$\forall i \in \{2,\dots,m\}: \qquad \sum_{u\in V} x_{ui} \le \sum_{u\in V} x_{u,i-1}$$

Nesosedna postajališča niso zaporedna:

$$\forall uv \notin E\ \forall i \in \{2,\dots,m\}: \quad x_{u,i-1} + x_{ui} + x_{v,i-1} + x_{vi} \le 1$$

Začnemo v postajališču p_1:

$$x_{p_1,1} = 1$$

Postajališča iz seznama si sledijo v podanem vrstnem redu:

$$\forall i \in \{1,\dots,m\}\ \forall j \in \{2,\dots,n\}: \qquad x_{p_{j-1},i} \le \sum_{h=i+1}^{m} x_{p_j,h}$$

Končamo v postajališču p_n:

$$\forall i \in \{1,\dots,m\}: (m-i)x_{p_n,i} + \sum_{u\in V}\sum_{h=i+1}^{m} x_{uh} \le m-i$$

Časovna omejitev:

$$\forall uv \in E\ \forall i \in \{2,\dots,m\}: \quad x_{u,i-1} + x_{ui} + x_{v,i-1} + x_{vi} \le y_{uv} + 1$$

$$\sum_{uv\in E} c_{uv} y_{uv} \le T$$

Naloga 2.8.
Za igralca $i \in A$ in j-ti klub ($1 \le j \le n$) bomo uvedli spremenljivko x_{ij}, za igralca $i \in B$ pa spremenljivko y_i. Njihove vrednosti interpretiramo kot

$$x_{ij} = \begin{cases} 1; & \text{igralca } i \text{ prodamo } j\text{-temu klubu, in} \\ 0 & \text{sicer;} \end{cases}$$

$$\text{ter} \quad y_i = \begin{cases} 1; & \text{odkupimo ogralca } i \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\max \quad \sum_{i\in B} q_i y_i - \sum_{i\in A}\sum_{j=1}^{n} q_i x_{ij} \qquad \text{p.p.}$$

$$\forall i \in A\ \forall j \in \{1,\dots,n\}: \qquad 0 \le x_{ij} \le 1, \qquad x_{ij} \in \mathbb{Z}$$

$$\forall i \in B: \qquad 0 \le y_i \le 1, \qquad y \in \mathbb{Z}$$

Vsakega igralca prodamo največ enemu klubu:

$$\forall i \in A: \qquad \sum_{j=1}^{n} x_{ij} \le 1$$

Omejitev stroškov:

$$\sum_{i\in B} r_i y_i - \sum_{i\in A}\sum_{j=1}^{n} p_{ij} x_{ij} \le S$$

Skupno število igralcev:

$$\sum_{i\in B} y_i - \sum_{i\in A}\sum_{j=1}^{n} x_{ij} = 0$$

Število igralcev za vsako pozicijo:

$$-1 \le \sum_{i\in B\cap G} y_i - \sum_{i\in A\cap G}\sum_{j=1}^{n} x_{ij} \le 1$$

$$-1 \le \sum_{i\in B\cap D} y_i - \sum_{i\in A\cap D}\sum_{j=1}^{n} x_{ij} \le 1$$

$$-1 \le \sum_{i\in B\cap M} y_i - \sum_{i\in A\cap M}\sum_{j=1}^{n} x_{ij} \le 1$$

$$-1 \le \sum_{i\in B\cap F} y_i - \sum_{i\in A\cap F}\sum_{j=1}^{n} x_{ij} \le 1$$

Igralca a in b gresta v isti klub:

$$\forall j \in \{1,\dots,n\}: \qquad x_{aj} = x_{bj}$$

Naloga 2.9.

(a) Naj bo x_i število žetonov, ki jih vložimo v dvoboj z i-tim nasprotnikom ($1 \le i \le n$). Poleg tega bomo uvedli še spremenljivke y_i in z_i ($1 \le i \le n$), katerih vrednosti interpretiramo kot

$$y_i = \begin{cases} 1, & \text{če ne izgubimo proti } i\text{-temu nasprotniku, in} \\ 0 & \text{sicer;} \end{cases}$$

ter

$$z_i = \begin{cases} 1, & \text{če premagamo } i\text{-tega nasprotnika, in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\max \sum_{i=1}^{n} (y_i + 2z_i) \quad \text{p.p.}$$
$$\forall i \in \{1,\dots,n\}: \quad x_i \geq 0, \quad x_i \in \mathbb{Z}$$
$$\forall i \in \{1,\dots,n\}: \quad 0 \leq y_i \leq 1, \quad y_i \in \mathbb{Z}$$
$$\forall i \in \{1,\dots,n\}: \quad 0 \leq z_i \leq 1, \quad z_i \in \mathbb{Z}$$

Porabimo M žetonov:

$$\sum_{i=1}^{n} x_i = M$$

$y_i = 1$ natanko tedaj, ko $x_i \geq c_i$:

$$\forall i \in \{1,\dots,n\}: \quad c_i y_i \leq x_i \leq c_i - 1 + (M+1) y_i$$

$z_i = 1$ natanko tedaj, ko $x_i > c_i$:

$$\forall i \in \{1,\dots,n\}: (c_i + 1) z_i \leq x_i \leq c_i + M z_i$$

(b) Zapišimo še dodatne omejitve.

Proti nasprotnikom iz A ne igramo izenačeno:

$$\forall i \in A: \quad y_i \leq z_i$$

Največ dva poraza proti nasprotnikom iz B:

$$\sum_{i \in B} y_i \geq |B| - 2$$

Če izgubimo proti u, dosežemo vsaj 4 točke proti v in w:

$$4y_u + y_v + y_w + 2z_v + 2z_w \geq 4$$

Največ k dvobojev z več kot t žetoni:

$$\forall i \in \{1,\dots,n\}: \quad 0 \leq r_i \leq 1, \quad r_i \in \mathbb{Z}$$
$$\forall i \in \{1,\dots,n\}: \quad x_i \leq t + M r_i$$
$$\sum_{i=1}^{n} r_i \leq k$$

Vsaj ℓ porazov:

$$\sum_{i=1}^{n} y_i \leq n - \ell$$

Naloga 2.10.
Za i-to kolekcijo ($1 \le i \le n$) bomo uvedli spremenljivki x_i in y_i, kjer je x_i število kupljenih izvodov i-te kolekcije, vrednost y_i pa interpretiramo kot

$$y_i = \begin{cases} 1, & \text{kupimo vsaj en izvod } i\text{-te kolekcije, in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\max \sum_{i=1}^{n} ((d_i - c_i)x_i + e_i y_i) \quad \text{p.p.}$$

Omejitev števila izvodov:

$$\forall i \in \{1,\dots,n\}: \qquad 0 \le x_i \le k_i, \qquad x_i \in \mathbb{Z}$$
$$\forall i \in \{1,\dots,n\}: \qquad 0 \le y_i \le 1, \qquad y_i \in \mathbb{Z}$$

Povezava med x_i in y_i:

$$\forall i \in \{1,\dots,n\}: \qquad y_i \le x_i \le k_i y_i$$

Omejitev kapitala za nakup:

$$\sum_{i=1}^{n} c_i x_i \le C$$

Omejitev števila proizvajalcev:

$$\sum_{i=1}^{n} y_i \le K$$

Omejitev proizvajalca p:

$$x_q + x_r - x_p \le (k_q + k_r)(1 - y_p)$$

Naloga 2.11.
Za h-ti dan ($1 \le h \le m$), i-to skupino ($1 \le i \le n$) in j-ti termin ($1 \le j \le k$) bomo uvedli spremenljivko x_{hij}, katere vrednost interpretiramo kot

$$x_{hij} = \begin{cases} 1, & i\text{-ta skupina nastopi } j\text{-ta v } h\text{-tem dnevu festivala, in} \\ 0 & \text{sicer.} \end{cases}$$

Zapišimo celoštevilski linearni program.

$$\max \sum_{h=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x_{hij} \quad \text{p.p.}$$
$$\forall h \in \{1,\dots,m\}\ \forall i \in \{1,\dots,n\}\ \forall j \in \{1,\dots,j\}: \qquad 0 \le x_{hij} \le 1, \quad x_{hij} \in \mathbb{Z}$$

Vsaka skupina nastopi največ enkrat:

$$\forall i \in \{1,\ldots,n\}: \\ \sum_{h=1}^{m}\sum_{j=1}^{k} x_{hij} \leq 1$$

Naenkrat nastopa ena skupina:

$$\forall h \in \{1,\ldots,m\}\ \forall j \in \{1,\ldots,k\}: \\ \sum_{i=1}^{n} x_{hij} = 1$$

Omejitev plačila:

$$\sum_{h=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{k} m_i x_{hij} \leq M$$

Vrstni red v dnevu:

$$\forall h \in \{1,\ldots,m\}\ \forall j \in \{2,\ldots,k\}: \\ \sum_{i=1}^{n} m_i (x_{hij} - x_{h,i,j-1}) \geq 0$$

Razporeditev po dnevih:

$$\forall h \in \{2,\ldots,n\}: \\ \sum_{i=1}^{n}\sum_{j=1}^{k} a_{ij}(x_{hij} - x_{h-1,i,j}) \geq 0$$

r ne nastopa, če nastopa s:

$$\sum_{h=1}^{m}\sum_{j=1}^{k} (x_{hrj} + x_{hsj}) \leq 1$$

t ne nastopa, če ni zadnja ali pred u:

$$\forall h \in \{1,\ldots,m\}: \\ \sum_{j=1}^{k-1} x_{htj} \leq \sum_{j=2}^{k} j x_{huj} - \sum_{j=1}^{k-1} j x_{htj} \leq k - (k-1)\sum_{j=1}^{k-1} x_{htj}$$

v in w ne nastopata na isti dan:

$$\forall h \in \{1,\ldots,m\}: \\ \sum_{j=1}^{k} (x_{hvj} + x_{hwj}) \leq 1$$

Naloga 2.12.
Za h-ti center ($1 \leq h \leq m$) ter i-tega in j-tega strokovnjaka ($1 \leq i, j \leq n$) bomo uvedli spremenljivko x_{hij}, katere vrednost interpretiramo kot

$$x_{hij} = \begin{cases} 1, & i\text{-ti in } j\text{-ti strokovnjak sta oba nastanjena} \\ & \text{v } h\text{-tem karantenskem centru, in} \\ 0 & \text{sicer.} \end{cases}$$

Informacija o tem, ali je i-ti strokovnjak nastanjen v h-tem centru, hrani spremenljivka x_{hii} ($1 \le h \le m$, $1 \le i \le n$).

Zapišimo celoštevilski linearni program.

$$\min \sum_{h=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} x_{hij} \qquad \text{p.p.}$$

$$\forall h \in \{1,\dots,m\}\ \forall i,j \in \{1,\dots,n\}: \qquad 0 \le x_{hij} \le 1, \qquad x_{hij} \in \mathbb{Z}$$

Vsak strokovnjak je nastanjen v natanko enem centru:

$$\forall i \in \{1,\dots,n\}: \qquad \sum_{h=1}^{m} x_{hii} = 1$$

Kapacitete centrov:

$$\forall h \in \{1,\dots,m\}: \qquad \sum_{i=1}^{n} x_{hii} \le k_h$$

Povezava med x_{hii} in x_{hij}:

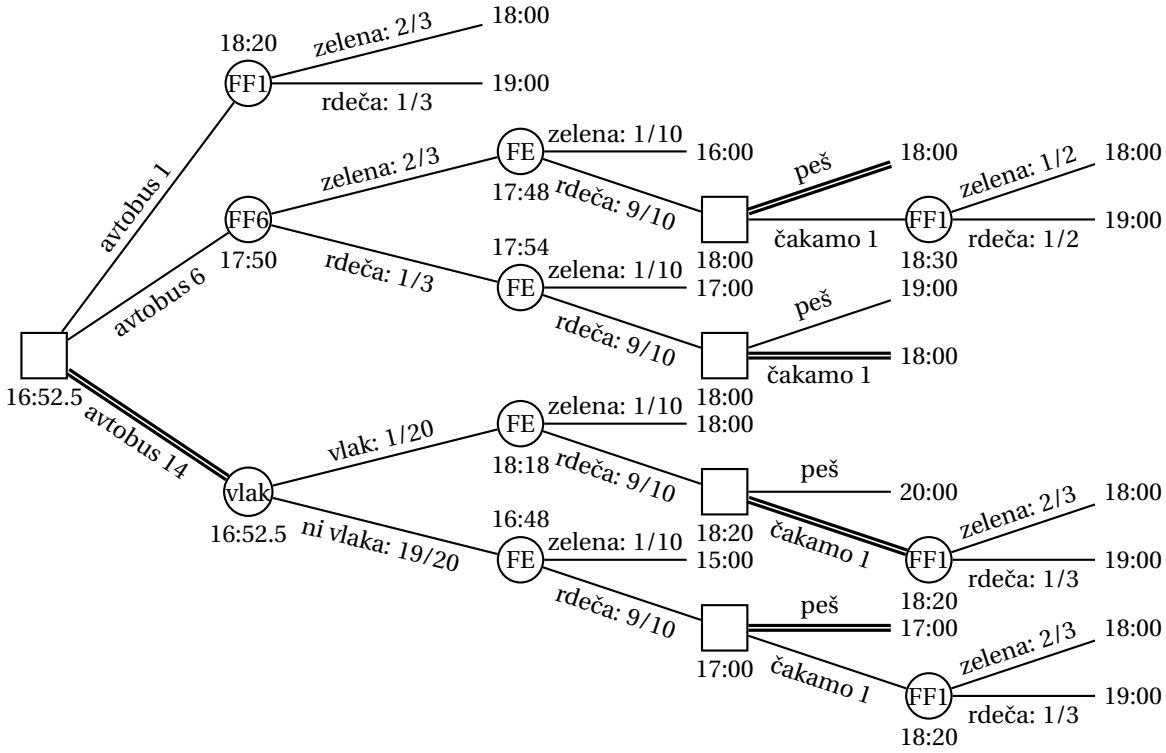
$$\forall h \in \{1,\dots,m\}\ \forall i,j \in \{1,\dots,n\}: 2x_{hij} \le x_{hii} + x_{hjj} \le x_{hij} + 1$$

Območja z mutacijo:

$$\forall i \in A\ \forall j \notin A: \qquad \sum_{h=1}^{m} x_{hij} = 0$$

Sorodniki:

$$\forall (i,j) \in B: \qquad \sum_{h=1}^{m} x_{hij} = 1$$



Slika 23: Odločitveno drevo za nalogo 3.1.

2.3 Teorija odločanja

Naloga 3.1.
Izračunamo lahko, da pot z avtobusom 1 traja 18 minut oziroma minuto dlje ob rdečem semaforju pri FF, pot z avtobusom 6 in naknadno pešačenje traja 16 minut ter minuto dlje ob rdečem semaforju pri FF in še dve minuti dlje ob rdečem semaforju pri FE, pot z avtobusom 14 in naknadno pešačenje pa traja 15 minut ter tri minute dlje ob čakanju na vlak in še dve minuti dlje ob rdečem semaforju pri FE. Če gremo na avtobus 6 in pri FF naletimo na zeleno luč, potem je verjetnost, da tudi avtobus 1 tam naleti na zeleno luč, enaka

$$P(\text{avtobus 1 ima zeleno pri FF} \mid \text{avtobus 6 ima zeleno pri FF}) = \frac{1/3}{2/3} = \frac{1}{2}.$$

S temi podatki lahko narišemo odločitveno drevo s slike 23, s katerega je razvidno, da pričakovani čas trajanja minimiziramo, če gremo na avtobus 14, potem pa v primeru čakanja na vlak in rdeče luči pri FE počakamo na avtobus 1, sicer pa pot nadaljujemo peš. Pričakovani čas trajanja poti je tako 16 minut in 52.5 sekund.

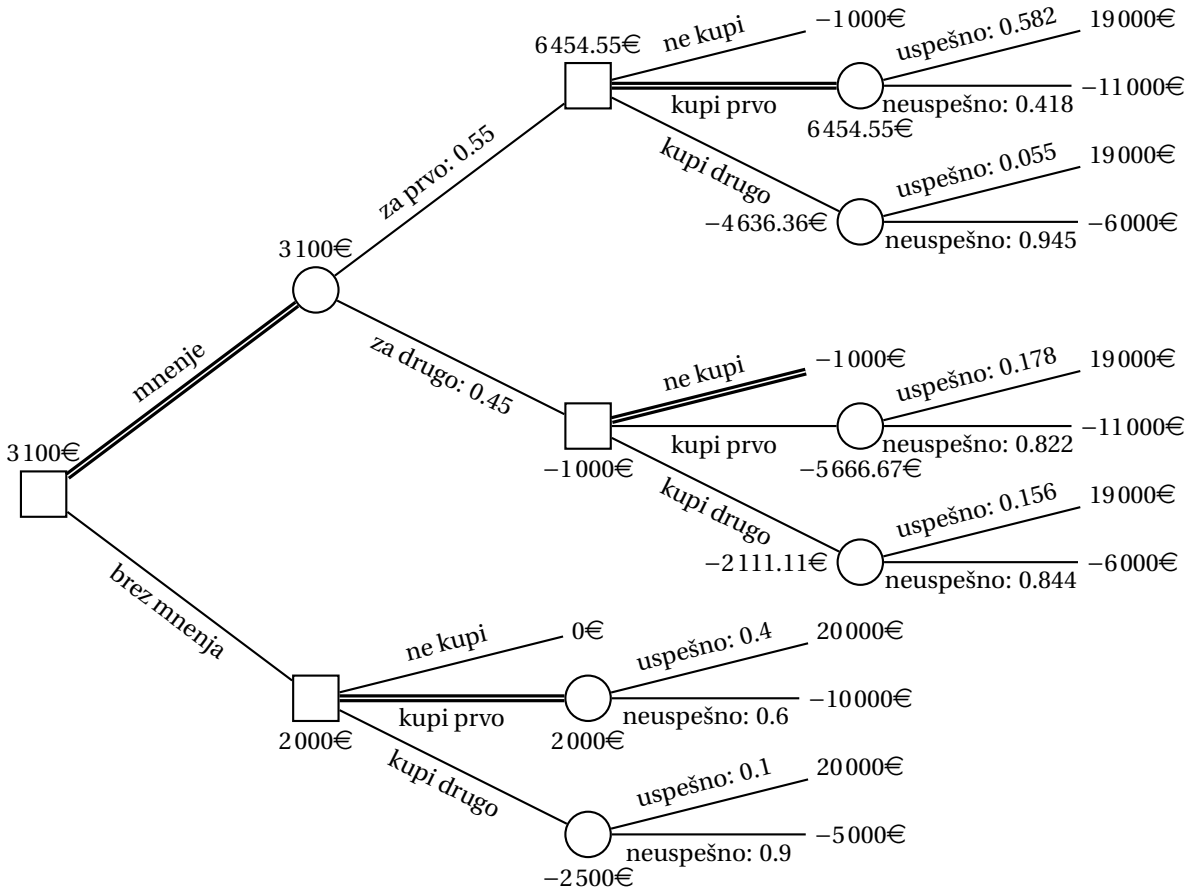
Naloga 3.2.
Naj bo A dogodek, da izvedenec pripiše večje možnosti prvemu podjetju, ter B_1, B_2 in B_0 dogodki, da uspe prvo, drugo oziroma nobeno podjetje. Izračunajmo verjetnosti za mnenje izvedenca ter uspešnosti podjetij v odvisnosti od njega.

$$\begin{aligned}
P(A) &= 0.4\cdot 0.8 + 0.1\cdot 0.3 + 0.5\cdot 0.4 = 0.55\\
P(\overline{A}) &= 0.4\cdot 0.2 + 0.1\cdot 0.7 + 0.5\cdot 0.6 = 0.45\\
P(B_0 \mid A) &= \frac{0.5\cdot 0.4}{0.55} = \frac{4}{11}\\
P(B_1 \mid A) &= \frac{0.4\cdot 0.8}{0.55} = \frac{32}{55}\\
P(B_2 \mid A) &= \frac{0.1\cdot 0.3}{0.55} = \frac{3}{55}\\
P(B_0 \mid \overline{A}) &= \frac{0.5\cdot 0.6}{0.45} = \frac{2}{3}\\
P(B_1 \mid \overline{A}) &= \frac{0.4\cdot 0.2}{0.45} = \frac{8}{45}\\
P(B_2 \mid \overline{A}) &= \frac{0.1\cdot 0.7}{0.45} = \frac{7}{45}
\end{aligned}$$

S pomočjo zgoraj izračunanih verjetnosti lahko narišemo odločitveno drevo s slike 24. Odločimo se torej za mnenje izvedenca – če je to naklonjeno prvemu podjetju, kupimo njegove delnice, sicer pa ne kupimo ničesar. Pričakovan dobiček je 3 100€.

Naloga 3.3.
Naj bo A dogodek, da nasprotnik na začetku vloži 10 žetonov, B_n dogodek, da odgovorimo z n žetoni ($n \in \{0, 10, 20\}$), C dogodek, da nasprotnik za tem izenači, in D dogodek, da dobimo igro. Izračunajmo ustrezne pogojne verjetnosti.

$$\begin{aligned}
P(A) &= 0.6\cdot 0.3 + 0.4\cdot 0.8 &&= 0.5\\
P(\overline{A}) &= 0.6\cdot 0.7 + 0.4\cdot 0.2 &&= 0.5\\
P(C \mid A\cap B_{20}) &= \frac{0.6\cdot 0.3\cdot 0.1 + 0.4\cdot 0.8\cdot 0.8}{0.5} &&= 0.548\\
P(\overline{C} \mid A\cap B_{20}) &= \frac{0.6\cdot 0.3\cdot 0.9 + 0.4\cdot 0.8\cdot 0.2}{0.5} &&= 0.452\\
P(C \mid \overline{A}\cap B_{10}) &= \frac{0.6\cdot 0.7\cdot 0.2 + 0.4\cdot 0.2\cdot 0.7}{0.5} &&= 0.28\\
P(\overline{C} \mid \overline{A}\cap B_{10}) &= \frac{0.6\cdot 0.7\cdot 0.8 + 0.4\cdot 0.2\cdot 0.3}{0.5} &&= 0.72\\
P(C \mid \overline{A}\cap B_{20}) &= \frac{0.6\cdot 0.7\cdot 0.1 + 0.4\cdot 0.2\cdot 0.8}{0.5} &&= 0.212\\
P(\overline{C} \mid \overline{A}\cap B_{20}) &= \frac{0.6\cdot 0.7\cdot 0.9 + 0.4\cdot 0.2\cdot 0.2}{0.5} &&= 0.788
\end{aligned}$$



Slika 24: Odločitveno drevo za nalogo 3.2.

$$P(D \mid A \cap B_{10}) = \frac{0.6 \cdot 0.3 \cdot 0.8 + 0.4 \cdot 0.8 \cdot 0.1}{0.5} = 0.352$$

$$P(\overline{D} \mid A \cap B_{10}) = \frac{0.6 \cdot 0.3 \cdot 0.2 + 0.4 \cdot 0.8 \cdot 0.9}{0.5} = 0.648$$

$$P(D \mid A \cap B_{20} \cap C) = \frac{0.6 \cdot 0.3 \cdot 0.1 \cdot 0.8 + 0.4 \cdot 0.8 \cdot 0.8 \cdot 0.1}{0.5 \cdot 0.548} \approx 0.146$$

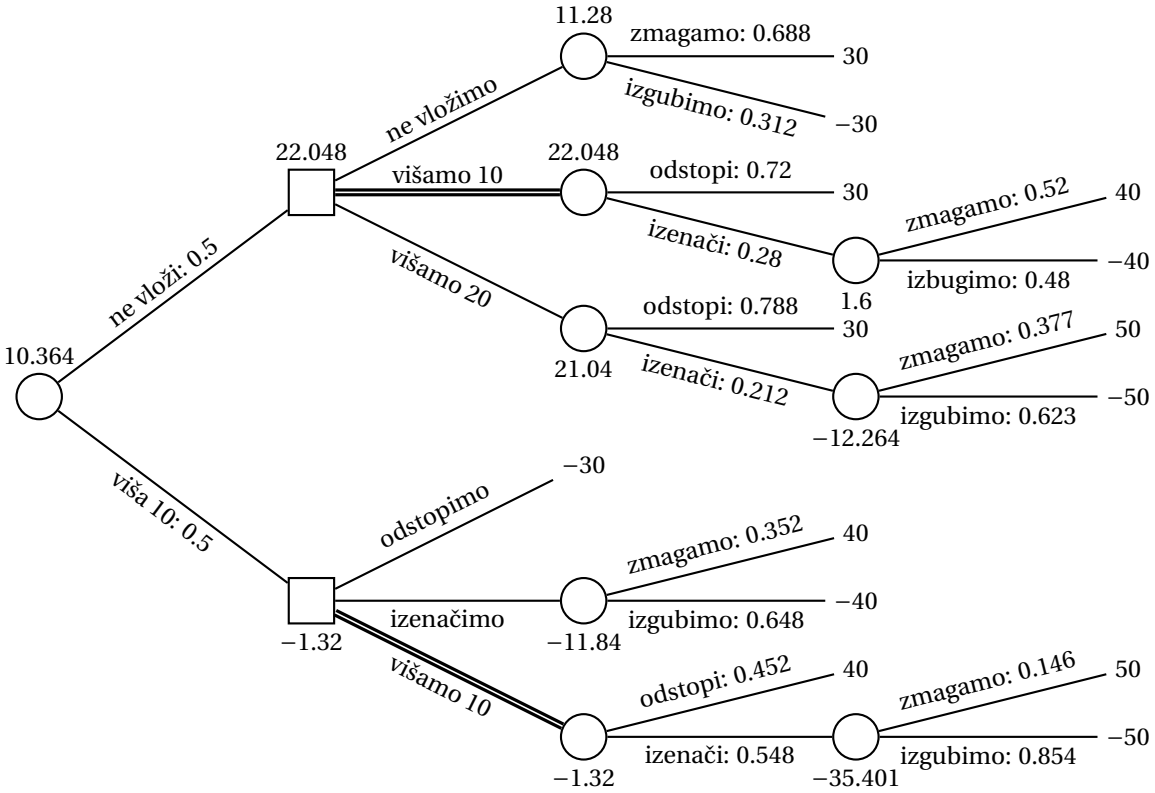
$$P(\overline{D} \mid A \cap B_{20} \cap C) = \frac{0.6 \cdot 0.3 \cdot 0.1 \cdot 0.2 + 0.4 \cdot 0.8 \cdot 0.8 \cdot 0.9}{0.5 \cdot 0.548} \approx 0.854$$

$$P(D \mid \overline{A} \cap B_0) = \frac{0.6 \cdot 0.7 \cdot 0.8 + 0.4 \cdot 0.2 \cdot 0.1}{0.5} = 0.688$$

$$P(\overline{D} \mid \overline{A} \cap B_0) = \frac{0.6 \cdot 0.7 \cdot 0.2 + 0.4 \cdot 0.2 \cdot 0.9}{0.5} = 0.312$$

$$P(D \mid \overline{A} \cap B_{10} \cap C) = \frac{0.6 \cdot 0.7 \cdot 0.2 \cdot 0.8 + 0.4 \cdot 0.2 \cdot 0.7 \cdot 0.1}{0.5 \cdot 0.28} = 0.52$$

$$P(\overline{D} \mid \overline{A} \cap B_{10} \cap C) = \frac{0.6 \cdot 0.7 \cdot 0.2 \cdot 0.2 + 0.4 \cdot 0.2 \cdot 0.7 \cdot 0.9}{0.5 \cdot 0.28} = 0.48$$



Slika 25: Odločitveno drevo za nalogo 3.3.

$$P(D \mid \overline{A} \cap B_{20} \cap C) = \frac{0.6 \cdot 0.7 \cdot 0.1 \cdot 0.8 + 0.4 \cdot 0.2 \cdot 0.8 \cdot 0.1}{0.5 \cdot 0.212} \approx 0.377$$
$$P(\overline{D} \mid \overline{A} \cap B_{20} \cap C) = \frac{0.6 \cdot 0.7 \cdot 0.1 \cdot 0.2 + 0.4 \cdot 0.2 \cdot 0.8 \cdot 0.9}{0.5 \cdot 0.212} \approx 0.623$$

S pomočjo zgoraj izračunanih verjetnosti lahko narišemo odločitveno drevo s slike 25. Opazimo, da se nam v vsakem primeru izplača višati za 10 žetonov (tj., če nasprotnik na začetku vloži 10 žetonov, odgovorimo z 20 žetoni). Pričakovani dobiček je 10.364 žetonov.

Naloga 3.4.
Izračunajmo najprej pričakovane vrednosti v vozliščih F, G in H odločitvenega drevesa s slike 2.

$$\begin{aligned} F &= 25 \cdot 2p - 10 \cdot (1-2p) = 70p - 10 \\ G &= 50 \cdot p - 10 \cdot (1-p) \quad\; = 60p - 10 \\ H &= 30 \cdot 4p - 30 \cdot (1-4p) = 240p - 30 \end{aligned}$$

Obravnavajmo najprej odločitve v vozliščih C, D in E. Za pot od vozlišča C k vozlišču F se odločimo, če velja $70p - 10 \geq 0$ oziroma $p \geq 1/7$. Za pot od vozlišča D k vozlišču G se odločimo, če velja $60p - 10 \geq -5$ oziroma $p \geq 1/12$. Za pot od vozlišča E k vozlišču H se odločimo, če velja $240p - 30 \geq -5$ oziroma $p \geq 5/48$. Sedaj lahko izračunamo pričakovano vrednost v vozlišču B v odvisnosti od parametra p.

$$\begin{aligned}
0 \leq p < \frac{1}{12} &\quad\Rightarrow\quad B = 0.6 \cdot (-5) + 0.4 \cdot (-5) &&= -5 \\
\frac{1}{12} \leq p < \frac{5}{48} &\quad\Rightarrow\quad B = 0.6 \cdot (60p - 10) + 0.4 \cdot (-5) &&= 36p - 8 \\
\frac{5}{48} \leq p \leq \frac{1}{4} &\quad\Rightarrow\quad B = 0.6 \cdot (60p - 10) + 0.4 \cdot (240p - 30) &&= 132p - 18
\end{aligned}$$

Obravnavajmo sedaj še odločitev v vozlišču A – za pot k vozlišču B se odločimo, če velja:

$$\begin{aligned}
0 \leq p < \frac{1}{12}: &\qquad -5 \geq 0 &&\Rightarrow \quad \bot \\
\frac{1}{12} \leq p < \frac{5}{48}: &\qquad 36p - 8 \geq 0 &&\Rightarrow \quad \bot \\
\frac{5}{48} \leq p < \frac{1}{7}: &\qquad 132p - 18 \geq 0 &&\Rightarrow \quad \frac{3}{22} \leq p < \frac{1}{7} \\
\frac{1}{7} \leq p \leq \frac{1}{4}: &\qquad 132p - 18 \geq 70p - 10 &&\Rightarrow \quad \frac{1}{7} \leq p \leq \frac{1}{4}
\end{aligned}$$

Odločamo se torej po sledečem pravilu.

- Če je $0 \leq p < 3/22$, se odločimo za pot preko vozlišča C v list z vrednostjo 0.
- Če je $3/22 \leq p \leq 1/4$, se odločimo za pot preko vozlišča B.
  - Če pridemo v vozlišče D, se odločimo za pot preko vozlišča G.
  - Če pridemo v vozlišče E, se odločimo za pot preko vozlišča H.

  Pričakovana vrednost je $132p - 18 \in [0, 15]$.

Proces odločanja je prikazan na sliki 26.

Naloga 3.5.

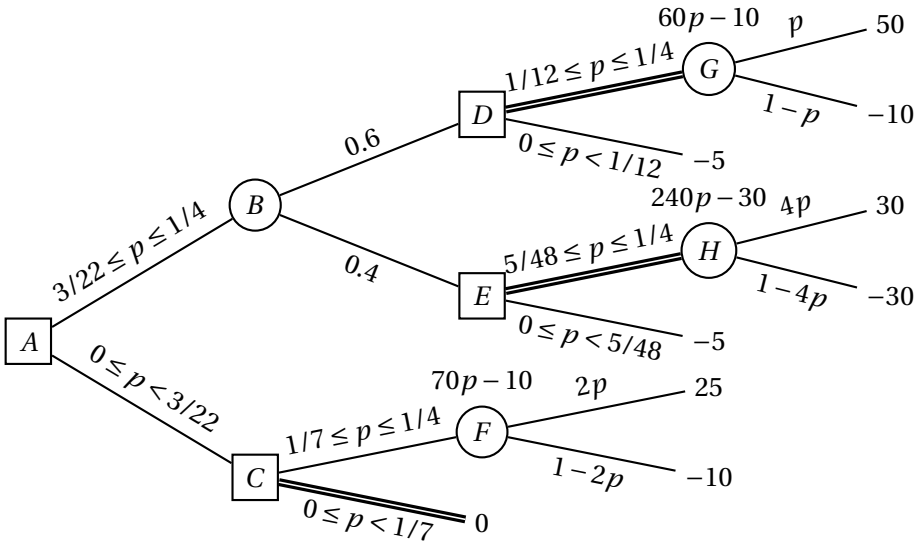
(a) Izračunajmo pričakovane dobičke pri različnih pogojih:

Izdela 500, prodaja po 50€, tržno zanimiv:

$$-10\,000€ + 500 \cdot 50€ = 15\,000€$$

Izdela 500, prodaja po 50€, tržno nezanimiv:

$$-10\,000€ + 250 \cdot 50€ = 2\,500€$$



Slika 26: Odločitveno drevo za nalogo 3.4 v odvisnosti od vrednosti parametra p.

Izdela 500, prodaja po 60€, tržno zanimiv:

$$-10\,000€ + 500 \cdot 60€ = 20\,000€$$

Izdela 500, prodaja po 60€, tržno nezanimiv:

$$-10\,000€ + 100 \cdot 60€ = -4\,000€$$

Izdela 1 000, prodaja po 50€, tržno zanimiv:

$$-18\,000€ + 650 \cdot 50€ = 14\,500€$$

Izdela 1 000, prodaja po 50€, tržno nezanimiv:

$$-18\,000€ + 250 \cdot 50€ = -5\,500€$$

Izdela 1 000, prodaja po 60€, tržno zanimiv:

$$-18\,000€ + 550 \cdot 60€ = 15\,000€$$

Izdela 1 000, prodaja po 60€, tržno nezanimiv:

$$-18\,000€ + 100 \cdot 60€ = -12\,000€$$

Odločitveno drevo je prikazano na sliki 27. Podjetnik naj se odloči, da naroči izdelavo 500 izdelkov, te pa prodaja po ceni 60€. Pričakovani dobiček je tedaj 15 200€.

(b) Najprej določimo verjetnosti, ki jih bomo potrebovali za odločitev.

$$P(\text{zmaga}) = 0.6 \cdot 0.8 + 0.1 \cdot 0.2 = 0.5$$
$$P(\text{ne zmaga}) = 0.4 \cdot 0.8 + 0.9 \cdot 0.2 = 0.5$$
$$P(\text{zanimiv} \mid \text{zmaga}) = \frac{0.6 \cdot 0.8}{0.5} = 0.96$$

$$P(\text{ni zanimiv} \mid \text{zmaga}) = \frac{0.1 \cdot 0.2}{0.5} = 0.04$$
$$P(\text{zanimiv} \mid \text{ne zmaga}) = \frac{0.4 \cdot 0.8}{0.5} = 0.64$$
$$P(\text{ni zanimiv} \mid \text{ne zmaga}) = \frac{0.9 \cdot 0.2}{0.5} = 0.36$$

Sedaj izračunajmo pričakovane dobičke pri različnih pogojih po zmagi na razpisu. Opazimo, da v nobenem pogoju pričakovano število prodanih izdelkov ne bo preseglo količine 980 izdelkov, kolikor jih ostane po kontroli kvalitete. Če ne zmaga na razpisu, so pričakovani dobički enaki tistim iz točke (a) pri izdelavi 1 000 kosov.

Zmaga, prodaja po 50€, tržno zanimiv:

$$-18\,000€ + 650 \cdot 50€ \cdot 1.2 + k = 21\,000€ + k$$

Zmaga, prodaja po 50€, tržno nezanimiv:

$$-18\,000€ + 250 \cdot 50€ \cdot 1.2 + k = -3\,000€ + k$$

Zmaga, prodaja po 60€, tržno zanimiv:

$$-18\,000€ + 550 \cdot 60€ \cdot 1.2 + k = 21\,600€ + k$$

Zmaga, prodaja po 60€, tržno nezanimiv:

$$-18\,000€ + 100 \cdot 60€ \cdot 1.2 + k = -10\,800€ + k$$

Z zgornjimi podatki lahko narišemo odločitveno drevo s slike 28. Izračunajmo pričakovane dobičke v vozliščih E, F, G, H.

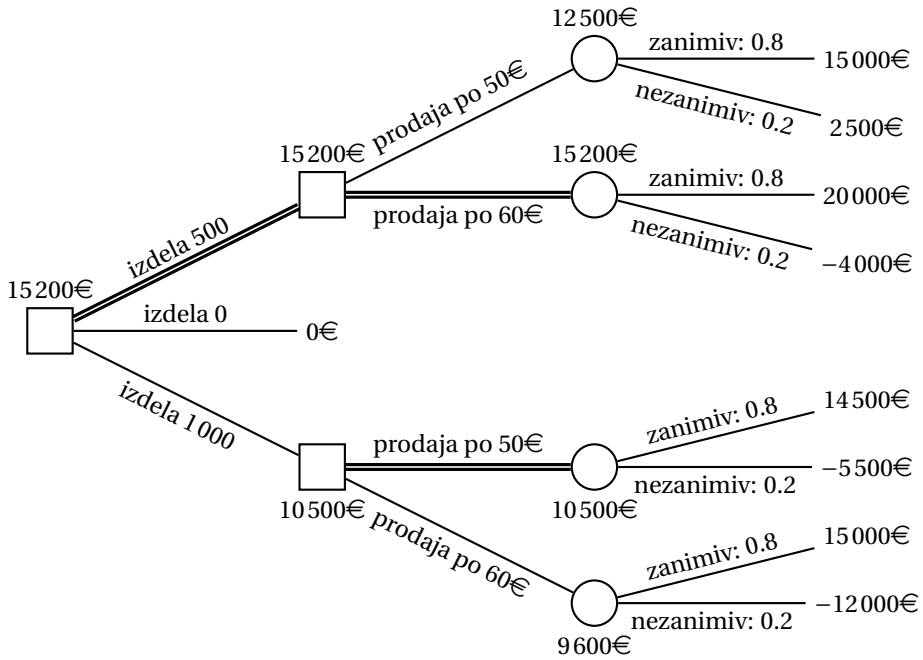
$$\begin{aligned}
E &= 0.96 \cdot (21\,000€ + k) + 0.04 \cdot (-3\,000€ + k) &&= 20\,040€ + k \\
F &= 0.96 \cdot (21\,600€ + k) + 0.04 \cdot (-10\,800€ + k) &&= 20\,304€ + k \\
G &= 0.64 \cdot 14\,500€ - 0.36 \cdot 5\,500€ &&= 7\,300€ \\
H &= 0.64 \cdot 15\,000€ - 0.36 \cdot 12\,000€ &&= 5\,280€
\end{aligned}$$

Očitno je v vozlišču C ugodnejša pot k vozlišču F, v vozlišču D pa je ugodnejša pot k vozlišču G – velja torej $C = 20\,304€ + k$ in $D = 7\,300€$. Sedaj lahko izračunamo še pričakovano vrednost v vozlišču B.

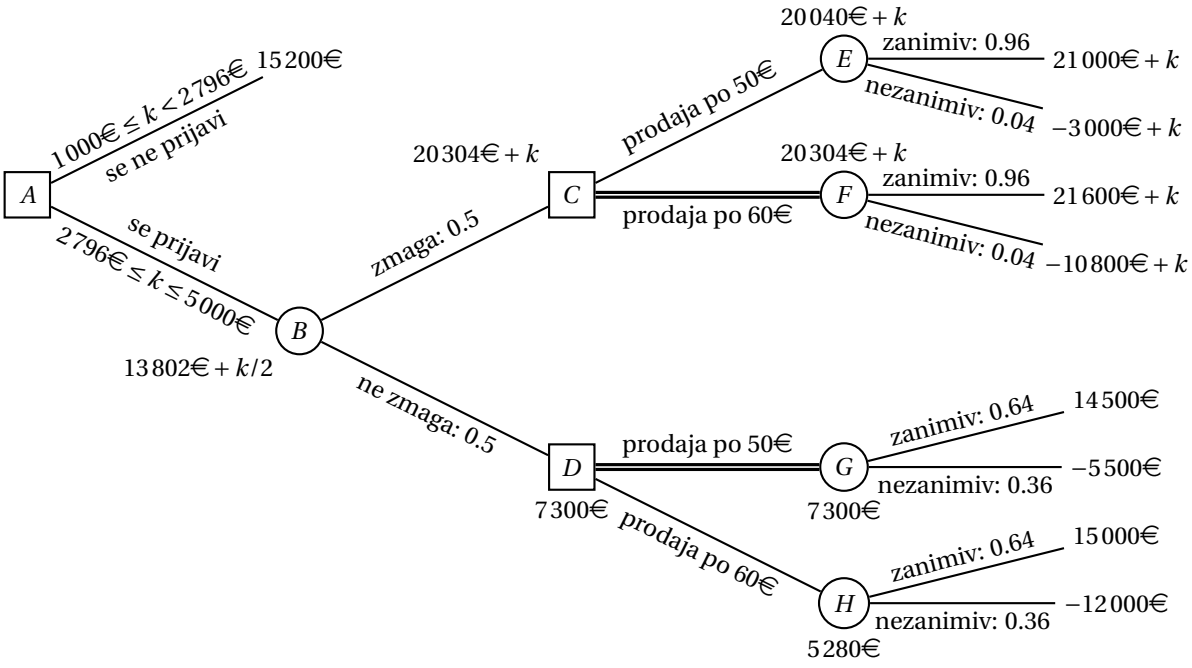
$$B = 0.5 \cdot (20\,304€ + k) + 0.5 \cdot 7\,300€ = 13\,802€ + \frac{k}{2}$$

Podjetnik naj se torej prijav na razpis, če velja $13\,802€ + k/2 \geq 15\,200€$ oziroma $k \geq 2\,796€$. Če nato zmaga na razpisu, naj izdelke prodaja po ceni 60€, v nasprotnem primeru pa naj jih prodaja po ceni 50€. Pričakovani dobiček je tedaj $13\,802€ + k/2 \in [15\,200€, 16\,302€]$.

Če velja $k < 2\,796€$, naj se podjetnik ne prijavi na razpis. Tako kot v točki (a) naj naroči izdelavo 500 izdelkov, te pa prodaja po ceni 60€. Pričakovani dobiček je tedaj 15 200€.



Slika 27: Odločitveno drevo za nalogo 3.5(a).



Slika 28: Odločitveno drevo za nalogo 3.5(b) v odvisnosti od vrednosti parametra k.

Naloga 3.6.
Če se odločimo za pot okoli gorovja, bomo porabili 12 ur, če se pa odločimo za pot čez prelaz, bomo porabili 6, 10 oziroma 14 ur, če je ta čist, delno zasnežen oziroma neprevozen. Če se pred tem odločimo za obisk vrača, se čas potovanja podaljša za eno uro.

Pričakovano število ur potovanja, če ne obiščemo vrača in se odločimo za pot čez prelaz, je enako

$$0.2 \cdot 6 + 0.1 \cdot 10 + 0.7 \cdot 14 = 12,$$

kar je ravno enako kot pot okoli gorovja.

Izračunajmo še verjetnosti za primer, da se odločimo za obisk vrača.

$$\begin{aligned}
P(\text{mir}) &= 0.9 \cdot 0.2 + 0.5 \cdot 0.1 + 0.1 \cdot 0.7 = \frac{3}{10} \\
P(\text{vojna}) &= 0.1 \cdot 0.2 + 0.5 \cdot 0.1 + 0.9 \cdot 0.7 = \frac{7}{10} \\
P(\text{čist} \mid \text{mir}) &= \frac{0.9 \cdot 0.2}{3/10} = \frac{3}{5} \\
P(\text{delno zasnežen} \mid \text{mir}) &= \frac{0.5 \cdot 0.1}{3/10} = \frac{1}{6} \\
P(\text{neprevozen} \mid \text{mir}) &= \frac{0.1 \cdot 0.7}{3/10} = \frac{7}{30} \\
P(\text{čist} \mid \text{vojna}) &= \frac{0.1 \cdot 0.2}{7/10} = \frac{1}{35} \\
P(\text{delno zasnežen} \mid \text{vojna}) &= \frac{0.5 \cdot 0.1}{7/10} = \frac{1}{14} \\
P(\text{neprevozen} \mid \text{vojna}) &= \frac{0.9 \cdot 0.7}{7/10} = \frac{9}{10}
\end{aligned}$$

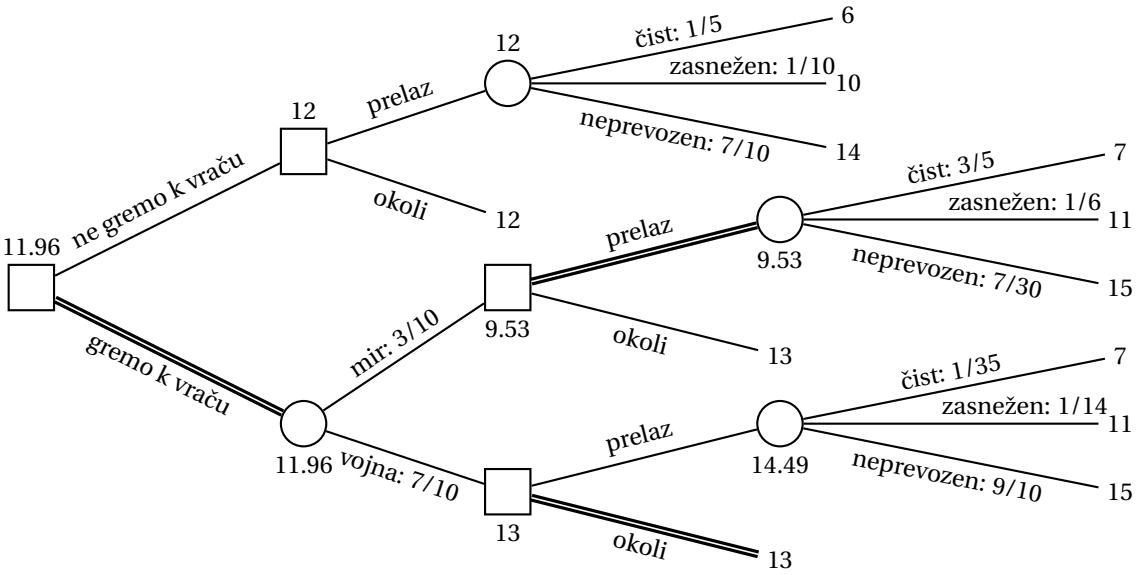
S pomočjo zgoraj izračunanih verjetnosti lahko narišemo odločitveno drevo s slike 29. Opazimo, da se nam izplača iti k vraču ter se odločiti za pot čez prelaz, če nam pove, da na gori vlada mir, in za pot okoli gorovja v nasprotnem primeru.

Naloga 3.7.
Izračunajmo najprej pričakovane vrednosti v vozliščih C, F in G odločitvenega drevesa s slike 4.

$$\begin{aligned}
C &= 0.5 \cdot (40p + 20) &&= 20p + 10 \\
F &= 50 \cdot p - 10 \cdot (1 - p) &&= 60p - 10 \\
G &= -30 \cdot p + 30 \cdot (1 - p) &&= -60p + 30
\end{aligned}$$

Obravnavajmo najprej odločitev v vozlišču D. Za pot k vozlišču F se odločimo, če velja $60p - 10 \geq -5$ oziroma $p \geq 1/12$.

Obravnavajmo sedaj še odločitev v vozlišču E. Za pot k vozlišču G se odločimo, če velja $-60p + 30 \geq -5$ oziroma $p \leq 7/12$.



Slika 29: Odločitveno drevo za nalogo 3.6.

Pričakovana vrednost v vozlišču B bo sledeča.

$$\begin{aligned}
p < \frac{1}{12}: &\qquad 0.9\cdot(-5)+(-60p+30)\cdot 0.1 = -1.5-6p\\
\frac{1}{12}\le p\le\frac{7}{12}: &\qquad 0.9\cdot(60p-10)+0.1\cdot(-60p+30) = -6+48p\\
p > \frac{7}{12}: &\qquad 0.9\cdot(60p-10)+0.1\cdot(-5) = -9.5+54p
\end{aligned}$$

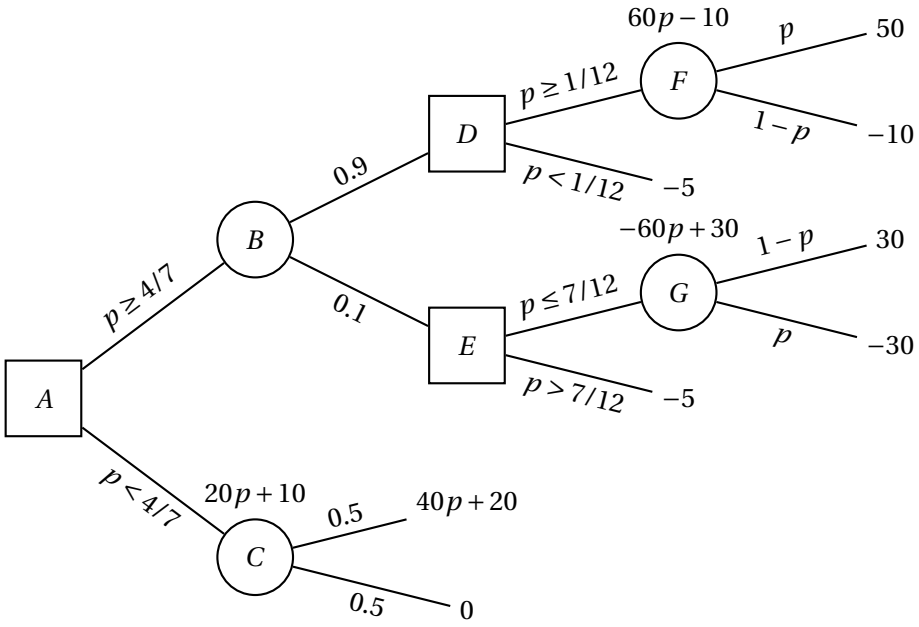
Obravnavajmo sedaj še odločitev v vozlišču A.

$$\begin{aligned}
0\le p < \frac{1}{12}: &\qquad \max\{-1.5-6p, 20p+10\} = 20p+10\\
\frac{1}{12}\le p\le\frac{7}{12}: &\qquad \max\{-6+48p, 20p+10\} = \begin{cases} 20p+10 & p<\frac{4}{7}\\ -6+48p & p\ge\frac{4}{7}\end{cases}\\
p > \frac{7}{12}: &\qquad \max\{-9.5+54p, 20p+10\} = -9.5+54p
\end{aligned}$$

Proces odločanja je prikazan na sliki 30.

Naloga 3.8.
Če se podjetje odloči za samostojno prodajo (brez predhodne akcijske ponudbe), bo ob uspehu imelo $100\,000\cdot(3€-0.5€)-1\,000€ = 249\,000€$ dobička, ob neuspehu pa $10\,000\cdot 3€-100\,000\cdot 0.5€-1\,000€ = -21\,000€$, torej $21\,000€$ izgube. Ker je pričakovani dobiček enak

$$0.7\cdot 249\,000€-0.3\cdot 21\,000€ = 168\,000€,$$



Slika 30: Odločitveno drevo za nalogo 3.7 v odvisnosti od vrednosti parametra p.

kar je več kot $100\,000 \cdot 1€ = 100\,000€$, kolikor bi zaslužili ob prodaji multinacionalki, se podjetju v primeru, da se ne odločijo za akcijsko ponudbo, bolj izplača samostojna prodaja.

Izračunajmo še dobičke, če se odločijo za akcijsko ponudbo:

Akcija uspe, produkt uspe:

$$1\,000 \cdot 2.5€ + 99\,000 \cdot 3€ - 100\,000 \cdot 0.5€ - 2 \cdot 1\,000€ = 247\,500€$$

Akcija uspe, produkt ne uspe:

$$1\,000 \cdot 2.5€ + 9\,000 \cdot 3€ - 100\,000 \cdot 0.5€ - 2 \cdot 1\,000€ = -22\,500€$$

Akcija uspe, prodajo multinacionalki:

$$1\,000 \cdot (2.5€ - 0.5€) + 99\,000 \cdot 1€ - 1\,000€ = 100\,000€$$

Akcija ne uspe, produkt uspe:

$$100 \cdot 2.5€ + 99\,900 \cdot 3€ - 100\,000 \cdot 0.5€ - 2 \cdot 1\,000€ = 247\,950€$$

Akcija ne uspe, produkt ne uspe:

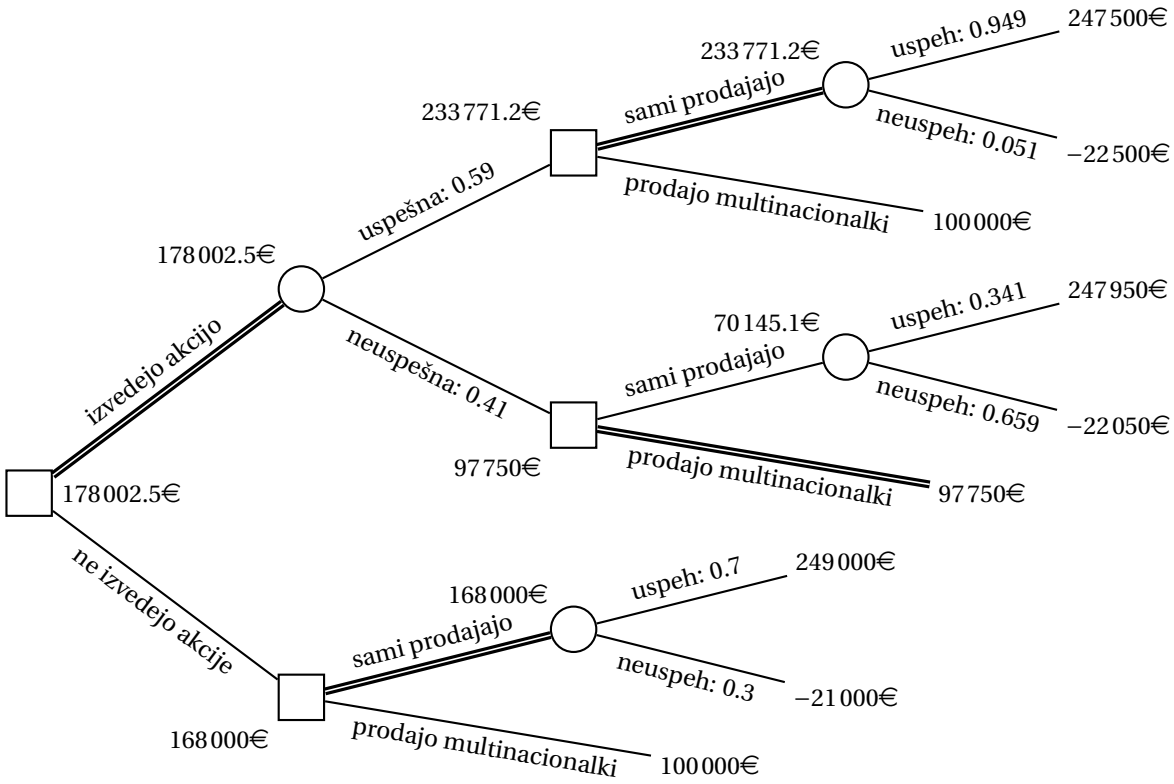
$$100 \cdot 2.5€ + 9\,900 \cdot 3€ - 100\,000 \cdot 0.5€ - 2 \cdot 1\,000€ = -22\,050€$$

Akcija ne uspe, prodajo multinacionalki:

$$100 \cdot 2.5€ - 1\,000 \cdot 0.5€ - 1\,000€ + 99\,000 \cdot 1€ = 97\,750€$$

Izračunajmo še verjetnosti uspeha akcije ter uspešnosti produkta ob vsaki napovedi.

$$P(\text{akcija uspešna}) = 0.7 \cdot 0.8 + 0.3 \cdot 0.1 = 0.59$$



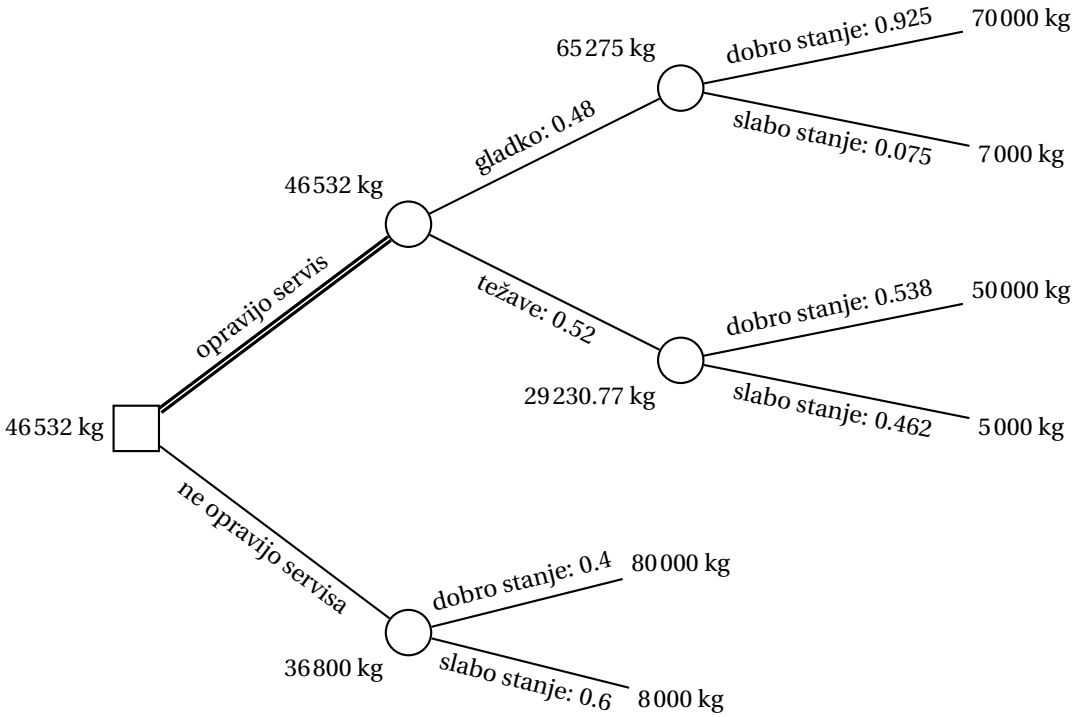
Slika 31: Odločitveno drevo za nalogo 3.8.

$$
\begin{aligned}
P(\text{akcija neuspešna}) &= 0.7 \cdot 0.2 + 0.3 \cdot 0.9 = 0.41 \\
P(\text{produkt uspe} \mid \text{akcija uspešna}) &= \frac{0.7 \cdot 0.8}{0.59} = \frac{56}{59} \approx 0.949 \\
P(\text{produkt ne uspe} \mid \text{akcija uspešna}) &= \frac{0.3 \cdot 0.1}{0.59} = \frac{3}{59} \approx 0.051 \\
P(\text{produkt uspe} \mid \text{akcija neuspešna}) &= \frac{0.7 \cdot 0.2}{0.41} = \frac{14}{41} \approx 0.341 \\
P(\text{produkt ne uspe} \mid \text{akcija neuspešna}) &= \frac{0.3 \cdot 0.9}{0.41} = \frac{27}{41} \approx 0.659
\end{aligned}
$$

S pomočjo zgoraj izračunanih verjetnosti lahko narišemo odločitveno drevo s slike 31. Opazimo, da se podjetju izplača izvesti akcijo, saj je pričakovani dobiček v tem primeru 178002.5€. Če akcija uspe, naj se odločijo za samostojno prodajo, če pa ne uspe, pa naj zaloge prodajo multinacionalki.

Naloga 3.9.
Izračunajmo verjetnosti za potek servisa in stanje strojev po izvedenem servisu.

$$P(\text{servis poteka gladko}) = 0.9 \cdot 0.4 + 0.2 \cdot 0.6 \qquad = 0.48$$



Slika 32: Odločitveno drevo za nalogo 3.9.

$$
\begin{aligned}
P(\text{servis se zavleče}) &= 0.1 \cdot 0.4 + 0.8 \cdot 0.6 && = 0.52 \\
P(\text{stroji v dobrem stanju} \mid \text{servis poteka gladko}) &= \frac{0.9 \cdot 0.4 + 0.2 \cdot 0.6 \cdot 0.7}{0.48} && = 0.925 \\
P(\text{stroji v slabem stanju} \mid \text{servis poteka gladko}) &= \frac{0.2 \cdot 0.6 \cdot 0.3}{0.48} && = 0.075 \\
P(\text{stroji v dobrem stanju} \mid \text{servis se zavleče}) &= \frac{0.1 \cdot 0.4 + 0.8 \cdot 0.6 \cdot 0.5}{0.52} && \approx 0.538 \\
P(\text{stroji v slabem stanju} \mid \text{servis se zavleče}) &= \frac{0.8 \cdot 0.6 \cdot 0.5}{0.52} && \approx 0.462
\end{aligned}
$$

S pomočjo zgoraj izračunanih verjetnosti lahko narišemo odločitveno drevo s slike 32. Opazimo, da se podjetju izplača opraviti servis, pričakovana količina proizvedene čokolade pa je tedaj 46532 kg.

Naloga 3.10.
Izračunajmo najprej pričakovane vrednosti v vozliščih F in G odločitvenega drevesa s slike 5.

$$
\begin{aligned}
F &= 160 \cdot p + 40 \cdot (1-p) && = 120p + 40 \\
G &= 120 \cdot 2p + 20 \cdot (1-2p) && = 200p + 20
\end{aligned}
$$

Obravnavajmo najprej odločitev v vozlišču D. Za pot k vozlišču F se odločimo, če velja $120p+40<60$ oziroma $p<1/6$.

Obravnavajmo sedaj še odločitev v vozlišču E. Za pot k vozlišču G se odločimo, če velja $200p+20<50$ oziroma $p<3/20$.

Pričakovana vrednost v vozlišču B bo sledeča.

$$\begin{aligned} p<\frac{1}{6}: &\qquad 0.7\cdot(120p+40)+0.3\cdot(100p+30)=114p+37\\ p\geq\frac{1}{6}: &\qquad 0.7\cdot 60+0.3\cdot(100p+30)=30p+51 \end{aligned}$$

Izračunajmo še pričakovano vrednost v vozlišču C.

$$\begin{aligned} p<\frac{3}{20}: &\qquad 0.6\cdot(200p+20)+0.4\cdot 70=120p+40\\ p\geq\frac{3}{20}: &\qquad 0.6\cdot 50+0.4\cdot 70=58 \end{aligned}$$

Nazadnje obravnavajmo še odločitev v vozlišču A.

$$\begin{aligned} 0\leq p<\frac{3}{20}: &\qquad \min\{114p+37,120p+40\}=114p+37\\ \frac{3}{20}\leq p<\frac{1}{6}: &\qquad \min\{114p+37,58\}=114p+37\\ \frac{1}{6}\leq p\leq\frac{1}{2}: &\qquad \min\{30p+51,58\}=\begin{cases}30p+51 & p<\frac{7}{30}\\ 58 & p\geq\frac{7}{30}\end{cases} \end{aligned}$$

Proces odločanja je prikazan na sliki 33. Ker velja $\frac{7}{30}>\frac{3}{20}$, opazimo, da v primeru $p\geq\frac{7}{30}$ iz vozlišča E nikoli ne izberemo poti proti vozlišču G.

Naloga 3.11.

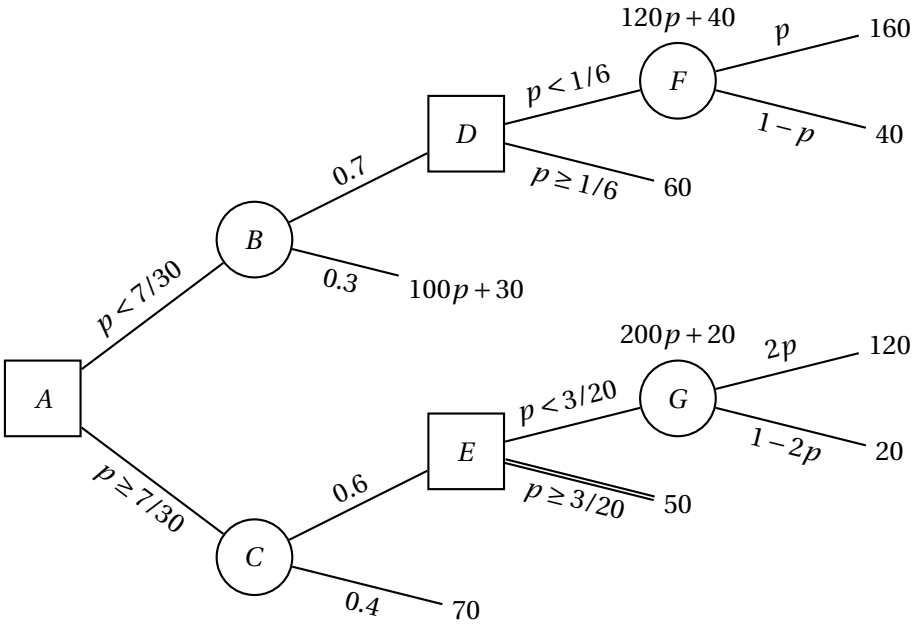
(a) Naj bo X_k $(200\leq k\leq 203)$ dobiček, če družba proda k vozovnic.

$$\begin{aligned} E(X_{200}) &= 200\cdot 35€ &&= 7000€\\ E(X_{201}) &= 201\cdot 35€-0.3\cdot 60€ &&= 7017€\\ E(X_{202}) &= 202\cdot 35€-(0.3\cdot 2+0.4)\cdot 60€ &&= 7010€\\ E(X_{203}) &= 203\cdot 35€-(0.3\cdot 3+0.4\cdot 2+0.2)\cdot 60€ &&= 6991€ \end{aligned}$$

Vidimo, da se družbi najbolj izplača prodati 201 vozovnico.

(b) Naj bo Y_k $(200\leq k\leq 203)$ dobiček, če družba proda k vozovnic, pri čemer lahko uporabi tudi dodatno mesto.

$$\begin{aligned} E(Y_{200}) &= 200\cdot 35€ &&= 7000€\\ E(Y_{201}) &= 201\cdot 35€-0.3\cdot 0.001\cdot 40000€ &&= 7023€ \end{aligned}$$



Slika 33: Odločitveno drevo za nalogo 3.10 v odvisnosti od vrednosti parametra p.

$$E(Y_{202}) = 202 \cdot 35€ - 0.3 \cdot 60€ - 0.7 \cdot 0.001 \cdot 40\,000€ \qquad = 7\,024€$$
$$E(Y_{203}) = 203 \cdot 35€ - (0.3 \cdot 2 + 0.4) \cdot 60€ - 0.9 \cdot 0.001 \cdot 40\,000€ = 7\,009€$$

Največji pričakovan dobiček je dosežen pri prodaji 202 vozovnic. Ker je ta večji kot v primeru, ko se dodatno mesto ne uporabi, se družbi torej to mesto izplača uporabiti.

2.4 Dinamično programiranje

Naloga 4.1.
Naj bo p_{ij} največji dobiček, ki ga lahko iztržimo z rezanjem kosa blaga dimenzij $i \times j$ ($1 \le i \le m$, $1 \le j \le n$). Določimo rekurzivne enačbe.

$$\begin{aligned} p_{ij} = \max \big(& \{c_h \mid 1 \le h \le k \wedge (a_h, b_h) \in \{(i,j),(j,i)\}\} \\ & \cup \{p_{\ell j} + p_{i-\ell,j} \mid 1 \le \ell \le i/2\} \\ & \cup \{p_{i\ell} + p_{i,j-\ell} \mid 1 \le \ell \le j/2\} \cup \{0\} \big) \end{aligned}$$

Prva množica tukaj obravnava primere, ko imamo kos blaga želene velikosti, naslednji dve pa obravnavata vodoravne oziroma navpične reze. Zadnja množica poskrbi, da je vrednost p_{11} definirana tudi v primeru, ko nam kos blaga velikosti 1×1 ne prinese dobička.

Vrednosti p_{ij} računamo v leksikografskem vrstnem redu indeksov (npr. najprej naraščajoče po i, nato pa naraščajoče po j). Maksimalni dobiček dobimo kot $p^* = p_{mn}$.

Naloga 4.2.
Algoritem, ki izhaja iz rekurzivnih enačb, ima časovno zahtevnost $O(mn(m+n+k))$. Z ustrezno podatkovno strukturo (zgoščena tabela) je mogoče časovno zahtevnost izboljšati na $O(mn(m+n)+k)$.

Izračunajmo vrednosti p_{ij} ($1 \le i \le m, 1 \le j \le n$) po rekurzivnih enačbah iz rešitve naloge 4.1. Pri tem upoštevamo, da velja $p_{ij} = p_{ji}$, tako da bomo izračunali samo primere z $i \ge j$.

$p_{11} = \max\{0\}$	$= 0$		
$p_{21} = \max\{0+0, 0\}$	$= 0$		
$p_{22} = \max\{\underline{6}, 0+0, 0\}$	$= 6$		izdelek 1
$p_{31} = \max\{\underline{3}, 0+0, 0\}$	$= 3$		izdelek 2
$p_{32} = \max\{\underline{7}, 0+6, 3+3, 0\}$	$= 7$		izdelek 4
$p_{33} = \max\{\underline{3+7}, 0\}$	$= 10$		razrez na 1×3 in 2×3
$p_{41} = \max\{\underline{5}, 0+3, 0+0, 0\}$	$= 5$		izdelek 3
$p_{42} = \max\{0+7, \underline{6+6}, 5+5, 0\}$	$= 12$		razrez na 2×3 in 2×2
$p_{43} = \max\{3+10, 7+7, \underline{5+12}, 0\}$	$= 17$		razrez na 4×1 in 4×2
$p_{51} = \max\{\underline{0+5}, 0+3, 0\}$	$= 5$		razrez na 1×1 in 4×1
$p_{52} = \max\{0+12, \underline{6+7}, 5+5, 0\}$	$= 13$		razrez na 2×2 in 3×2
$p_{53} = \max\{\underline{3+17}, 7+10, 5+13, 0\}$	$= 20$		razrez na 1×3 in 4×3

Maksimalni dobiček je torej $p^* = p_{53} = 20$, dosežemo pa ga tako, da iz prvotnega kosa blaga dimenzij 5×3 odrežemo kos dimenzij 1×3 za izdelek s ceno 3, nato od preostanka dimenzij 4×3 odrežemo kos dimenzij 4×1 za izdelek s ceno 5,

nazadnje pa preostanek dimenzij 4×2 razrežemo na dva kosa dimenzij 2×2 za izdelka s ceno 6.

Naloga 4.3.

(a) Naj bo x_i največji produkt strnjenega podzaporedja, ki se konča pri številu a_i, in y_i najmanjši tak produkt. Določimo začetne pogoje in rekurzivne enačbe.

$$\begin{aligned} x_1 &= y_1 = a_1 \\ x_i &= \max\{a_i, a_i x_{i-1}, a_i y_{i-1}\} && (2 \le i \le n) \\ y_i &= \min\{a_i, a_i x_{i-1}, a_i y_{i-1}\} && (2 \le i \le n) \end{aligned}$$

Vrednosti x_i in y_i računamo naraščajoče po indeksu i. Največji produkt strnjenega podzaporedja dobimo kot $x^* = \max\{x_i \mid 1 \le i \le n\}$.

(b) Za izračun vrednosti x_i in y_i pri vsakem i potrebujemo konstantno časa. Ker naredimo n takih korakov, je časovna zahtevnost algoritma $O(n)$.

(c) Izračunajmo vrednosti x_i in y_i $(2 \le i \le 10)$.

$$\begin{aligned} x_2 &= \max\{-2, \underline{-2 \cdot 0.9}, -2 \cdot 0.9\} && = -1.8 \\ y_2 &= \min\{\underline{-2}, -2 \cdot 0.9, -2 \cdot 0.9\} && = -2 \\ x_3 &= \max\{-0.6, -0.6 \cdot -1.8, \underline{-0.6 \cdot -2}\} && = 1.2 \\ y_3 &= \min\{\underline{-0.6}, -0.6 \cdot -1.8, -0.6 \cdot -2\} && = -0.6 \\ x_4 &= \max\{-0.5, -0.5 \cdot 1.2, \underline{-0.5 \cdot -0.6}\} && = 0.3 \\ y_4 &= \min\{-0.5, \underline{-0.5 \cdot 1.2}, -0.5 \cdot -0.6\} && = -0.6 \\ x_5 &= \max\{-2, -2 \cdot 0.3, \underline{-2 \cdot -0.6}\} && = 1.2 \\ y_5 &= \min\{\underline{-2}, -2 \cdot 0.3, -2 \cdot -0.6\} && = -2 \\ x_6 &= \max\{5, \underline{5 \cdot 1.2}, 5 \cdot -2\} && = 6 \\ y_6 &= \min\{5, 5 \cdot 1.2, \underline{5 \cdot -2}\} && = -10 \\ x_7 &= \max\{0.1, \underline{0.1 \cdot 6}, 0.1 \cdot -10\} && = 0.6 \\ y_7 &= \min\{0.1, 0.1 \cdot 6, \underline{0.1 \cdot -10}\} && = -1 \\ x_8 &= \max\{\underline{3}, 3 \cdot 0.6, 3 \cdot -1\} && = 3 \\ y_8 &= \min\{3, 3 \cdot 0.6, \underline{3 \cdot -1}\} && = -3 \\ x_9 &= \max\{0.5, \underline{0.5 \cdot 3}, 0.5 \cdot -3\} && = 1.5 \\ y_9 &= \min\{0.5, 0.5 \cdot 3, \underline{0.5 \cdot -3}\} && = -1.5 \\ x_{10} &= \max\{-3, -3 \cdot 1.5, \underline{-3 \cdot -1.5}\} && = 4.5 \\ y_{10} &= \min\{-3, \underline{-3 \cdot 1.5}, -3 \cdot -1.5\} && = -4.5 \end{aligned}$$

Največji produkt strnjenega zaporedja je torej $x^* = 6 = \prod_{i=2}^{6} a_i$.

Naloga 4.4.

(a) Za izračun pričakovanega dobička bomo definirali funkcije $v_i(z)$ $(i = 1,2,3)$. Zapišimo njihove definicije.

$$\begin{aligned} v_1(z) &= n_1 + k_1 z \\ v_2(z) &= \max\{v_1(z-y) + n_2 + k_2 y \mid a_2 \le y \le z - a_1\} \\ v_3(z) &= \max\{v_2(z-y)\cdot(n_3 + k_3 y) \mid a_3 \le y \le z - a_1 - a_2\} \end{aligned}$$

Pričakovani dobiček dobimo tako, da izračunamo $v_3(m)$.

(b) Vstavimo podatke v zgornje formule.

$$\begin{aligned} v_1(z) &= 3 + 1.5z \\ v_2(z) &= \max\{3 + 1.5(z-y) + 4 + 2y \mid 3 \le y \le z - 4\} \\ &= \max\{7 + 1.5z + 0.5y \mid 3 \le y \le z - 4\} \\ &= 5 + 2z \qquad (z \ge 7) \\ v_3(15) &= \max\{(5 + 2(15-y))\cdot(0.4 + 0.3y) \mid 2 \le y \le 8\} \\ &= \max\{-0.6y^2 + 9.7y + 14 \mid 2 \le y \le 8\} \end{aligned}$$

Naj bo $f(y) = -0.6y^2 + 9.7y + 14$ – gre za kvadraten polinom z negativnim vodilnim členom, tako da ima maksimum tam, kjer je odvod ničeln.

$$\begin{aligned} 0 = f'(y) &= -1.2y + 9.7 \\ y &= 9.7/1.2 > 8 \end{aligned}$$

Opazimo torej, da maksimum leži desno od intervala $[2,8]$, kar pomeni, da velja $v_3(15) = f(8) = 53.6$. Podjetje bo torej maksimiziralo pričakovani dobiček na 53.8 milijonov evrov, če dodeli 8 milijonov evrov marketingu, 3 milijone evrov oblikovalcem in 4 milijone evrov razvijalcem.

Naloga 4.5.

(a) Naj bo q_{ij} največja verjetnost, ki jo lahko dosežejo, da pridobijo podporo prvih j strank, če k njim pošljejo i lobistov. Določimo začetne pogoje in rekurzivne enačbe.

$$\begin{aligned} q_{i1} &= p_{i1} && (1 \le i \le m) \\ q_{ij} &= \max\{q_{h,j-1} p_{i-h,j} \mid 0 \le h \le i\} && (2 \le j \le n, 0 \le i \le m) \end{aligned}$$

Vrednosti q_{ij} računamo v leksikografskem vrstnem redu glede na (j,i). Največjo verjetnost, da pridobijo podporo vseh n strank, dobimo kot $q^* = q_{mn}$.

(b) Ker velja $p_{0j} = 0$ za vsak j, bo veljalo $q_{ij} = 0$ za vse $i < j$. Tako lahko v zgornji rekurzivni enačbi upoštevamo le primere z $j-1 \le h \le i-1$. Izračunajmo torej vrednosti q_{i2} $(2 \le i \le 5)$ in $q^* = q_{63}$.

$$\begin{aligned}
q_{22} &= 0.2 \cdot 0.4 && = 0.08\\
q_{32} &= \max\{0.2 \cdot 0.5, \underline{0.5 \cdot 0.4}\} && = 0.2\\
q_{42} &= \max\{0.2 \cdot 0.5, 0.5 \cdot 0.5, \underline{0.7 \cdot 0.4}\} && = 0.28\\
q_{52} &= \max\{0.2 \cdot 0.6, 0.5 \cdot 0.5, \underline{0.7 \cdot 0.5}, 0.8 \cdot 0.4\} && = 0.35\\
q_{63} &= \max\{0.08 \cdot 0.9, \underline{0.2 \cdot 0.8}, 0.28 \cdot 0.4, 0.35 \cdot 0.3\} && = 0.16
\end{aligned}$$

Največja verjetnost je torej $q^* = q_{63} = 0.16$. Poiščimo še optimalno razporeditev lobistov.

$q_{63} = q_{32} p_{33}$	3 lobisti k tretji stranki
$q_{32} = q_{21} p_{12}$	1 lobist k drugi stranki
$q_{21} = p_{21}$	2 lobista k prvi stranki

Naloga 4.6.

(a) Naj bosta p_i in q_i največji vsoti za ustrezne postavitve domin do i-tega polja, pri čemer na i-tem polju dovolimo le del domine z znakom + oziroma − (tj., prepovemo − oziroma +, dovolimo pa, da i-to polje ni pokrito). Določimo začetne pogoje in rekurzivne enačbe.

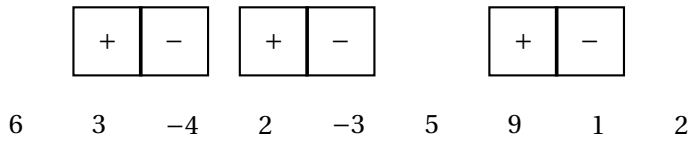
$$\begin{aligned}
p_0 = p_1 = 0, &\qquad p_i = \max\{p_{i-1}, q_{i-1}, p_{i-2} - a_{i-1} + a_i\} &\qquad (2 \le i \le n)\\
q_0 = q_1 = 0, &\qquad q_i = \max\{p_{i-1}, q_{i-1}, q_{i-2} + a_{i-1} - a_i\} &\qquad (2 \le i \le n)
\end{aligned}$$

Vrednosti p_i in q_i $(0 \le i \le n)$ računamo naraščajoče po indeksu i. Največjo vsoto dobimo kot $p^* = \max\{p_n, q_n\}$.

(b) Za izračun vrednosti p_i in q_i pri vsakem i potrebujemo konstantno časa. Ker naredimo n takih korakov, je časovna zahtevnost algoritma $O(n)$.

(c) Izračunajmo vrednosti p_i in q_i $(2 \le i \le 9)$.

$$\begin{aligned}
p_2 &= \max\{\underline{0}, 0, 0-6+3\} && = 0 & \qquad q_2 &= \max\{0, 0, \underline{0+6-3}\} && = 3\\
p_3 &= \max\{0, \underline{3}, 0-3+(-4)\} && = 3 & \qquad q_3 &= \max\{0, 3, \underline{0+3+(-4)}\} && = 7\\
p_4 &= \max\{3, \underline{7}, 0-(-4)+2\} && = 7 & \qquad q_4 &= \max\{3, \underline{7}, 3+(-4)-2\} && = 7\\
p_5 &= \max\{\underline{7}, 7, 3-2+(-3)\} && = 7 & \qquad q_5 &= \max\{7, 7, \underline{7+2-(-3)}\} && = 12\\
p_6 &= \max\{7, 12, \underline{7-(-3)+5}\} && = 15 & \qquad q_6 &= \max\{7, \underline{12}, 7+(-3)-5\} && = 12\\
p_7 &= \max\{\underline{15}, 12, 7-5+9\} && = 15 & \qquad q_7 &= \max\{\underline{15}, 12, 7+5-9\} && = 15\\
p_8 &= \max\{\underline{15}, 15, 15-9+1\} && = 15 & \qquad q_8 &= \max\{15, 15, \underline{15+9-1}\} && = 23
\end{aligned}$$



Slika 34: Optimalno pokritje za nalogo 4.6(c).

$$p_9 = \max\{15, \underline{23}, 15-1+2\} = 23 \qquad q_9 = \max\{15, \underline{23}, 15+1-2\} = 23$$

Optimalno pokritje ima torej vsoto $p^* = \max\{p_9, q_9\} = 23$. Poglejmo, kam moramo postaviti domine.

$$\begin{aligned} p^* = p_9 = q_8 &= q_6 + a_7 - a_8 && [+\,-] \text{ na } (7,8) \\ q_6 = q_5 &= q_3 + a_4 - a_5 && [+\,-] \text{ na } (4,5) \\ q_3 &= q_1 + a_2 - a_3 && [+\,-] \text{ na } (2,3) \end{aligned}$$

Postavitev je prikazana na sliki 34.

Naloga 4.7.

(a) Zapišimo definicijo funkcije $q(x)$ za $0 \le x \le 4$.

$$\begin{aligned} q(x) &= \max\left(\{0.15x\} \cup \{0.1 \mid x = 1\} \cup \{0.35 \mid x = 2\} \cup \{0.5 \mid x = 3\}\right) \\ &= \begin{cases} 0.35 & x = 2 \text{ (Diskretna d.d.z.)} \\ 0.5 & x = 3 \text{ (Diskretna d.d.z.)} \\ 0.15x & \text{sicer (Zvezna d.z.z.)} \end{cases} \end{aligned}$$

(b) Naj bosta $v_1(x)$ in $v_2(x)$ največji pričakovani vrednosti naložbenih strategij, pri katerih imamo na voljo x milijonov evrov oziroma to količino v celoti vložimo v naložbo in zavarovanje. Zapišimo rekurzivni enačbi.

$$\begin{aligned} v_2(x) &= \max\left\{(x-y)(3+0.4q(y)) \;\middle|\; 0 \le y \le \min\{4,x\}\right\} \\ v_1(x) &= \max\left\{x - y + v_2(y) \;\middle|\; 0 \le y \le x\right\} \end{aligned}$$

(c) Najprej izrazimo $v_2(x)$ glede na vrednost x ($0 \le x \le 50$).

$$\begin{aligned} v_2(x) = \max\Big(&\{3.14(x-2) \mid x \ge 2\} \cup \{3.2(x-3) \mid x \ge 3\} \cup \\ &\left\{(x-y)(3+0.06y) \;\middle|\; 0 \le y \le \min\{4,x\}, y \notin \{2,3\}\right\}\Big) \end{aligned}$$

Naj bo $f(y)$ izraz v zadnjem oklepaju. Opazimo, da gre za kvadraten polinom v y z negativnim vodilnim členom, tako da lahko s pomočjo odvajanja poiščemo njegov maksimum.

$$f(y) = -0.06y^2 + (0.06x - 3)y + 3x$$

$$f'(y) = -0.12y + 0.06x - 3 = 0$$
$$y = x/2 - 25 \leq 0$$

Opazimo, da je maksimum vedno dosežen za vrednost y, ki ni večja od spodnje meje ustreznega intervala, tako da je znotraj njega maksimum dosežen pri $y = 0$ in znaša $f(0) = 3x$ – tj., premije ne plačamo. S primerjavo vseh treh izrazov tako dobimo

$$v_2(x) = \begin{cases} 3x & 0 \leq x \leq 314/7 \text{ (brez zavarovanja)} \\ 3.14(x-2) & 314/7 < x \leq 50 \text{ (Diskretna d.d.z. s premijo 2 mio evrov)} \end{cases}$$

Optimalno strategijo vlaganja sedaj dobimo tako, da izračunamo $v_1(50)$.

$$v_1(50) = \max\big(\{50 + 2y \mid 0 \leq y \leq 314/7\} \cup \{43.72 + 2.14y \mid 314/7 < x \leq 50\}\big)$$

Ker oba izraza naraščata z y, zadostuje preveriti njuni vrednosti pri zgornji meji ustreznega intervala. Tako opazimo, da največjo vrednost dobimo pri $y = 50$. Optimalna strategija vlaganja je torej taka, pri kateri vlagatelj 48 milijonov evrov vloži v naložbo, 2 milijona evrov porabi za premijo pri zavarovalnici Diskretna d.d.z., zase pa ne obdrži ničesar. Pričakovana vrednost naložbene strategije je tako $v^* = v_1(50) = 150.72$ milijonov evrov.

Naloga 4.8.

(a) Naj bo v_i najnižja cena izgradnje počivališč na odseku od začetka avtoceste do lokacije x_i, če zadnje počivališče zgradimo na tej lokaciji. Zapišimo začetne pogoje in rekurzivne enačbe.

$$v_0 = x_0 = 0$$
$$v_i = c_i + \min\big(\{v_j \mid 0 \leq j \leq i-1, x_i - x_j \leq K\} \cup \{\infty\}\big) \quad (1 \leq i \leq n)$$

Vrednosti v_i računamo naraščajoče po indeksu i. Najnižjo ceno izgradnje počivališč na celotnem odseku dobimo kot

$$v^* = \min\big(\{v_j \mid 0 \leq j \leq n, M - x_j \leq K\} \cup \{\infty\}\big).$$

Če velja $v^* = \infty$, potem izgradnja počivališč pod danimi pogoji ni mogoča.

(b) V algoritmu izračunamo n vrednosti v_i, pri čemer pri vsaki obravnavamo največ n možnosti; prav tako obravnavamo največ n možnosti pri računanju v^*. Časovna zahtevnost algoritma je torej $O(n^2)$.

(c) Izračunajmo vrednosti v_i $(1 \leq i \leq 8)$. Ker se lokacije zaporednih počivališč razlikujejo za manj kot K, prav tako pa sta prva in zadnja možna lokacija

oddaljeni manj kot K od začetka oziroma konca odseka, bodo vse vrednosti v_i končne, enako pa velja tudi za v^*.

$$\begin{aligned}
x_1 &= 5 & v_1 &= 18 + \min\{\underline{0}\} &&= 18\\
x_2 &= 12 & v_2 &= 11 + \min\{\underline{0}, 18\} &&= 11\\
x_3 &= 22 & v_3 &= 21 + \min\{\underline{0}, 18, 11\} &&= 21\\
x_4 &= 34 & v_4 &= 16 + \min\{18, \underline{11}, 21\} &&= 27\\
x_5 &= 49 & v_5 &= 23 + \min\{\underline{21}, 27\} &&= 44\\
x_6 &= 65 & v_6 &= 15 + \min\{\underline{44}\} &&= 59\\
x_7 &= 83 & v_7 &= 19 + \min\{\underline{59}\} &&= 78\\
x_8 &= 91 & v_8 &= 13 + \min\{\underline{59}, 78\} &&= 72
\end{aligned}$$

Ker sta samo zadnji dve lokaciji oddaljeni manj kot K od konca odseka, je najmanjša cena izgradnje počivališč enaka $v^* = \min\{78, 72\} = 72$. Rekonstruirajmo še optimalno postavitev.

$$\begin{aligned}
v^* = v_8 &= c_8 + v_6 && \text{počivališče na } x_8 = 91\\
v_6 &= c_6 + v_5 && \text{počivališče na } x_6 = 65\\
v_5 &= c_5 + v_3 && \text{počivališče na } x_5 = 49\\
v_3 &= c_3 + v_0 && \text{počivališče na } x_3 = 22
\end{aligned}$$

Naloga 4.9.

(a) Naj bo p_i najmanjša cena postavitve baznih postaj do i-te milje tako, da se v celoti pokrije interval $[0, i]$. Določimo začetne pogoje in rekurzivne enačbe.

$$\begin{aligned}
p_{-1} &= p_0 = 0\\
p_i &= \min\{a_{i-1} + p_{i-2}, a_i + p_{i-1}\} && (1 \le i \le n)
\end{aligned}$$

Vrednosti p_i računamo naraščajoče po indeksu i $(0 \le i \le n)$. Najmanjšo ceno postavitve baznih postaj dobimo s $p^* = p_n$.

(b) Za izračun vrednosti p_i za posamezen i potrebujemo konstantno mnogo časa. Ker ta izračun opravimo $(n-1)$-krat, je torej časovna zahtevnost ustreznega algoritma $O(n)$.

(c) Izračunajmo vrednosti p_i $(1 \le i \le 10)$.

$$\begin{aligned}
p_1 &= \min\{\underline{4+0}, 6+0\} &&= 4\\
p_2 &= \min\{6+0, \underline{1+4}\} &&= 5\\
p_3 &= \min\{\underline{1+4}, 10+5\} &&= 5
\end{aligned}$$

$$
\begin{aligned}
p_4 &= \min\{\underline{10+5}, 14+5\} &&= 15\\
p_5 &= \min\{\underline{14+5}, 21+15\} &&= 19\\
p_6 &= \min\{21+15, \underline{15+19}\} &&= 34\\
p_7 &= \min\{\underline{15+19}, 6+34\} &&= 34\\
p_8 &= \min\{\underline{6+34}, 10+34\} &&= 40\\
p_9 &= \min\{10+34, \underline{3+40}\} &&= 43\\
p_{10} &= \min\{\underline{3+40}, 2+43\} &&= 43
\end{aligned}
$$

Cena postavitve baznih postaj je torej $p^* = p_{10} = 43$. Poglejmo, kam moramo postaviti bazne postaje.

$$
\begin{aligned}
p_{10} &= a_9 + p_8 && \text{postaja na 9}\\
p_8 &= a_7 + p_6 && \text{postaja na 7}\\
p_6 &= a_6 + p_5 && \text{postaja na 6}\\
p_5 &= a_4 + p_3 && \text{postaja na 4}\\
p_3 &= a_2 + p_1 && \text{postaja na 2}\\
p_1 &= a_0 && \text{postaja na 0}
\end{aligned}
$$

(d) Naj bo q_i najmanjša cena postavitve baznih postaj (pri čemer dovolimo tudi večje postaje) do i-te milje tako, da se v celoti pokrije interval $[0, i]$. Določimo začetne pogoje in rekurzivne enačbe.

$$
\begin{aligned}
b_{-1} &= \infty\\
q_{-3} = q_{-2} = q_{-1} &= q_0 = 0\\
q_i = \min\{&b_{i-2} + \min\{q_{i-4},\ q_{i-3}\},\\
&b_{i-1} + \min\{q_{i-3},\ q_{i-2}\},\\
&b_i + \min\{q_{i-2},\ q_{i-1}\},\\
&a_{i-1} + q_{i-2},\\
&a_i + q_{i-1}\} \qquad (1 \le i \le n)
\end{aligned}
$$

Vrednosti q_i računamo naraščajoče po indeksu i ($0 \le i \le n$). Najmanjšo ceno postavitve baznih postaj dobimo s $q^* = q_n$.

(e) Izračunajmo vrednosti q_i ($1 \le i \le 10$).

$$
\begin{aligned}
q_1 &= \min\{\infty + 0, 10+0, 12+0, \underline{4+0}, 6+0\} &&= 4\\
q_2 &= \min\{10+0, 12+0, \underline{3+0}, 6+0, 1+4\} &&= 3\\
q_3 &= \min\{12+0, \underline{3+0}, 18+3, 1+4, 10+3\} &&= 3\\
q_4 &= \min\{\underline{3+0}, 18+3, 24+3, 10+3, 14+3\} &&= 3\\
q_5 &= \min\{18+3, 24+3, 25+3, \underline{14+3}, 21+3\} &&= 17\\
q_6 &= \min\{24+3, 25+3, \underline{20+3}, 21+3, 15+17\} &&= 23
\end{aligned}
$$

$$
\begin{aligned}
q_7 &= \min\{25+3, \underline{20+3}, 11+17, 15+17, 6+23\} &&= 23\\
q_8 &= \min\{\underline{20+3}, 11+17, 16+23, 6+23, 10+23\} &&= 23\\
q_9 &= \min\{11+17, 16+23, 7+23, 10+23, \underline{3+23}\} &&= 26\\
q_{10} &= \min\{16+23, 7+23, 4+23, \underline{3+23}, 2+26\} &&= 26
\end{aligned}
$$

Cena postavitve baznih postaj je torej $q^* = q_{10} = 26$. Poglejmo, kam moramo postaviti bazne postaje.

$q_{10} = a_9 + q_8$	manjša postaja na 9
$q_8 = b_6 + q_4$	večja postaja na 6
$q_4 = b_2 + q_0$	večja postaja na 2

Naloga 4.10.

(a) Zapišimo začetni pogoj in rekurzivne enačbe.

$$
\begin{aligned}
c_0 &= 0\\
c_i &= c_{i-1} + p_i \quad (1 \le i \le n)
\end{aligned}
$$

Vrednosti c_i $(0 \le i \le n)$ računamo naraščajoče po indeksu i, za izračun pa porabimo $O(n)$ časa.

(b) Naj bo x_{ij} $(1 \le i \le j \le n)$ cena postavitve razdelilnikov, ki razdelijo vodo za delavnice od i do j. Zapišimo začetne pogoje in rekurzivne enačbe.

$$
\begin{aligned}
x_{ii} &= 0 && (1 \le i \le n)\\
x_{ij} &= c_j - c_{i-1} + \min\left\{x_{ik} + x_{k+1,j} \;\middle|\; i \le k < j\right\} && (1 \le i < j \le n)
\end{aligned}
$$

Vrednosti x_{ij} $(1 \le i \le j \le n)$ računamo naraščajoče po razliki $j - i$, nato pa še naraščajoče po indeksu i. Optimalno rešitev dobimo kot $x^* = x_{1n}$. Časovna zahtevnost izračuna je $O(n^3)$.

(c) Najprej izračunajmo vrednosti c_i $(1 \le i \le 6)$.

$$
\begin{aligned}
c_1 &= \;\; 0+4 \;\; &&= 4\\
c_2 &= \;\; 4+19 &&= 23\\
c_3 &= 23+17 &&= 40\\
c_4 &= 40+7 \;\; &&= 47\\
c_5 &= 47+5 \;\; &&= 52\\
c_6 &= 52+9 \;\; &&= 61
\end{aligned}
$$

Sedaj izračunajmo še vrednosti x_{ij} $(1 \le i < j \le 6)$.

$$
x_{12} = 23 - 0 \;\; +0+0 \qquad\qquad = 23
$$

$$
\begin{aligned}
x_{23} &= 40 - 4 \quad + 0 + 0 && = 36\\
x_{34} &= 47 - 23 + 0 + 0 && = 24\\
x_{45} &= 52 - 40 + 0 + 0 && = 12\\
x_{56} &= 61 - 47 + 0 + 0 && = 14\\
x_{13} &= 40 - 0 \quad + \min\{0 + 36, \underline{23 + 0}\} && = 63\\
x_{24} &= 47 - 4 \quad + \min\{\underline{0 + 24}, 36 + 0\} && = 67\\
x_{35} &= 52 - 23 + \min\{\underline{0 + 12}, 24 + 0\} && = 41\\
x_{46} &= 61 - 40 + \min\{0 + 14, \underline{12 + 0}\} && = 33\\
x_{14} &= 47 - 0 \quad + \min\{0 + 67, \underline{23 + 24}, 63 + 0\} && = 94\\
x_{25} &= 52 - 4 \quad + \min\{\underline{0 + 41}, 36 + 12, 67 + 0\} && = 89\\
x_{36} &= 61 - 23 + \min\{\underline{0 + 33}, 24 + 14, 41 + 0\} && = 71\\
x_{15} &= 52 - 0 \quad + \min\{0 + 89, \underline{23 + 41}, 63 + 12, 94 + 0\} && = 116\\
x_{26} &= 61 - 4 \quad + \min\{0 + 71, \underline{36 + 33}, 67 + 14, 89 + 0\} && = 126\\
x_{16} &= 61 - 0 \quad + \min\{0 + 126, \underline{23 + 71}, 63 + 33, 94 + 14, 116 + 0\} && = 155
\end{aligned}
$$

Cena postavitve razdelilnikov je torej $x^* = x_{16} = 155$. Poglejmo, kako naj jih postavimo.

$$
\begin{aligned}
x_{16} &= c_6 - c_0 + x_{12} + x_{36} && \text{delimo na } 1,2 \text{ in } 3 \text{ do } 6\\
x_{12} &= c_2 - c_0 + x_{11} + x_{22} && \text{delimo na } 1 \text{ in } 2\\
x_{36} &= c_6 - c_2 + x_{33} + x_{46} && \text{delimo na } 3 \text{ in } 4 \text{ do } 6\\
x_{46} &= c_6 - c_3 + x_{45} + x_{66} && \text{delimo na } 4,5 \text{ in } 6\\
x_{45} &= c_5 - c_3 + x_{44} + x_{55} && \text{delimo na } 4 \text{ in } 5
\end{aligned}
$$

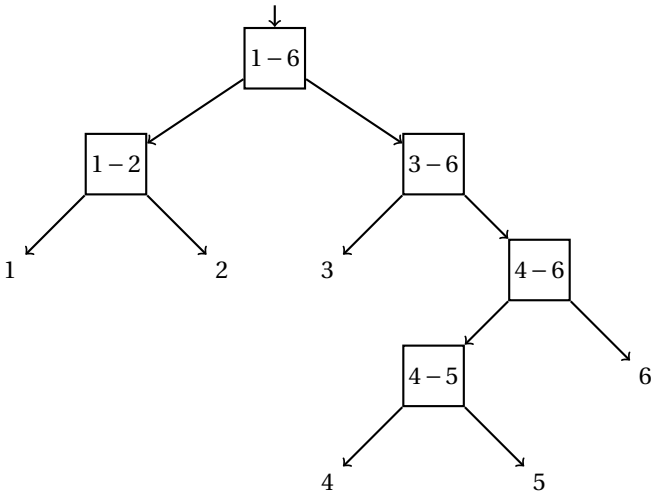
Postavitev je prikazana na sliki 35.

Naloga 4.11.

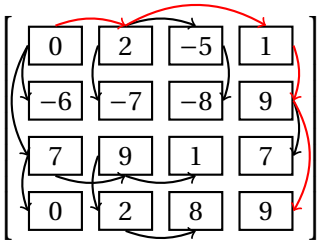
(a) Naj bo p_{ij} največja vsota, ki jo lahko dosežemo z zaporedjem skokov od A_{11} do A_{ij}. Določimo začetni pogoj in rekurzivne enačbe.

$$
\begin{aligned}
p_{11} &= A_{11}\\
p_{1j} &= A_{1j} + \max\left\{p_{1\ell} \;\middle|\; 1 \le \ell \le j-1\right\} && (2 \le j \le n)\\
p_{i1} &= A_{i1} + \max\left\{p_{\ell 1} \;\middle|\; 1 \le \ell \le i-1\right\} && (2 \le i \le m)\\
p_{ij} &= A_{ij} + \max(\left\{p_{i\ell} \;\middle|\; 1 \le \ell \le j-1\right\} \cup \left\{p_{\ell j} \;\middle|\; 1 \le \ell \le i-1\right\}) && (2 \le i \le m,\\
& && \quad 2 \le j \le n)
\end{aligned}
$$

Vrednosti p_{ij} računamo v leksikografskem vrstnem redu indeksov (npr. najprej naraščajoče po i, nato pa naraščajoče po j). Maksimalno vsoto obiskanih mest dobimo kot $p^* = p_{mn}$. Rekurzivne enačbe lahko rešimo v času $O(mn(m+n))$.



Slika 35: Postavitev razdelilnikov za nalogo 4.10(c).



Slika 36: Reševanje in optimalna rešitev za nalogo 4.11(b).

(b) Matrika vrednosti $(p_{ij})_{i,j=1}^{4}$ je prikazana na sliki 36, pri čemer je pri vsaki vrednosti puščica od tiste vrednosti, ki je bila uporabljena pri njenem izračunu. Z rdečo barvo je prikazano zaporedje $(1,1),(1,2),(1,4),(2,4),(4,4)$ z največjo vsoto $p^* = 9$.

Naloga 4.12.

(a) Naj bo d_i $(i = 1,2)$ optimalni dobiček pri vlaganju v i-to podjetje. Potem velja

$$d_i = \begin{cases} \max\{(n_i + k_i x)(n_3 + k_3(m-x)) \mid a_i \le x \le m - a_3\}, & \text{če } a_i \le m - a_3, \text{ in} \\ 0 & \text{sicer.} \end{cases}$$

Optimalni dobiček potem dobimo kot $d^* = \max\{d_1, d_2\}$.

(b) Naj bo $q_i(x)$ $(i = 1,2)$ kvadratni polinom, ki nastopa v zgornjem izrazu. Da izračunamo d_1 in d_2, bomo poiskali maksimum polinomov q_1 in q_2 v ustreznih mejah.

Vrednost d_1 je maksimalna vrednost izraza

$$q_1(x) = (8+4x)(4+10-x) = -4x^2+48x+112$$

za $x \in [4,7]$. Ker ima polinom negativen vodilni člen, je njegov globalni maksimum tam, kjer ima njegov odvod ničlo.

$$\begin{aligned} q_1'(x) = -8x+48 &= 0 \\ x &= 6 \end{aligned}$$

Ker maksimum leži na iskanem intervalu, velja $d_1 = q_1(6) = 256$.

Vrednost d_2 je maksimalna vrednost izraza

$$q_2(x) = (12+2x)(4+10-x) = -2x^2+16x+168$$

za $x \in [5,7]$. Ker ima polinom negativen vodilni člen, je njegov globalni maksimum tam, kjer ima njegov odvod ničlo.

$$\begin{aligned} q_2'(x) = -4x+16 &= 0 \\ x &= 4 \end{aligned}$$

Ker maksimum leži levo od iskanega intervala, leži maksimum na njegovem levem robu, torej velja $d_2 = q_2(5) = 198$.

Optimalni dobiček je tako $d^* = \max\{256, 198\} = 256$ in ga dosežemo tako, da vložimo 6 v prvo podjetje in 4 v marketing.

2.5 Algoritmi na grafih

Naloga 5.1.
Sledili bomo naslednjemu algoritmu.

function BFS($G = (V, E), S,$ VISIT)
  visited ← slovar z vrednostjo FALSE za vsak $v \in V$
  pred ← slovar z vrednostjo NULL za vsak $v \in V$
  for $s \in S$ **do**
    if $\neg$*visited*$[s]$ **then**
      visited$[s]$ ← TRUE
      $Q \leftarrow [s]$
      while $\neg Q$.isEmpty() **do**
        $u \leftarrow Q$.pop()
        VISIT$(u,$ *pred*$[u])$
        for $v \in$ ADJ(G, u) **do**
          if $\neg$*visited*$[v]$ **then**
            visited$[v]$ ← TRUE
            pred$[v] \leftarrow u$
            Q.append(v)
          end if
        end for
      end while
    end if
  end for
  return *pred*
end function

Časovna zahtevnost preiskovanja je $O(m+n)$ (kjer je n število vozlišč in m število povezav grafa), poleg tega pa algoritem opravi še $O(n)$ klicev funkcije VISIT.

Potek zgornjega algoritma na grafu s slike 9 (pri čemer za množico začetnih vozlišč S vzamemo množico vseh vozlišč V, za VISIT pa vzamemo NOOP, torej ne naredimo ničesar), pri čemer sledimo abecednemu vrstnemu redu vozlišč, je prikazan v tabeli 9. Gozd iskanja v širino je prikazan na sliki 37, iz katere je razvidno, da so drevesne povezave

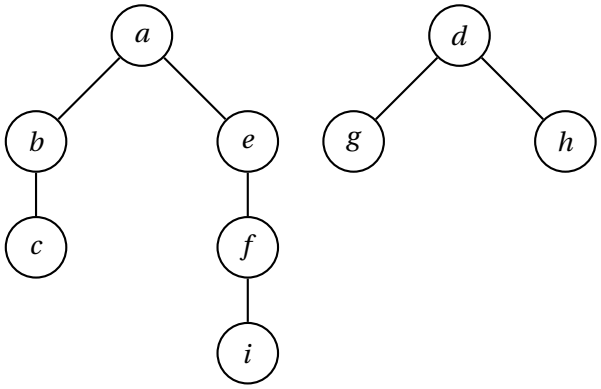
$$(a,b),(b,c),(a,e),(e,f),(f,i),(d,g),(d,h).$$

Naloga 5.2.
Sledili bomo naslednjemu algoritmu. Predpostavili bomo, da imamo na voljo podatkovno strukturo za prednostno vrsto, ki za vsak element hrani njegovo prioriteto (to lahko tudi spreminjamo). Podatkovna struktura ima metodo pop(), ki vrne in odstrani element z najmanjšo prioriteto skupaj z njegovo prioriteto. Tukaj bodo elementi vozlišča grafa, prioritete pa dolžine najkrajših najdenih poti od začetnega vozlišča.

function DIJKSTRA$(G = (V, E), s \in V, L : E \to \mathbb{R}_+)$
  $d \leftarrow$ slovar z vrednostjo ∞ za vsako vozlišče $v \in V$
  pred ← slovar z vrednostjo NULL za vsako vozlišče $v \in V$

s	u	v	Q	množica označenih vozlišč
a			$[a]$	$\{a\}$
a	a	b	$[b]$	$\{a,b\}$
a	a	e	$[b,e]$	$\{a,b,e\}$
a	b	c	$[e,c]$	$\{a,b,c,e\}$
a	b	e		$\{a,b,c,e\}$
a	e	a	$[c]$	$\{a,b,c,e\}$
a	e	b		$\{a,b,c,e\}$
a	e	f	$[c,f]$	$\{a,b,c,e,f\}$
a	c	b	$[f]$	$\{a,b,c,e,f\}$
a	c	f		$\{a,b,c,e,f\}$
a	f	c		$\{a,b,c,e,f\}$
a	f	e		$\{a,b,c,e,f\}$
a	f	i	$[i]$	$\{a,b,c,e,f,i\}$
a	i	f	$[\,]$	$\{a,b,c,e,f,i\}$
b				$\{a,b,c,e,f,i\}$
c				$\{a,b,c,e,f,i\}$
d			$[d]$	$\{a,b,c,d,e,f,i\}$
d	d	g	$[g]$	$\{a,b,c,d,e,f,g,i\}$
d	d	h	$[g,h]$	$\{a,b,c,d,e,f,g,h,i\}$
d	g	d	$[h]$	$\{a,b,c,d,e,f,g,h,i\}$
d	g	h		$\{a,b,c,d,e,f,g,h,i\}$
d	h	d		$\{a,b,c,d,e,f,g,h,i\}$
d	h	g	$[\,]$	$\{a,b,c,d,e,f,g,h,i\}$
e				$\{a,b,c,d,e,f,g,h,i\}$
f				$\{a,b,c,d,e,f,g,h,i\}$
g				$\{a,b,c,d,e,f,g,h,i\}$
h				$\{a,b,c,d,e,f,g,h,i\}$
i				$\{a,b,c,d,e,f,g,h,i\}$

Tabela 9: Potek izvajanja algoritma za nalogo 5.1.



Slika 37: Gozd iskanja v širino za nalogo 5.1.

$Q \leftarrow$ prednostna vrsta s prioriteto ∞ za vsako vozlišče $v \in V$ (na vrhu je vozlišče z najmanjšo prioriteto)
$Q[s] \leftarrow 0$
while $\neg Q$.isEmpty() **do**
$u, d[u] \leftarrow Q$.pop()
for $v \in$ ADJ(G, u) **do**
if $v \in Q \wedge Q[v] > d[u] + L_{uv}$ **then**
$Q[v] \leftarrow d[u] + L_{uv}$
$pred[v] \leftarrow u$
end if
end for
end while
return $(d, pred)$
end function

Če za prednostno vrsto uporabimo običajen slovar, je časovna zahtevnost metode pop() linearna v številu elementov slovarja, spreminjanje vrednosti v slovarje pa se opravi v konstantnem času. Časovna zahtevnost zgornjega algoritma je tako $O(n^2)$, kjer je n število vozlišč v grafu.

Če pa za prednostno vrsto vzamemo kopico, sta časovni zahtevnosti metode pop() in spreminjanja vrednosti v kopici logaritemski v velikosti kopice. Časovna zahtevnost zgornjega algoritma je v tem primeru $O((m+n)\log n)$, kjer je m število povezav v grafu.

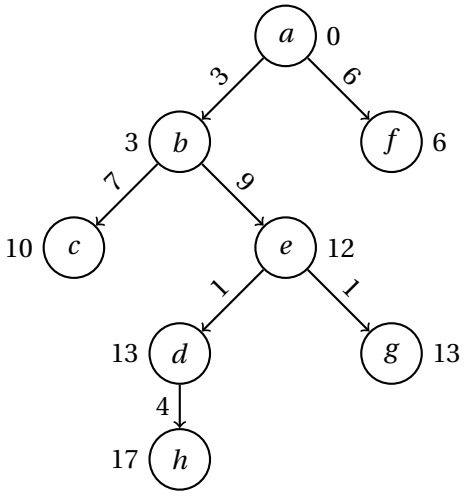
Potek zgornjega algoritma na grafu s slike 10, pri čemer sledimo abecednemu vrstnemu redu vozlišč, je prikazan v tabeli 10. Drevo najkrajših poti je prikazano na sliki 38.

Naloga 5.3.
Sledili bomo naslednjemu algoritmu.

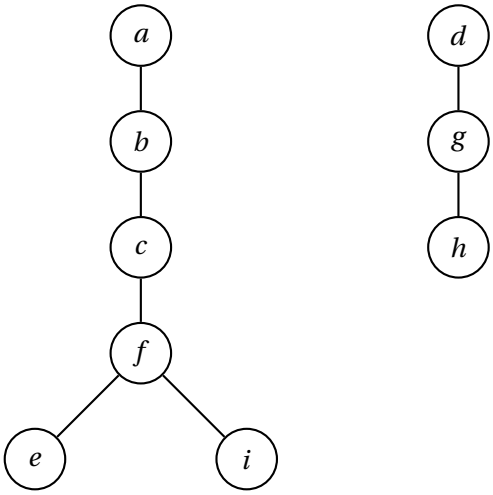
function DFS$(G = (V, E), S,$ PREVISIT$,$ POSTVISIT$)$
for $v \in V$ **do**
$visited[v] \leftarrow$ FALSE
end for

u	v	Q	razdalje in predhodniki
		$[(a,0)]$	$[0,\infty,\infty,\infty,\infty,\infty,\infty,\infty]$
a	b	$[(b,3)]$	$[0,3_a,\infty,\infty,\infty,\infty,\infty,\infty]$
a	f	$[(b,3),(f,6)]$	$[0,3_a,\infty,\infty,\infty,6_a,\infty,\infty]$
b	c	$[(f,6),(c,10)]$	$[0,3_a,10_b,\infty,\infty,6_a,\infty,\infty]$
b	e	$[(f,6),(c,10),(e,12)]$	$[0,3_a,10_b,\infty,12_b,6_a,\infty,\infty]$
f	g	$[(c,10),(e,12),(g,14)]$	$[0,3_a,10_b,\infty,12_b,6_a,14_g,\infty]$
c	d	$[(e,12),(g,14),(d,17)]$	$[0,3_a,10_b,17_c,12_b,6_a,14_g,\infty]$
e	d	$[(d,13),(g,14),(d,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,14_g,\infty]$
e	g	$[(g,13),(d,13),(g,14),(d,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,\infty]$
g	f	$[(d,13),(g,14),(d,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,\infty]$
d	b	$[(g,14),(d,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,\infty]$
d	h	$[(g,14),(d,17),(h,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,17_d]$
g	f	$[(d,17),(h,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,17_d]$
d	b	$[(h,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,17_d]$
d	h	$[(h,17)]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,17_d]$
h	g	$[\,]$	$[0,3_a,10_b,13_e,12_b,6_a,13_e,17_d]$

Tabela 10: Potek izvajanja algoritma za nalogo 5.2.



Slika 38: Drevo najkrajših poti za nalogo 5.2.



Slika 39: Gozd iskanja v globino za nalogo 5.3.

```
function RAZIŠČI(v, w)
    if visited[v] then
        return
    end if
    visited[v] ← TRUE
    PREVISIT(v, w)
    for u ∈ ADJ(G, v) do:
        RAZIŠČI(u, v)
    end for
    POSTVISIT(v, w)
end function
for v ∈ S do
    RAZIŠČI(v, NULL)
end for
end function
```

Časovna zahtevnost preiskovanja je $O(m+n)$ (kjer je n število vozlišč in m število povezav grafa), poleg tega pa algoritem opravi še $O(n)$ klicev funkcij PREVISIT in POSTVISIT.

Potek zgornjega algoritma na grafu s slike 9 (pri čemer za množico začetnih vozlišč S vzamemo množico vseh vozlišč V, za PREVISIT in POSTVISIT pa vzamemo NOOP, torej ne naredimo ničesar), pri čemer sledimo abecednemu vrstnemu redu vozlišč, je prikazan v tabeli 11. Gozd iskanja v globino je prikazan na sliki 39, iz katere je razvidno, da so drevesne povezave

$$(a,b),(b,c),(c,f),(f,e),(f,i),(d,g),(g,h).$$

Naloga 5.4.
Sledili bomo naslednjemu algoritmu.

v	u	množica označenih vozlišč
a		$\{a\}$
a	b	$\{a,b\}$
b	a	$\{a,b\}$
b	c	$\{a,b,c\}$
c	b	$\{a,b,c\}$
c	f	$\{a,b,c,f\}$
f	c	$\{a,b,c,f\}$
f	e	$\{a,b,c,e,f\}$
e	a	$\{a,b,c,e,f\}$
e	b	$\{a,b,c,e,f\}$
e	f	$\{a,b,c,e,f\}$
f	i	$\{a,b,c,e,f,i\}$
i	f	$\{a,b,c,e,f,i\}$
b	e	$\{a,b,c,e,f,i\}$
a	e	$\{a,b,c,e,f,i\}$
b		$\{a,b,c,e,f,i\}$
c		$\{a,b,c,e,f,i\}$
d		$\{a,b,c,d,e,f,i\}$
d	g	$\{a,b,c,d,e,f,g,i\}$
g	d	$\{a,b,c,d,e,f,g,i\}$
g	h	$\{a,b,c,d,e,f,g,h,i\}$
h	d	$\{a,b,c,d,e,f,g,h,i\}$
h	g	$\{a,b,c,d,e,f,g,h,i\}$
d	h	$\{a,b,c,d,e,f,g,h,i\}$

Tabela 11: Potek izvajanja algoritma za nalogo 5.3.

```
function BellmanFord(G = (V, E), s ∈ V, L : E → ℝ)
    d ← slovar z vrednostjo ∞ za vsako vozlišče v ∈ V
    pred ← slovar z vrednostjo Null za vsako vozlišče v ∈ V
    d[s] ← 0
    i ← 0
    trenutna ← {s}
    while ¬trenutna.isEmpty() do
        i ← i + 1
        if i > |V| then
            return graf ima negativen cikel
        end if
        naslednja ← prazna množica
        for u ∈ trenutna do
            for v ∈ Adj(G, u) do
                if d[v] > d[u] + L_uv then
                    d[v] ← d[u] + L_uv
                    pred[v] ← u
                    naslednja.add(v)
                end if
            end for
        end for
        trenutna ← naslednja
    end while
    return (d, pred)
end function
```

V i-tem obhodu zanke **while** slovar d vsebuje dolžine tistih najkrajših poti od s do vsakega drugega vozlišča, ki vsebujejo največ i povezav. Če graf nima negativnih ciklov, je v vsaki najkrajši poti največ $n-1$ povezav (kjer je n število vozlišč) in zato algoritem konča najkasneje po n-tem obhodu zanke **while**. V nasprotnem primeru obstaja najkrajša pot z neskončno povezavami, kar algoritem zazna ob vstopu v $(n+1)$-vi obhod zanke. Časovna zahtevnost algoritma je tako $O(mn)$, kjer je m število povezav grafa.

Potek zgornjega algoritma na grafu s slike 11 je prikazan v tabeli 12. Drevo najkrajših poti je prikazano na sliki 40.

Naloga 5.5.

(a) Upoštevamo, da je graf $G = (V, E)$ acikličen, torej za vsaki vozlišči $u, v \in V$ velja

$$u \to v \implies \mathit{postlabel}(u) > \mathit{postlabel}(v),$$

kjer je *postlabel* števec, s katerim označimo vozlišče, ko ga v algoritmu DFS docela preiščemo.

Trditev lahko dokažemo z obravnavanjem dveh primerov, in sicer, ko z DFS pridemo v vozlišče u pred v, in ko pridemo v vozlišče v pred u. V prvem primeru bomo pred določitvijo pooznake vozlišču u morali obiskati vse njegove sosede, torej tudi v, ki mu bomo določili pooznako pred u. V drugem

i	u	v	L_{uv}	razdalje in predhodniki
				$[0,\infty,\infty,\infty,\infty,\infty,\infty,\infty]$
1	a	b	3	$[0,3_a,\infty,\infty,\infty,\infty,\infty,\infty]$
1	a	f	6	$[0,3_a,\infty,\infty,\infty,6_a,\infty,\infty]$
2	b	c	7	$[0,3_a,10_b,\infty,\infty,6_a,\infty,\infty]$
2	b	e	9	$[0,3_a,10_b,\infty,12_b,6_a,\infty,\infty]$
2	f	g	8	$[0,3_a,10_b,\infty,12_b,6_a,14_f,\infty]$
3	c	d	-7	$[0,3_a,10_b,3_c,12_b,6_a,14_f,\infty]$
3	e	d	1	$[0,3_a,10_b,3_c,12_b,6_a,14_f,\infty]$
3	e	g	1	$[0,3_a,10_b,3_c,12_b,6_a,13_e,\infty]$
3	g	f	-1	$[0,3_a,10_b,3_c,12_b,6_a,13_e,\infty]$
4	d	b	9	$[0,3_a,10_b,3_c,12_b,6_a,13_e,\infty]$
4	d	h	4	$[0,3_a,10_b,3_c,12_b,6_a,13_e,7_d]$
4	g	f	-1	$[0,3_a,10_b,3_c,12_b,6_a,13_e,7_d]$
5	h	g	-5	$[0,3_a,10_b,3_c,12_b,6_a,2_h,7_d]$
6	g	f	-1	$[0,3_a,10_b,3_c,12_b,1_g,2_h,7_d]$
7	f	g	8	$[0,3_a,10_b,3_c,12_b,1_g,2_h,7_d]$

Tabela 12: Potek izvajanja algoritma za nalogo 5.4.

primeru pridemo prej do v, a ker je graf acikličen, od v ne bomo prišli do u, torej bomo spet v obdelali prej.

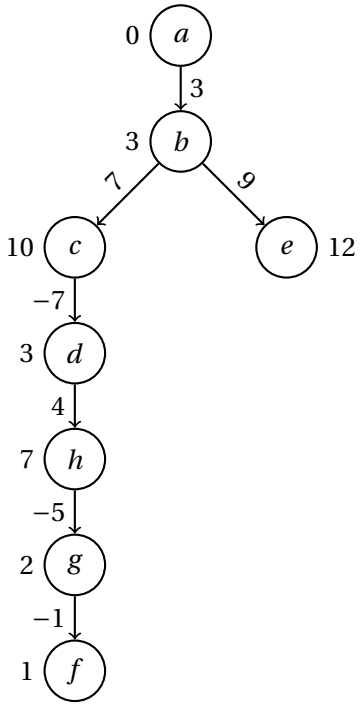
Trditev nam omogoča zapis krajšega algoritma, ki bo uporabil algoritem DFS iz naloge 5.3. Njegova ideja je, da padajoče uredimo vozlišča po njihovih pooznakah in tako dobimo inverzno topološko ureditev. To dosežemo tako, da po popolni obdelavi vozlišča tega postavimo na sklad.

```
function TOPO(G = (V, E))
    toposklad ← []
    function POSTVISIT(u, v)
        toposklad.append(u)
    end function
    DFS(G, V, NOOP, POSTVISIT)
    toposklad.reverse()
    return toposklad
end function
```

Časovna zahtevnost algoritma je $O(m+n)$, kjer je n število vozlišč in m število povezav grafa.

Klic TOPO(G) nam vrne ureditev vozlišč $[g,a,h,b,c,f,d,e]$.

(b) Tukaj bomo uporabili rezultat iz prejšnje točke in osnoven koncept dinamičnega programiranja.



Slika 40: Drevo najkrajših poti za nalogo 5.4.

Označimo z L_{uv} utež povezave $u \to v$. Naj $d_G(u)$ predstavlja najkrajšo razdaljo od izbranega vozlišča s do vozlišča u.

$$d_G(s) = 0$$
$$d_G(u) = \min_{v \to u}(d_G(v) + L_{vu}) \qquad (u \in V \setminus \{s\})$$

Da bomo imeli vse potrebne $d_G(v)$ definirane pred izračunom $d_G(u)$, poskrbimo z rezultatom iz prejšnje točke, saj pri topološki ureditvi vodijo vse povezave le naprej, Po premiku na naslednje vozlišče tako ne potrebujemo kasnejših vozlišč, pač pa le prejšnja, za katera vrednost že poznamo.

Imamo torej vse, kar potrebujemo za izračun najkrajše poti. Za beleženje vozlišč, skozi katera vodi ta pot, bomo uporabili seznam predhodnikov, ki bo nakazoval, katero vozlišče smo izbrali za predhodnika.

$$pred(s) = \text{NULL}$$
$$pred(u) = \arg\min_{v \to u}(d_G(v) + L_{vu}) \qquad (u \in V \setminus \{s\})$$

Zapišimo algoritem, ki poišče razdalje od s do ostalih vozlišč. Poleg razdalj bo za vsako vozlišče povedal še, iz katerega vozlišča smo prišli do njega.

```
function NajkrajšaPot(G = (V, E), s, L : E → ℝ)
    d_G ← slovar z vrednostjo ∞ za vsako vozlišče v ∈ V
    d_G[s] ← 0
    pred ← slovar z vrednostjo Null za vsako vozlišče v ∈ V
    for u ∈ Topo(G) do
        for v ∈ Adj(G, u) do
            if d_G[v] > d_G[u] + L_uv then
                d_G[v] ← d_G[u] + L_uv
                pred[v] ← u
            end if
        end for
    end for
    return (d_G, pred)
end function
```

Časovna zahtevnost algoritma je $O(m+n)$, kjer je n število vozlišč in m število povezav grafa.

Če želimo rekonstruirati pot, sledimo predhodnikom podanega končnega vozlišča t, dokler ne pridemo do s.

```
function RekonstruirajPot(pred, t)
    u ← t
    pot ← [t]
    while pred[u] ≠ Null do
        u ← pred[u]
        pot.append(u)
    end while
    pot.reverse()
    return pot
end function
```

Klic NajkrajšaPot(G, g, L) vrne seznam d_G z razdaljami od vozlišča g do ostalih vozlišč grafa in seznam *pred* s prednikom vsakega vozlišča v najkrajši poti od g. Potek izvajanja algoritma je prikazan v tabeli 13. Seznam vozlišč, skozi katera je algoritem potoval, da je opravil pot od g do e, nato dobimo s klicem RekonstruirajPot$(pred, e)$ – to so $[g, a, b, e]$. Dolžino te poti lahko preberemo kot $d_G[e] = -1$.

Pripomnimo, da algoritem deluje pravilno ne glede na položaj začetnega vozlišča s v topološki ureditvi. Za vsako vozlišče u, ki v topološki ureditvi nastopa pred s, namreč velja $d_G(u) = \infty$, tako da se ne nastavi kot predhodnik nobenemu vozlišču. Tako se vozlišču s ne bo nastavil predhodnik. Algoritem lahko nekoliko pohitrimo tako, da gledamo vozlišča v topološki ureditvi le od s naprej, prejšnja pa ignoriramo, saj iz s ne moremo priti do njih. Prav tako bi se lahko ustavili, ko dosežemo vozlišče t.

(c) Uporabimo algoritem NajdaljšaPot, ki deluje na enak način kot NajkrajšaPot, le da namesto ∞ vzame v d_G za začetne vrednosti $-\infty$, v zanki pa na vsakem koraku preverja pogoj $d_G[v] < d_G[u] + L_{uv}$. Časovna zahtevnost algoritma tako ostane enaka. Klic RekonstruirajPot$(pred, e)$, kjer je d_G, *pred*

izhod klica NAJDALJŠAPOT(G, g, L), vrne pot $[g, h, b, c, f, e]$ dolžine 24. Potek izvajanja algoritma je prikazan v tabeli 14.

Naloga 5.6.

(a) Iz danega neusmerjenega grafa $G = (V, E)$ in funkcije uteži vozlišč $c : V \to [0, 1]$ bomo konstruirali usmerjen graf $G' = (V, E')$, kjer je $E' = \{uv, vu \mid uv \in E\}$ (tj., vsako neusmerjeno povezavo iz G nadomestimo z nasprotno usmerjenima povezavama med krajiščema), in funkcijo uteži povezav $L : E' \to [0, 1]$ s predpisom $L(uv) = c(u)$ (tj., cena povezave je enaka ceni začetnega vozlišča v G). Na grafu G' s funkcijo uteži povezav L nato poženemo varianto algoritma DIJKSTRA iz naloge 5.2.

```
function DijkstraProb(G = (V, E), s ∈ V, L : E → [0, 1])
    d ← slovar z vrednostjo 0 za vsako vozlišče v ∈ V
    pred ← slovar z vrednostjo Null za vsako vozlišče v ∈ V
    Q ← prednostna vrsta s prioriteto 0 za vsako vozlišče v ∈ V (na vrhu je vozlišče
z največjo prioriteto)
    Q[s] ← 1
    while ¬Q.isEmpty() do
        u, d[u] ← Q.pop()
        for v ∈ Adj(G, u) do
            if v ∈ Q ∧ Q[v] < d[u] · L_uv then
                Q[v] ← d[u] · L_uv
                pred[v] ← u
            end if
        end for
    end while
    return (d, pred)
end function
```

V primerjavi z algoritmom DIJKSTRA smo torej zamenjali seštevanje z množenjem in obrnili primerjave, poleg tega pa smo še ustrezno spremenili začetne vrednosti. Preslikava $x \mapsto -\log x$ pošlje uteži na interval $[0, \infty]$, pri čemer se urejenost obrne, množenje se pa nadomesti s seštevanjem – zgornji algoritem je torej ekvivalenten algoritmu DIJKSTRA s tako preslikanimi utežmi, in ima tako isto časovno zahtevnost (tj., $O(n^2)$, če za prednostno vrsto uporabimo običajen slovar, in $O(m \log n)$, če za prednostno vrsto vzamemo kopico, kjer je n število vozlišč in m število povezav v grafu).

(b) Potek izvajanja zgornjega algoritma je prikazan v tabeli 15, iz katere razberemo, da je najvarnejša pot LA – SF – RE – SLC – DE – KC – SL – CH, z verjetnostjo, da nas ne oropajo, enako 0.31752.

							d_G								
u	v	$d_G(v)$	$\overset{?}{>}$	$d_G(u)$	$+$	L_{uv}	a	b	c	d	e	f	g	h	$pred[v]$
g	a	∞	$>$	0	$+$	(-1)	-1	∞	∞	∞	∞	∞	0	∞	g
g	d	∞	$>$	0	$+$	6	-1	∞	∞	6	∞	∞	0	∞	g
g	f	∞	$>$	0	$+$	6	-1	∞	∞	6	∞	6	0	∞	g
g	h	∞	$>$	0	$+$	6	-1	∞	∞	6	∞	6	0	6	g
a	b	∞	$>$	-1	$+$	2	-1	1	∞	6	∞	6	0	6	a
a	c	∞	$>$	-1	$+$	(-2)	-1	1	-3	6	∞	6	0	6	a
a	h	6	$>$	-1	$+$	0	-1	1	-3	6	∞	6	0	-1	a
h	b	1	$\not>$	-1	$+$	3	-1	1	-3	6	∞	6	0	-1	a
h	f	6	$>$	-1	$+$	6	-1	1	-3	6	∞	5	0	-1	h
b	c	-3	$\not>$	1	$+$	6	-1	1	-3	6	∞	5	0	-1	a
b	e	∞	$>$	1	$+$	(-2)	-1	1	-3	6	-1	5	0	-1	b
c	d	6	$>$	-3	$+$	2	-1	1	-3	-1	-1	5	0	-1	c
c	f	5	$>$	-3	$+$	6	-1	1	-3	-1	-1	3	0	-1	c
f	e	-1	$\not>$	3	$+$	6	-1	1	-3	-1	-1	3	0	-1	b
d	e	-1	$\not>$	-1	$+$	7	-1	1	-3	-1	-1	3	0	-1	b

Tabela 13: Potek izvajanja algoritma NAJKRAJŠAPOT za nalogo 5.5(b).

							d_G								
u	v	$d_G(v)$	$\overset{?}{<}$	$d_G(u)$	$+$	L_{uv}	a	b	c	d	e	f	g	h	$pred[v]$
g	a	$-\infty$	$<$	0	$+$	(-1)	-1	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	0	$-\infty$	g
g	d	$-\infty$	$<$	0	$+$	6	-1	$-\infty$	$-\infty$	6	$-\infty$	$-\infty$	0	$-\infty$	g
g	f	$-\infty$	$<$	0	$+$	6	-1	$-\infty$	$-\infty$	6	$-\infty$	6	0	$-\infty$	g
g	h	$-\infty$	$<$	0	$+$	6	-1	$-\infty$	$-\infty$	6	$-\infty$	6	0	6	g
a	b	$-\infty$	$<$	-1	$+$	2	-1	1	$-\infty$	6	$-\infty$	6	0	6	a
a	c	$-\infty$	$<$	-1	$+$	(-2)	-1	1	-3	6	$-\infty$	6	0	6	a
a	h	6	$\not<$	-1	$+$	0	-1	1	-3	6	$-\infty$	6	0	6	g
h	b	1	$<$	6	$+$	3	-1	9	-3	6	$-\infty$	6	0	6	h
h	f	6	$<$	6	$+$	6	-1	9	-3	6	$-\infty$	12	0	6	h
b	c	-3	$<$	9	$+$	6	-1	9	15	6	$-\infty$	12	0	6	b
b	e	$-\infty$	$<$	9	$+$	(-2)	-1	9	15	6	7	12	0	6	b
c	d	6	$<$	15	$+$	2	-1	9	15	17	7	12	0	6	c
c	f	12	$<$	15	$+$	6	-1	9	15	17	7	21	0	6	c
f	e	7	$<$	21	$+$	6	-1	9	15	17	27	21	0	6	f
d	e	27	$\not<$	17	$+$	7	-1	9	15	17	27	21	0	6	f

Tabela 14: Potek izvajanja algoritma NAJDALJŠAPOT za nalogo 5.5(c).

LA	SF	PH	LV	RE	EP	AQ	DE	SLC	DA	OC	KC	OM	ME	SL	CH
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
*	1_{LA}	1_{LA}	1_{LA}												
	*			1_{SF}											
		*			0.8_{PH}	0.8_{PH}									
			*				0.5_{LV}	0.5_{LV}							
				*				0.7_{RE}							
					*				0.48_{EP}						
						*	0.56_{AQ}			0.56_{AQ}					
							0.63_{SLC}	*							
							*				0.504_{DE}	0.504_{DE}			
										*			0.336_{OC}		
											*			0.3528_{KC}	
												*			0.2016_{OM}
									*				0.384_{DA}		
													*		
														*	0.31752_{SL}
															*

Tabela 15: Potek izvajanja algoritma za nalogo 5.6(b).

Naloga 5.7.

(a) Konstruiramo graf G, katerega vozlišča so kraji z našega seznama, med vozliščema pa je povezava, če lahko cesto med ustreznima krajema prevozimo v enem dnevu. Če vozlišči s in t predstavljata začetno točko in destinacijo, potem lahko na grafu G izvedemo iskanje v širino z začetkom v vozlišču s ter v dobljenem drevesu poiščemo pot do vozlišča t.

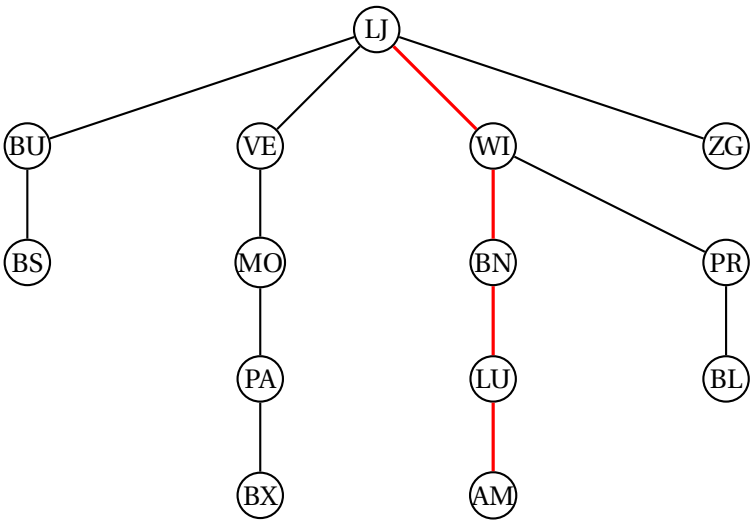
(b) Iz grafa G izpeljemo graf G', ki ga dobimo tako, da odstranimo notranja vozlišča poti iz prejšnje točke (tj., s in t pustimo v G'). Potem lahko na grafu G' izvedemo iskanje v širino z začetkom v vozlišču t ter v dobljenem drevesu poiščemo pot do vozlišča s.

(c) Iskanje v širino na grafu G s slike 14 nam da drevo s slike 41, s katere je razvidna pot LJ – WI – BN – LU – AM. Graf G' torej dobimo tako, da iz G odstranimo vozlišča WI, BN in LU. Iskanje v širino nam da drevo s slike 42, s katere je razvidna povratna pot AM – BL – PR – BS – BU – LJ.

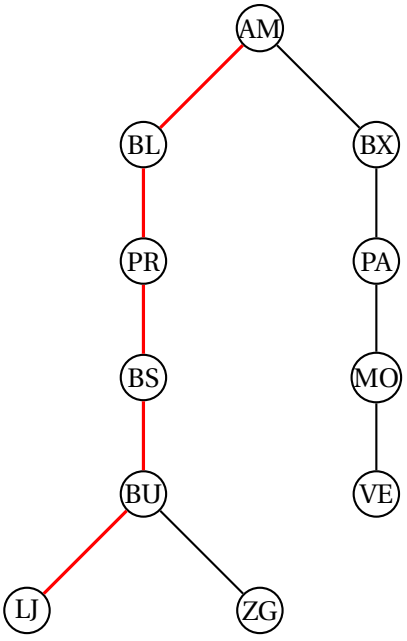
Naloga 5.8.

(a) Drevo iskanja v globino je prikazano na sliki 43, pri čemer sta pri vsakem vozlišču u prikazani še vrednosti (ℓ_u, p_u).

Prerezna vozlišča grafa s slike 15 so a, b in c. Koren drevesa iskanja v globino a je prerezno vozlišče, ker ima več kot enega naslednika. Ostala prerezna vozlišča prepoznamo po tem, da imajo takega naslednika v drevesu iskanja



Slika 41: Drevo iskanja v širino na grafu G za nalogo 5.7.



Slika 42: Drevo iskanja v širino na grafu G' za nalogo 5.7.

v globino, da v njegovem poddrevesu ni vozlišč s prečnimi povezavami do predhodnikov obravnavanega vozlišča – drugače povedano, notranje vozlišče u v drevesu iskanja v globino je prerezno vozlišče natanko tedaj, ko obstaja njegov naslednik w, za katerega velja $p_w \geq \ell_u$.

(b) Zapišimo rekurzivno formulo.

$$p_u = \min\left(\{p_w \mid w \in V, u \rightarrow w\} \cup \{\ell_w \mid w \in V, u \sim w \vee u = w\}\right)$$

(c) Zapišimo psevdokodo za funkcijo PREVISIT.

```
function PREVISIT(u, v)
    if v = NULL then
        ℓ_u ← 0
    else
        ℓ_u ← ℓ_v + 1
    end if
end function
```

(d) Zapišimo psevdokodo za funkcijo POSTVISIT.

```
function POSTVISIT(u, v)
    if (v = NULL ∧ |ADJ(G, u)| > 1) ∨ (v ≠ NULL ∧ ∃w ∈ ADJ(G, u) : p_w ≤ ℓ_u) then
        izhod.append(u)
    end if
    p_u ← min({p_w | w ∈ ADJ(G, u) ∧ ℓ_w > ℓ_u} ∪ {ℓ_w | w ∈ ADJ(G, u) ∨ w = u})
end function
```

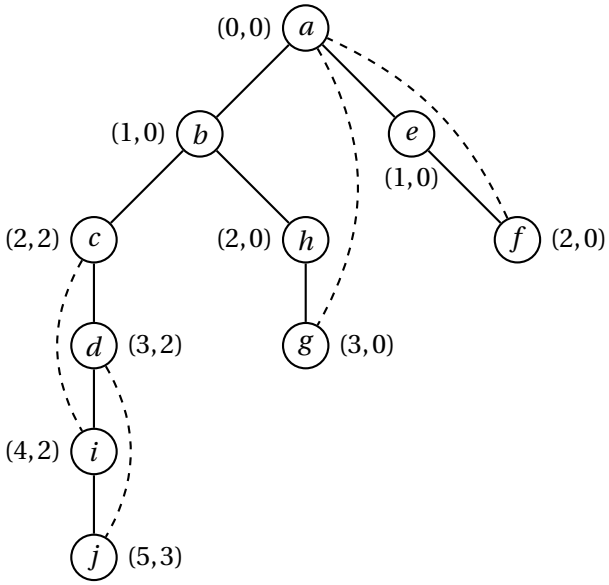
(e) Iskanje v globino teče v času $O(m+T)$, pri čemer je m število povezav v grafu, T pa je čas, porabljen za klic funkcij PREVISIT in POSTVISIT za vsako vozlišče grafa. Funkcija PREVISIT teče v konstantnem času, klic funkcije POSTVISIT(u,v) pa teče v času $O(d_G(u))$, kjer je $d_G(u)$ stopnja vozlišča u v grafu G. Ker je vsota stopenj vozlišč grafa enaka dvakratniku števila povezav, lahko sklenemo, da celoten algoritem teče v času $O(m)$.

Naloga 5.9.

(a) Topološko ureditev dobimo z uporabo algoritma TOPO iz naloge 5.5(a). Dobimo $[s,b,a,d,h,e,c,f,i,g,t]$. Graf v tej ureditvi si lahko ogledamo na sliki 44.

(b) Najkrajšo pot od vozlišča s do vozlišča t lahko izračunamo z algoritmom NAJKRAJŠAPOT iz naloge 5.5(b). Ta nam vrne pot $s-a-d-h-i-t$ dolžine 11. Delovanje algoritma si lahko ogledamo v tabeli 16.

(c) Ker se algoritem odloča neodvisno od svojih prejšnjih odločitev, so vsi dogodki neodvisni. Verjetnost, da pridemo do vozlišča v, bo vsota verjetnosti vseh dogodkov potovanja po (različnih) poteh, ki se končajo v v. Verjetnost potovanja po določeni poti pa bo produkt verjetnosti potovanja po vsaki



Slika 43: Drevo iskanja v globino za nalogo 5.8.

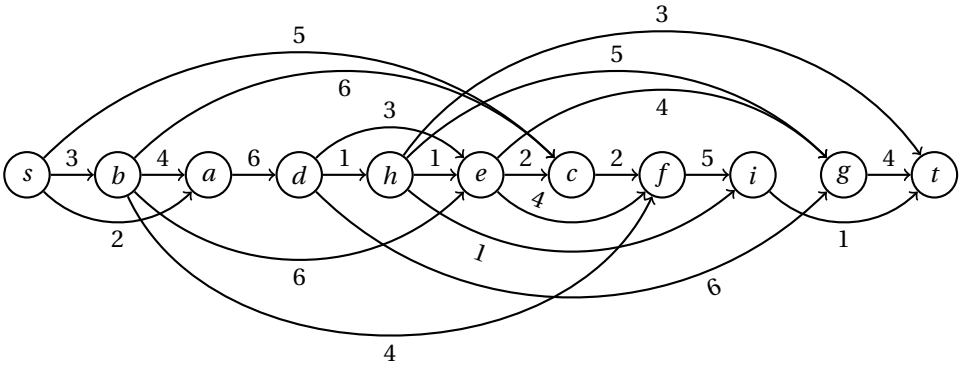
vsebujoči povezavi. Za verjetnosti q_v v usmerjenem acikličnem grafu torej velja zveza

$$q_v = \sum_{u \to v} p_{uv} q_u$$
$$q_s = 1.$$

Vrednosti q_v bomo iskali po topološkem vrstnem redu.

function DAGVERJETNOSTI($G = (V, E), s, \ell : E \to \mathbb{R}_+$)
  $q \leftarrow$ slovar z vrednostjo 0 za vsako vozlišče $v \in V$
  $q[s] \leftarrow 1$
  for $u \in$ TOPO(G) **do**
    $p \leftarrow \sum_{w \in \text{ADJ}_G(u)} \ell_{uw}$
    for $v \in \text{ADJ}_G(u)$ **do**
      $q[v] \leftarrow q[v] + q[u]\ell_{uv}/p$
    end for
  end for
  return q
end function

Časovna zahtevnost algoritma je $O(m)$, saj gremo po vsaki povezavi preko vsakega vozlišča enkrat, pri čemer pa moramo seveda poskrbeti, da vsote v številcu definicije vrednosti p_{uv} ne računamo vsakič sproti, pač pa to storimo pred posodabljanjem verjetnosti sosedom vozlišča u.



Slika 44: Predstavitev topološko urejenega grafa za nalogo 5.9(a).

u	v	$pred[v]$	$d_G[v]$
s	a	s	2
s	b	s	3
s	c	s	5
b	a	s	2
b	c	s	5
b	e	b	9
b	f	b	7
a	d	a	8
d	e	b	9
d	g	d	14
d	h	d	9
h	e	b	9
h	g	d	14
h	i	h	10
h	t	h	12
e	c	s	5
e	f	b	7
e	g	e	13
g	t	h	12
c	f	b	7
f	i	h	10
i	t	i	11

Tabela 16: Potek izvajanja algoritma NAJKRAJŠAPOT za nalogo 5.9(b).

u	v	Q	BX	AM	BL	PR	BS	PA	LU	BN	WI	BU	MO	VE	LJ	ZG
		[(LJ, 0)]	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0	∞
LJ	VE	[(VE, 35)]												35_{LJ}		
LJ	WI	[(WI, 33), (VE, 35)]									33_{LJ}					
LJ	ZG	[(ZG, 20), (WI, 33), (VE, 35)]														20_{LJ}
LJ	BU	[(ZG, 20), (BU, 25), (WI, 33), (VE, 35)]										25_{LJ}				
ZG	LJ	[(BU, 25), (WI, 33), (VE, 35)]														
ZG	BU	[(BU, 25), (WI, 33), (VE, 35)]														
BU	LJ	[(WI, 33), (VE, 35)]														
BU	ZG	[(WI, 33), (VE, 35)]														
BU	WI	[(WI, 33), (VE, 35)]														
BU	BS	[(WI, 33), (VE, 35), (BS, 40)]					40_{BU}									
WI	LJ	[(VE, 35), (BS, 40)]														
WI	BU	[(VE, 35), (BS, 40)]														
WI	BS	[(VE, 35), (BS, 40)]														
WI	PR	[(VE, 35), (BS, 40), (PR, 53)]				53_{WI}										
WI	BN	[(VE, 35), (BS, 40), (PR, 53), (BN, 68)]								68_{WI}						
VE	LJ	[(BS, 40), (PR, 53), (BN, 68)]														
VE	MO	[(BS, 40), (PR, 53), (BN, 68), (MO, 79)]											79_{VE}			
BS	BU	[(PR, 53), (BN, 68), (MO, 79)]														
BS	WI	[(PR, 53), (BN, 68), (MO, 79)]														
BS	PR	[(PR, 53), (BN, 68), (MO, 79)]														
PR	BS	[(BN, 68), (MO, 79)]														
PR	WI	[(BN, 68), (MO, 79)]														
PR	BL	[(BN, 68), (MO, 79), (BL, 83)]			83_{PR}											
BN	WI	[(MO, 79), (BL, 83)]														
BN	MO	[(MO, 79), (BL, 83)]														
BN	LU	[(MO, 79), (BL, 83), (LU, 101)]							101_{BN}							
MO	VE	[(BL, 83), (LU, 101)]														
MO	BN	[(BL, 83), (LU, 101)]														
MO	PA	[(BL, 83), (LU, 101), (PA, 133)]						133_{MO}								
BL	PR	[(LU, 101), (PA, 133)]														
BL	AM	[(LU, 101), (AM, 127), (PA, 133)]		127_{BL}												
LU	BN	[(AM, 127), (PA, 133)]														
LU	AM	[(AM, 127), (PA, 133)]														
LU	BX	[(AM, 127), (PA, 133), (BX, 150)]	150_{LU}													
LU	PA	[(AM, 127), (PA, 133), (BX, 150)]														
AM	BL	[(PA, 133), (BX, 150)]														
AM	LU	[(PA, 133), (BX, 150)]														
AM	BX	[(PA, 133), (BX, 150)]														
PA	MO	[(BX, 150)]														
PA	LU	[(BX, 150)]														
PA	BX	[(BX, 150)]														
BX	AM	[ ]														
BX	LU	[ ]														
BX	PA	[ ]														

Tabela 17: Potek izvajanja algoritma za nalogo 5.10.

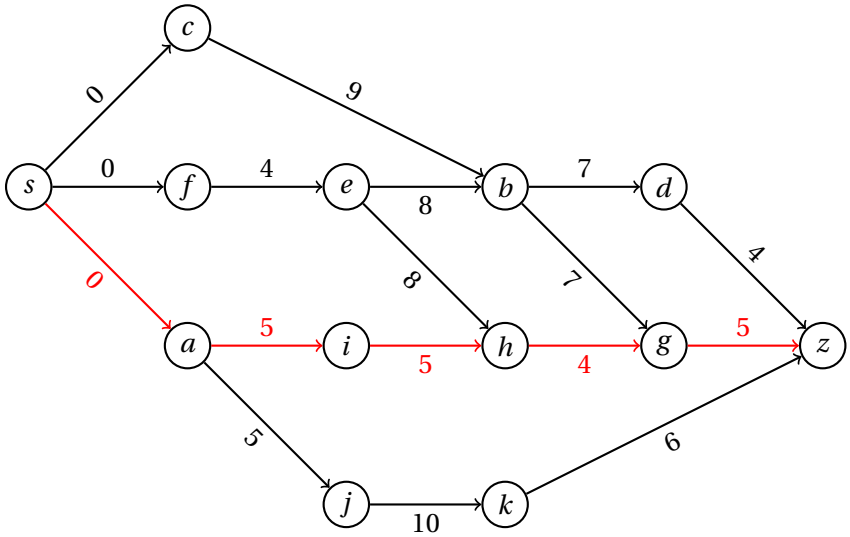
Naloga 5.10.

(a) Imamo neusmerjen graf $G = (V, E)$ z utežmi na vozliščih c_u ($u \in V$) in povezavah ℓ_{uv} ($uv \in E$). Problem bi bilo najlažje reševati z Dijkstrovim algoritmom na usmerjenem grafu $D = (V, A)$, kjer je $A = \{uv, vu \mid uv \in E\}$ množica usmerjenih povezav, z utežmi $L_{uv} = \ell_{uv} + c_v$ ($uv \in A$).

(b) Potek reševanja problema je prikazan v tabeli 17. Najcenejša pot od LJ do BX je LJ – WI – BN – LU – BX s ceno 150.

Naloga 5.11.

(a) Ustrezni graf je prikazan na sliki 45, iz katere je razvidna topološka ureditev $s, c, f, a, e, i, j, b, h, k, d, g, z$. Vozlišče s je bilo dodano kot izhodiščna točka s povezavami dolžine 0 do predmetov, h katerim lahko Peter pristopi takoj.



Slika 45: Graf in najkrajša pot za nalogo 5.11.

s	c	f	a	e	i	j	b	h	k	d	g	z
0	0_s	0_s	0_s	4_f	5_a	5_a	9_c	10_i	15_j	16_b	14_h	19_g

Tabela 18: Izračunane vrednosti v algoritmu za nalogo 5.11(b).

(b) S pomočjo algoritma NAJKRAJŠAPOT iz naloge 5.5(b) poiščemo razdalje od vozlišča s do ostalih vozlišč grafa – te so zbrane v tabeli 18. Iz izhoda lahko rekonstruiramo najkrajšo pot $s-a-i-h-g-z$ dolžine 19, ki je prikazana tudi na sliki 45. Peter naj torej zaporedoma opravi predmete a, i, h in g, kar mu bo omogočilo, da po 19 tednih pristopi k zaključnemu izpitu.

Naloga 5.12.

(a) Definirali bomo usmerjen graf $G' = (V', E')$, kjer je

$$\begin{aligned} V' &= \{s,t\} \cup \{v_A, v_B \mid v \in V \setminus \{s,t\}\} \quad \text{in} \\ E' &= \{u_X v_Y \mid uv \in X \setminus Y,\ X, Y \in \{A,B\},\ u, v \notin \{s,t\}\} \\ &\quad \cup \{s v_Y \mid sv \in E \setminus Y,\ Y \in \{A,B\},\ v \notin \{s,t\}\} \\ &\quad \cup \{u_X t \mid ut \in X,\ X \in \{A,B\},\ u \notin \{s,t\}\} \\ &\quad \cup \{st \mid st \in E\}. \end{aligned}$$

Trenutno vozlišče	s	r_A	r_B	u_A	u_B	v_A	v_B	w_A	w_B	t
	0	∞	∞	∞	∞	∞	∞	∞	∞	∞
s	0	∞	4_s	10_s	∞	∞	∞	1_s	∞	12_s
w_A	0	∞	3_{w_A}	10_s	∞	∞	2_{w_A}	1_s	∞	12_s
v_B	0	4_{v_B}	3_{w_A}	10_s	∞	∞	2_{w_A}	1_s	∞	12_s
r_B	0	4_{v_B}	3_{w_A}	10_s	∞	5_{r_B}	2_{w_A}	1_s	∞	10_{r_B}
r_A	0	4_{v_B}	3_{w_A}	10_s	∞	5_{r_B}	2_{w_A}	1_s	6_{r_A}	10_{r_B}
v_A	0	4_{v_B}	3_{w_A}	10_s	7_{v_A}	5_{r_B}	2_{w_A}	1_s	6_{r_A}	10_{r_B}
w_B	0	4_{v_B}	3_{w_A}	9_{w_B}	7_{v_A}	5_{r_B}	2_{w_A}	1_s	6_{r_A}	10_{r_B}
u_B	0	4_{v_B}	3_{w_A}	9_{w_B}	7_{v_A}	5_{r_B}	2_{w_A}	1_s	6_{r_A}	10_{r_B}
u_A	0	4_{v_B}	3_{w_A}	9_{w_B}	7_{v_A}	5_{r_B}	2_{w_A}	1_s	6_{r_A}	10_{r_B}
t	0	4_{v_B}	3_{w_A}	9_{w_B}	7_{v_A}	5_{r_B}	2_{w_A}	1_s	6_{r_A}	10_{r_B}

Tabela 19: Potek reševanja za nalogo 5.12(b).

Cene povezav ohranimo – velja torej

$$\begin{aligned} L'_{u_X v_Y} &= L_{uv} && (u_X v_Y \in E'), \\ L'_{s v_Y} &= L_{sv} && (s v_Y \in E'), \\ L'_{u_X t} &= L_{ut} && (u_X t \in E'), \\ L'_{st} &= L_{ut} && (st \in E'). \end{aligned}$$

Z Dijkstrovim algoritmom lahko poiščemo najkrajšo pot od s do t v grafu G', ki bo oblike $s - w^{(1)}_{X_1} - w^{(2)}_{X_2} - \cdots - w^{(k)}_{X_k} - t$, kjer velja $X_i \in \{A, B\}$ $(1 \le i \le k)$ in $X_{i+1} \neq X_i = X_{i+2}$ $(1 \le i \le k-2)$. Iz te poti lahko izluščimo najkrajšo alternirajočo pot $s - w^{(1)} - w^{(2)} - \cdots - w^{(k)} - t$ v grafu G.

Graf G' ima $2n-2$ vozlišč in m povezav, kjer je n število vozlišč in m število povezav v grafu G. Če v Dijkstrovem algoritmu za prednostno vrsto uporabimo kopico, je časovna zahtevnost algoritma tako $O(m \log n)$.

(b) Potek izvajanja Dijkstrovega algoritma na pomožnem grafu G' je prikazan v tabeli 19. Najkrajša pot v grafu G' je $s - w_A - r_B - t$ dolžine 10, kar ustreza najkrajši alternirajoči poti $s - w - r - t$ v grafu G.

Naloga 5.13.

(a) Algoritem BFS iz naloge 5.1 vrne želeni slovar. Ker lahko koren poljubno izberemo, lahko slovar *pred* dobimo s klicem BFS(T, V, NOOP). Neposredni nasledniki vozlišča u v drevesu T so tisti njegovi sosedi $v \in \text{ADJ}(T, u)$, za katere velja $v \neq pred[u]$.

(b) Naj bosta x_u in y_u teži najtežje neodvisne množice S v poddrevesu s korenom u (glede na slovar *pred*), za katero velja $u \in S$ oziroma $u \notin S$. Potem velja

$$x_u = c_u + \sum_{\substack{v \sim u \\ v \neq pred[u]}} y_v \qquad \text{in}$$
$$y_u = \sum_{\substack{v \sim u \\ v \neq pred[u]}} \max\{x_v, y_v\}.$$

Vrednosti x_u in y_u računamo od listov v smeri proti korenu r (tj., v topološki ureditvi glede na *pred*). Težo najtežje neodvisne množice dobimo kot $c^* = \max\{x_r, y_r\}$.

(c) Da iz vrednosti x_u in y_u ($u \in V$) izluščimo najtežjo neodvisno množico S v drevesu T, se sprehodimo od korena proti listom (npr. z iskanjem v širino) in posamezno vozlišče u dodamo v množico S, če njegov prednik ni v S in velja $x_u \geq y_u$.

```
function NajtežjaNeodvisnaMnožica(T = (V, E), r, (x_u)_{u∈V}, (y_u)_{u∈V})
    S ← prazna množica
    function Visit(u, v)
        if v ∉ S ∧ x_u ≥ y_u then
            S.add(u)
        end if
    end function
    BFS(T, {r}, Visit)
    return S
end function
```

(d) Naj bo n število vozlišč drevesa T. Potem ima T natanko $n-1$ povezav. Za izračun najtežje neodvisne množice naredimo tri preglede v širino, pri čemer v vsakem vozlišču porabimo konstantno mnogo časa. Časovna zahtevnost celotnega algoritma je torej $O(n)$.

(e) Izračunajmo vrednosti x_u in y_u ($u \in V$), če za koren vzamemo vozlišče a.

x_i	$= 9$	y_i	$= 0$
x_j	$= 4$	y_j	$= 0$
x_k	$= 2$	y_k	$= 0$
x_d	$= 1$	y_d	$= 0$
$x_e = 1+0+0$	$= 1$	$y_e = \max\{\underline{9},0\} + \max\{\underline{4},0\}$	$= 13$
x_f	$= 4$	y_f	$= 0$
$x_g = 8+0$	$= 8$	$y_g = \max\{\underline{2},0\}$	$= 2$
x_h	$= 4$	y_h	$= 0$
$x_b = 1+0+13+0$	$= 14$	$y_b = \max\{\underline{1},0\} + \max\{1,\underline{13}\} + \max\{\underline{4},0\}$	$= 18$
$x_c = 9+2+0$	$= 11$	$y_c = \max\{\underline{8},0\} + \max\{\underline{4},0\}$	$= 12$

$$x_a = 8 + 18 + 12 \quad = 38 \qquad y_a = \max\{14, \underline{18}\} + \max\{11, \underline{12}\} \qquad = 30$$

Teža najtežje neodvisne množice S je torej $c^* = \max\{x_a, y_a\} = 38$. Poglejmo, katera vozlišča so v množici S.

$$\begin{aligned}
c^* &= x_a \\
x_a &= c_a + y_b + y_c && a \in S \\
y_b &= x_d + y_e + x_f && b \notin S \\
y_c &= x_g + x_h && c \notin S \\
x_d &= c_d && d \in S \\
y_e &= x_i + x_j && e \notin S \\
x_f &= c_f && f \in S \\
x_g &= c_g + y_k && g \in S \\
x_h &= c_h && h \in S \\
x_i &= c_i && i \in S \\
x_j &= c_j && j \in S \\
y_k &= 0 && k \notin S
\end{aligned}$$

Najtežja neodvisna množica je torej $S = \{a, d, f, g, h, i, j\}$.

Naloga 5.14.

(a) Za dani graf $G = (V, E)$, vozlišči $s, t \in V$, uteži povezav L_{uv} $(uv \in E)$ in uteži vozlišč c_v $(v \in V)$ bomo izračunali nove uteži povezav $\ell_{uv} = L_{uv} + c_v$ $(uv \in E)$, ki predstavljajo skupen strošek prehoda od operaterja u k operaterju v. Ker so te uteži še vedno lahko tudi negativne, bomo v grafu G poiskali najkrajšo pot od s do t s pomočjo Bellman-Fordovega algoritma, ki teče v času $O(mn)$, kjer je n število vozlišč in m število povezav grafa G.

(b) Najprej izračunajmo nove uteži povezav.

$$\begin{aligned}
\ell_{\text{brez, Bobitel}} &= 65 + 15 = 80 \\
\ell_{\text{brez, Sodafon}} &= 85 + 10 = 95 \\
\ell_{\text{brez, Rega}} &= 95 + 0 \quad = 95 \\
\ell_{\text{brez, Fenmobil}} &= 50 + 30 = 80 \\
\ell_{\text{brez, Z}^0} &= 60 + 5 \quad = 65 \\
\ell_{\text{brez, TeleJanez}} &= 45 + 20 = 65 \\
\ell_{\text{Bobitel, Sodafon}} &= 10 + 10 = 20 \\
\ell_{\text{Sodafon, Rega}} &= -5 + 0 \quad = -5 \\
\ell_{\text{Rega, Fenmobil}} &= -15 + 30 = 15
\end{aligned}$$

i	u	v	ℓ_{uv}	*brez*	*Bobitel*	*Sodafon*	*Rega*	*Fenmobil*	Z^0	*TeleJanez*
1	*brez*	ostali		0	80_{brez}	95_{brez}	95_{brez}	80_{brez}	65_{brez}	65_{brez}
1	*Sodafon*	*Rega*	-5	0	80_{brez}	95_{brez}	$90_{Sodafon}$	80_{brez}	65_{brez}	65_{brez}
1	*TeleJanez*	*Fenmobil*	10	0	80_{brez}	95_{brez}	$90_{Sodafon}$	$75_{TeleJanez}$	65_{brez}	65_{brez}
1	*Fenmobil*	*Sodafon*	10	0	80_{brez}	$90_{Fenmobil}$	$90_{Sodafon}$	$75_{TeleJanez}$	65_{brez}	65_{brez}
2	*Sodafon*	*Rega*	-5	0	80_{brez}	$90_{Fenmobil}$	$85_{Sodafon}$	$75_{TeleJanez}$	65_{brez}	65_{brez}

Tabela 20: Potek izvajanja algoritma za nalogo 5.14(b).

$$\begin{aligned}
\ell_{Fenmobil,\, Z^0} &= -10+5 &&= -5\\
\ell_{Z^0,\, TeleJanez} &= 5+20 &&= 25\\
\ell_{TeleJanez,\, Bobitel} &= 0+15 &&= 15\\
\ell_{TeleJanez,\, Fenmobil} &= -20+30 &&= 10\\
\ell_{Fenmobil,\, Sodafon} &= 5+10 &&= 15\\
\ell_{Sodafon,\, TeleJanez} &= 20+20 &&= 40
\end{aligned}$$

Za izračun najcenejše poti od vozlišča *brez* do vozlišča *Rega* bomo sledili algoritmu BELLMANFORD iz naloge 5.4, pri čemer bomo upoštevali vrstni red povezav iz zgornjega seznama. Strnjen potek algoritma (samo koraki, kjer se katera od razdalj spremeni) je prikazan v tabeli 20. Za najcenejše zaporedje prehodov med operaterji tako dobimo *TeleJanez*, *Fenmobil*, *Sodafon*, *Rega*, skupni strošek pa je 75.

Naloga 5.15.

(a) Pomagali si bomo z iskanjem v širino. Zapišimo algoritem, ki kliče funkcijo BFS iz naloge 5.1.

```
function ŠTEVILONAJKRAJŠIH(G = (V, E), s)
    število ← slovar s ključi v ∈ V in vrednostmi 0
    globina ← slovar s ključi v ∈ V in vrednostmi ∞
    število[s] ← 1
    globina[s] ← 0
    function VISIT(u, v)
        for w ∈ ADJ(G, u) do
            if globina[w] = ∞ then
                globina[w] ← globina[u] + 1
            end if
            if globina[w] = globina[u] + 1 then
                število[w] ← število[w] + število[u]
            end if
        end for
    end function
    BFS(G, {s}, VISIT)
    return globina
```

	globina									*število*								
u	a	b	c	d	e	f	g	h	i	a	b	c	d	e	f	g	h	i
	∞	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0	0	0	0	1
i	∞	∞	∞	∞	∞	1	1	1	0	0	0	0	0	0	1	1	1	1
f	∞	∞	2	∞	2	1	1	1	0	0	0	1	0	1	1	1	1	1
g	∞	∞	2	2	2	1	1	1	0	0	0	2	1	1	1	1	1	1
h	∞	∞	2	2	2	1	1	1	0	0	0	2	1	2	1	1	1	1
c	3	3	2	2	2	1	1	1	0	2	2	2	1	2	1	1	1	1
e	3	3	2	2	2	1	1	1	0	2	4	2	1	2	1	1	1	1
d	3	3	2	2	2	1	1	1	0	3	4	2	1	2	1	1	1	1
a	3	3	2	2	2	1	1	1	0	3	4	2	1	2	1	1	1	1
b	3	3	2	2	2	1	1	1	0	3	4	2	1	2	1	1	1	1

Tabela 21: Potek izvajanja algoritma za nalogo 5.15(b).

end function

Vidimo, da s klicanjem funkcije VISIT na vsakem vozlišču vsako povezavo obravnavamo največ dvakrat, tako da je časovna zahtevnost algoritma $O(m)$.

(b) Potek izvajanja algoritma je prikazan v tabeli 21, pri čemer je bil upoštevan abecedni vrstni red obiskovanja sosedov posameznega vozlišča. Končni rezultat lahko preberemo iz zadnje vrstice tabele.

Naloga 5.16.

(a) Zgledovali se bomo po algoritmu NAJKRAJŠAPOT iz naloge 5.5. Zapišimo algoritem, ki kliče funkciji TOPO in REKONSTRUIRAJPOT iz iste naloge, poleg tega pa kliče tudi funkcijo ADJIN, ki vrne množico vozlišč v podanem usmerjenem grafu s povezavo do podanega vozlišča.

function JADRNICA($G = (V, E), s, t, (k_u)_{u \in V}$)
  $c \leftarrow$ slovar z vrednostjo $-\infty$ za vsako vozlišče $v \in V$
  $pred \leftarrow$ slovar z vrednostjo NULL za vsako vozlišče $v \in V$
  for $u \in$ TOPO(G) **do**
    for $v \in$ ADJIN(G, u) **do**
      if $c[v] > c[u]$ **then**
        $c[v] \leftarrow c[u]$
        $pred[v] \leftarrow u$
      end if
    end for
    if $c[u] > k_u$ **then**
      $c[u] \leftarrow k_u$
    end if
  end for

KP	PU	RI	ML	ZD	DO	ŠI	ST	HV	DU
20	18_{KP}	14_{PU}	12_{PU}	16_{PU}	15_{ZD}	13_{ZD}	$13_{\text{ŠI}}$	15_{DO}	15_{HV}

Tabela 22: Izračunane vrednosti v algoritmu za nalogo 5.16(b).

```
    return REKONSTRUIRAJPOT(pred, t)
  end function
```

Algoritem opravi topološko urejanje grafa, nato vsako povezavo pregleda enkrat, nazadnje pa opravi še obratni prehod po najdeni poti, tako da je njegova časovna zahtevnost $O(m)$.

(b) Graf s slike 21 ima topološko ureditev KP, PU, RI, ML, ZD, DO, ŠI, ST, HV, DU. Izračunane vrednosti v tabelah c in *pred* v algoritmu iz točke (a) so prikazane v tabeli 22, iz katere je razvidno, da gre lahko na jadrnico največ 15 prijateljev, kar je mogoče doseči z izbiro poti KP – PU – ZD – DO – HV – DU.

Naloga 5.17.
Sledili bomo naslednjemu algoritmu.

```
function FLOYDWARSHALL(G = (V, E), L : E → ℝ)
  d ← slovar z vrednostjo ∞ za vsak par vozlišč u, v ∈ V
  pred ← slovar z vrednostjo NULL za vsak par vozlišč u, v ∈ V
  for u ∈ V do
    d[u, u] ← 0
  end for
  for uv ∈ E do
    d[u, v] ← L_uv
    pred[u, v] ← u
  end for
  for w ∈ V do
    for u ∈ V do
      if d[u, w] + d[w, u] < 0 then
        return graf ima negativen cikel
      end if
      for v ∈ V do
        ℓ ← d[u, w] + d[w, v]
        if ℓ < d[u, v] then
          d[u, v] ← ℓ
          pred[u, v] ← pred[w, v]
        end if
      end for
    end for
  end for
  return (d, pred)
end function
```

Po vsakem obhodu zunanje zanke **for** slovar d vsebuje dolžine tistih najkrajših poti, ki v svoji notranjosti (tj., brez začetnega in končnega vozlišča) obiščejo le

w	u	v	$d[uv]$	$pred[u,v]$
	a	b	3	a
	a	f	6	a
	b	c	7	b
	b	e	9	b
	c	d	-7	c
	d	b	9	d
	d	h	4	d
	e	d	1	e
	e	g	1	e
	f	g	8	f
	g	f	-1	g
	h	g	-5	h
b	a	c	10	b
b	a	e	12	b
b	d	c	16	b
b	d	e	18	b
c	a	d	3	c
c	b	d	0	c
d	a	h	7	d
d	b	h	4	d
d	c	b	2	d
d	c	e	11	b
d	c	h	-3	d
d	e	b	10	d
d	e	c	17	b
d	e	h	5	d

w	u	v	$d[uv]$	$pred[u,v]$
e	a	g	13	e
e	b	g	10	e
e	c	g	12	e
e	d	g	19	e
g	b	f	9	g
g	c	f	11	g
g	d	f	18	g
g	e	f	0	g
g	h	f	-6	g
h	a	f	1	g
h	a	g	2	h
h	b	f	-2	g
h	b	g	-1	h
h	c	f	-9	g
h	c	g	-8	h
h	d	f	-2	g
h	d	g	-1	h
h	e	f	-1	g
h	e	g	0	h

Tabela 23: Potek izvajanja algoritma za nalogo 5.17.

tista vozlišča, ki jih je ta zanka že obravnavala. Algoritem pri tem preverja, ali je našel zanko negativne dolžine – tedaj sklene, da v grafu obstaja negativen cikel. Če graf nima negativnih ciklov, po izhodu iz zunanje zanke slovar d vsebuje dolžine najkrajših poti med vsemi pari vozlišč. Časovna zahtevnost algoritma je $O(n^3)$, kjer je n število vozlišč grafa.

Potek zgornjega algoritma na grafu s slike 11 je prikazan v tabeli 23 (prikazani so samo koraki, ko se katera od vrednosti v slovarju d spremeni). Izračunane razdalje so povzete v tabeli 24, v kateri je prikazano tudi, preko katerega vozlišča gre pot, preden obišče ciljno vozlišče (prazni vnosi pomenijo, da ustrezna pot ne obstaja).

	a	b	c	d	e	f	g	h
a	0	3_a	10_b	3_c	12_b	1_g	2_h	7_d
b		0	7_b	0_c	9_b	-2_g	-1_h	4_d
c		2_d	0	-7_c	11_b	-9_g	-8_h	-3_d
d		9_d	16_b	0	18_b	-2_g	-1_h	4_d
e		10_d	17_b	1_e	0	-1_g	0_h	5_d
f						0	8_f	
g						-1_g	0	
h						-6_g	-5_h	0

Tabela 24: Najkrajše razdalje za nalogo 5.17.

2.6 CPM/PERT

Naloga 6.1.

(a) Projekt predstavimo z usmerjenim acikličnim grafom $G = (V, E)$, kjer vozlišča predstavljajo opravila, posebej pa dodamo še začetno vozlišče s in končno vozlišče t. Opravilo u povežemo z opravilom v, če je u pogoj za začetek opravila v. Poleg tega vozlišče s povežemo z opravili, ki nimajo predpogojev; opravila, ki niso predpogoj za katero drugo opravilo, pa povežemo z vozliščem t. Povezave iz opravila u uteži mo s trajanjem opravila u, na povezave iz s pa postavimo utež 0.

Na grafu G uporabimo algoritem NAJDALJŠAPOT iz naloge 5.5(c) z začetkom v s in tako za vsako opravilo dobimo najzgodnejši čas, ko ga lahko začnemo. Dolžina najdaljše poti do vozlišča t predstavlja najkrajše možno trajanje celotnega projekta. Nato na obratnem grafu $G' = (V, E')$, kjer je $E' = \{vu \mid uv \in E\}$ množica obratnih povezav z enakimi utežmi, še enkrat uporabimo algoritem NAJDALJŠAPOT, tokrat z začetkom v t, in dobljene razdalje odštejemo od najkrajšega možnega trajanja projekta. Tako za vsako opravilo dobimo še najpoznejši možen čas začetka, da se celotno trajanje projekta ne poveča. Pri kritičnih opravilih sta oba časa enaka.

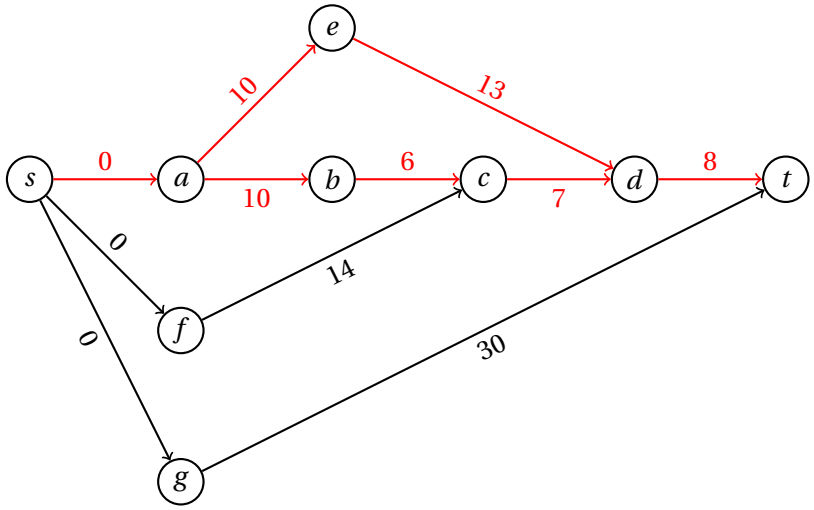
Projekt lahko predstavimo z uteženim grafom s slike 46, iz katerega je razvidna topološka ureditev $s, a, f, g, b, e, c, d, t$. V tabeli 25 so podani rezultati, dobljeni z zgornjim postopkom. Vidimo, da je najkrajše trajanje projekta 31 dni, kritična opravila pa so a, b, c, d, e.

(b) Opazimo, da vsa kritična opravila ležijo na dveh poteh od s do t. Kritični poti sta torej $s-a-e-d-t$ in $s-a-b-c-d-t$.

(c) Iz tabele 25 je razvidno, da je najmanj kritično opravilo f, saj ga lahko brez podaljševanja celotnega trajanja podaljšamo za 2 dni, kar je več kot katerokoli drugo opravilo.

(d) Skrajšati želimo katero od kritičnih opravil, ki leži na obeh kritičnih poteh (taki sta a in d), saj sicer ne bomo skrajšali celotnega trajanja projekta. Ker ima pot $s-g-t$ dolžino 30 dni, bomo lahko celotno trajanje skrajšali za največ 1 dan. To lahko dosežemo, če skrajšamo opravilo a, ki bo tako namesto 10 trajalo 9 dni, skupaj pa bo projekt trajal 30 dni.

Naloga 6.2.
Zapisali bomo linearni program, ki bo za vsako opravilo imel spremenljivki x_u in y_u, ki določata število dni, za katerega skrajšamo trajanje opravila u, oziroma čas začetka izvajanja opravila u.



Slika 46: Graf odvisnosti med opravili in kritična pot za nalogo 6.1.

	s	a	f	g	b	e	c	d	t
najzgodnejši začetek	0	0_s	0_s	0_s	10_a	10_a	16_b	$23_{c,e}$	31_d
najpoznejši začetek	0_a	0_e	2_c	1_t	10_c	10_d	16_d	23_t	31
razlika	0	0^*	2	1	0^*	0^*	0^*	0^*	0

Tabela 25: Razporejanje opravil za nalogo 6.1.

$$\min \qquad 200x_a + 100x_b + 150x_c + 160x_d + 250x_e + 240x_f + 300x_g$$

Opravila brez pogojev:

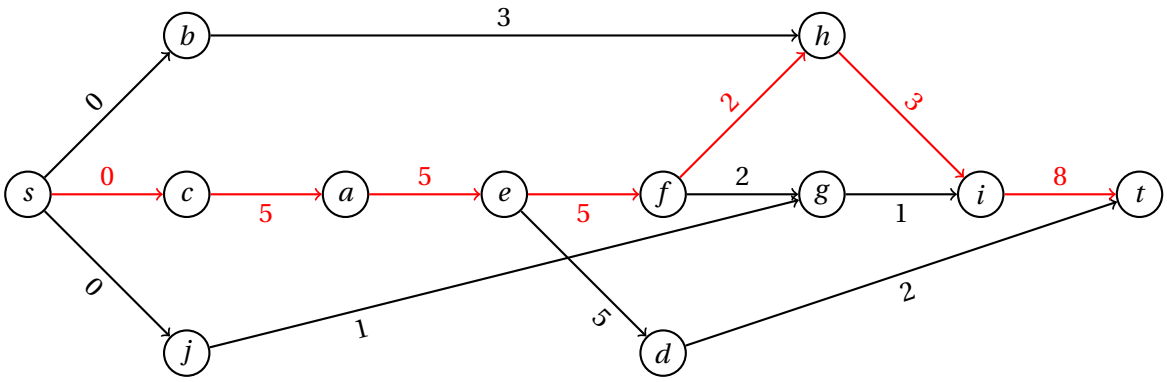
$$y_a, \quad y_d, \quad y_e \geq 0$$

Odvisnosti med opravili:

$$y_b - y_a + x_a \geq 10$$
$$y_e - y_a + x_a \geq 10$$
$$y_c - y_b + x_b \geq 6$$
$$y_d - y_c + x_c \geq 7$$
$$y_d - y_e + x_e \geq 13$$
$$y_c - y_f + x_f \geq 14$$

Želeni čas trajanja:

$$y_d - x_d \leq 19$$
$$y_g - x_g \leq -3$$



Slika 47: Graf odvisnosti med opravili in kritična pot za nalogo 6.3.

Minimalno trajanje opravil:

$$0 \le x_a \le 3$$
$$0 \le x_b \le 1$$
$$0 \le x_c \le 2$$
$$0 \le x_d \le 2$$
$$0 \le x_e \le 4$$
$$0 \le x_f \le 3$$
$$0 \le x_g \le 5$$

Naloga 6.3.

(a) Projekt lahko predstavimo z uteženim grafom s slike 47, iz katerega je razvidna topološka ureditev $s, b, c, j, a, e, d, f, g, h, i, t$.

(b) V tabeli 26 so podani najzgodnejši začetki opravil in najpoznejši začetki, da se celotno trajanje priprave ne podaljša. Priprava bo torej trajala najmanj 28 minut, edina kritična pot pa je $s - c - a - e - f - h - i - t$, ki tako vsebuje tudi vsa kritična opravila.

(c) Iz tabele 26 je razvidno, da je najmanj kritično opravilo j, saj lahko njegovo trajanje podaljšamo za 18 minut, ne da bi vplivali na trajanje celotnega projekta.

	s	b	c	j	a	e	d	f	g	h	i	t
najprej	0	0_s	0_s	0_s	5_c	10_a	15_e	15_e	17_f	17_f	20_h	28_i
najkasneje	0_c	14_h	0_a	18_g	5_e	10_f	26_t	15_h	19_i	17_i	20_t	28
razlika	0	14	0^*	18	0^*	0^*	11	0^*	2	0^*	0^*	0
proste rezerve	0	14	0	16	0	0	11	0	2	0	0	0
varnostne rezerve	0	14	0	18	0	0	11	0	0	0	0	0
neodvisne rezerve	0	14	0	16	0	0	11	0	0	0	0	0

Tabela 26: Razporejanje opravil za nalogo 6.3 in rezerve za nalogo 6.4.

Naloga 6.4.
Skupna rezerva opravila je največji čas, za katerega lahko zamaknemo končanje opravila, ne da bi s tem vplivali na čas končanja projekta. Izračunamo jo kot minimalno razliko najkasnejšega začetka naslednika in najzgodnejšega konca predhodnika, od katere odštejemo trajanje opravila. Skupne rezerve računamo pri določitvi kritičnih in najmanj kritičnih opravil ter so za nalogo 6.3 prikazane v vrstici "razlika" tabele 26.

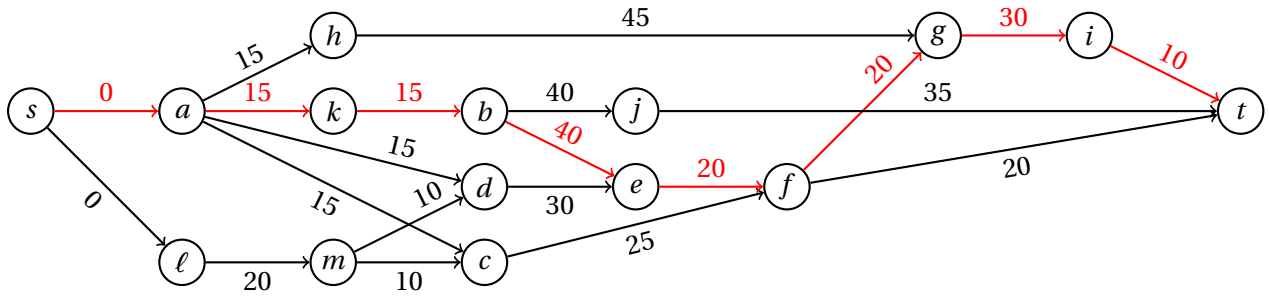
Prosta rezerva opravila je največji čas, za katerega lahko zamaknemo končanje opravila, če z nasledniki začnemo ob najzgodnejšem možnem času. Izračunamo jo kot minimalno razliko najzgodnejšega začetka naslednika in najzgodnejšega konca predhodnika, od katere odštejemo trajanje opravila.

Varnostna rezerva opravila je največji čas, za katerega lahko zamaknemo končanje opravila, ne da bi s tem vplivali na čas končanja projekta, če s predhodniki končamo ob najkasnejšem možnem času. Izračunamo jo kot minimalno razliko najkasnejšega začetka naslednika in najkasnejšega konca predhodnika, od katere odštejemo trajanje opravila.

Neodvisna rezerva opravila je največji čas, za katerega lahko zamaknemo končanje opravila, če z nasledniki začnemo ob najzgodnejšem možnem času in s predhodniki končamo ob najkasnejšem možnem času. Izračunamo jo kot minimalno razliko najzgodnejšega začetka naslednika in najkasnejšega konca predhodnika, od katere odštejemo trajanje opravila.

Tudi proste, varnostne in neodvisne rezerve opravil iz naloge 6.3 so prikazane v tabeli 26.

Naloga 6.5.
Prvi kuhar bo prevzel opravila c, a, e, f, h, i, drugi pa opravila b, j, d, g. Iz tabele 26 je razvidno, da ga najbolj omejuje razporeditev opravilo g – najhitreje ga lahko začne 17 minut po začetku priprave, in tedaj ga konča 18 minut po začetku. Taka razporeditev mu dovoljuje, da z opravilom d začne 15 minut po začetku priprave in ga konča do začetka opravila g. Opravili b in j lahko tako opravi pred tem – začne torej 11 minut po začetku priprave in obe zaključi (vrstni red ni pomemben), preden se loti opravila d. Drugi kuhar lahko tako zaporedoma izvede opravila b, j, d, g, z začetkom 11 minut po začetku priprave in s koncem 7



Slika 48: Graf odvisnosti med opravili in kritična pot za nalogo 6.6.

	s	a	ℓ	h	k	m	b	d	c	j	e	f	g	i	t
najprej	0	0_s	0_s	15_a	15_a	20_ℓ	30_k	30_m	30_m	70_b	70_b	90_e	110_f	140_g	150_i
najkasneje	0_a	0_k	10_m	65_g	15_b	30_d	30_e	40_e	65_f	115_t	70_f	90_g	110_i	140_t	150
razlika	0	0^*	10	50	0^*	10	0^*	10	35	45	0^*	0^*	0^*	0^*	0

Tabela 27: Razporejanje opravil za nalogo 6.6.

minut kasneje (torej brez prestanka).

Naloga 6.6.

(a) Projekt lahko predstavimo z uteženim grafom s slike 48, iz katerega je razvidna topološka ureditev $s, a, \ell, h, k, m, b, d, c, j, e, f, g, i, t$.

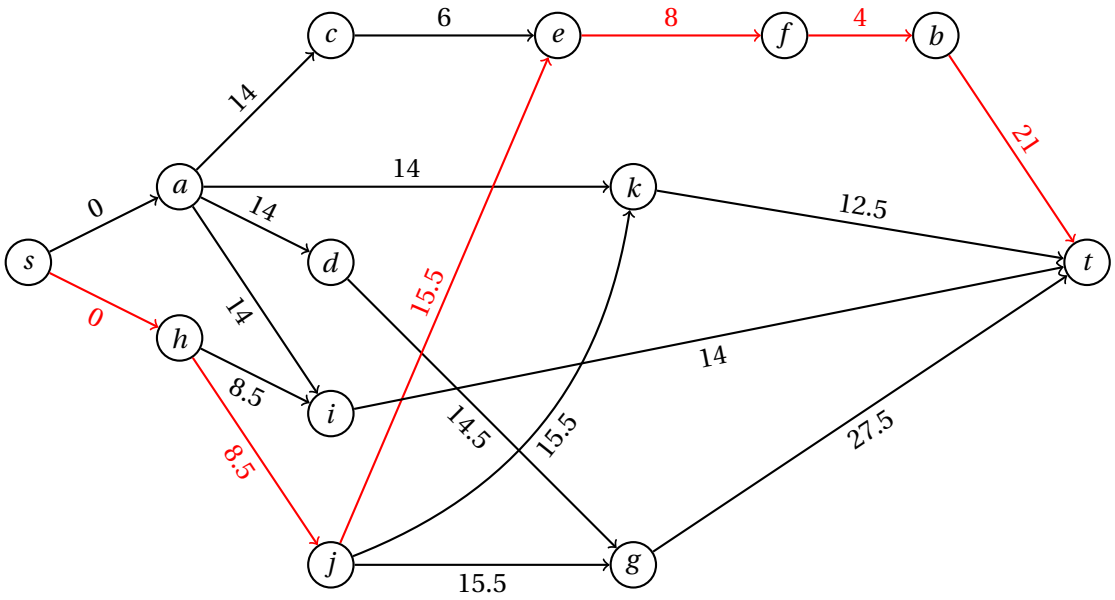
(b) V tabeli 27 so podani najzgodnejši začetki opravil in najpoznejši začetki, da se celotno trajanje projekta ne podaljša. Izdelava bo torej trajala 150 dni, edina kritična pot pa je $s-a-k-b-e-f-g-i-t$, ki tako vsebuje tudi vsa kritična opravila.

(c) Iz tabele 27 je razvidno, da je najmanj kritično opravilo h, saj lahko njegovo trajanje podaljšamo za 50 dni, ne da bi vplivali na trajanje celotnega projekta.

Naloga 6.7.

(a) Projekt lahko predstavimo z uteženim grafom s slike 49, iz katerega je razvidna topološka ureditev $s, a, h, c, d, j, i, k, e, g, f, b, t$. Uteži povezav ustrezajo pričakovanim trajanjem opravil, ki so prikazana v tabeli 28.

(b) V tabeli 28 so podani najzgodnejši začetki opravil in najpoznejši začetki, da se celotno trajanje gradnje ne podaljša. Pričakovano trajanje gradnje je torej $\mu = 57$ dni, edina kritična pot pa je $s-h-j-e-f-b-t$, ki tako vsebuje tudi vsa kritična opravila.



Slika 49: Graf odvisnosti med opravili in kritična pot za nalogo 6.7.

	s	a	h	c	d	j	i	k	e	g	f	b	t
pričakovano		14	8.5	6	14.5	15.5	14	12.5	8	27.5	4	21	
varianca		4	2.25	1	6.25	2.25	1	2.25	1	2.25	0	4	
najprej	0	0_s	0_s	14_a	14_a	8.5_h	14_a	24_j	24_j	28.5_d	32_e	36_f	57_b
najkasneje	0_h	1_d	0_j	18_e	15_g	8.5_e	43_t	44.5_t	24_f	29.5_t	32_b	36_t	57
razlika	0	1	0^*	4	1	0^*	29	20.5	0^*	1	0^*	0^*	0

Tabela 28: Razporejanje opravil za nalogo 6.7.

(c) Iz tabele 28 je razvidno, da je najmanj kritično opravilo i, saj lahko njegovo trajanje podaljšamo za 29 dni v primerjavi s pričakovanim, ne da bi vplivali na trajanje celotnega projekta.

(d) Variance trajanj opravil so prikazane v tabeli 28. Izračunajmo standardni odklon trajanja pričakovane kritične poti.

$$\sigma = \sqrt{4+1+0+2.25+2.25} = 3.082$$

Naj bo X slučajna spremenljivka, ki meri čas izvajanja pričakovane kritične poti. Izračunajmo verjetnost končanja v roku $T = 55$ dni.

$$P(X \leq 55) = \Phi\left(\frac{T-\mu}{\sigma}\right) = \Phi(-0.6489) = 0.2578$$

Naloga 6.8.

(a) Projekt lahko predstavimo z uteženim grafom s slike 50, iz katerega je razvidna topološka ureditev $s, a, b, c, d, f, g, h, e, t$. V tabeli 29 so podani najzgodnejši začetki opravil in najpoznejši začetki, da se celotno trajanje izdelave ne podaljša. Izdelava bo trajala najmanj 88 dni, edina kritična pot pa je $s-b-h-e-t$, ki tako vsebuje vsa kritična opravila.

(b) Iz tabele 29 je razvidno, da je najmanj kritično opravilo c, saj lahko njegovo trajanje podaljšamo za 41 dni, ne da bi vplivali na trajanje celotnega projekta.

(c) Zapišimo linearni program, ki bo za vsako opravilo imel spremenljivki x_u in y_u, ki določata število dni, za katerega skrajšamo trajanje opravila u, oziroma čas začetka izvajanja opravila u.

$$\begin{aligned} \min \quad & 1\,000x_a + 1\,500x_b + 800x_c + 1\,200x_d \\ & + 500x_e + 1\,100x_f + 1\,300x_g + 1\,400x_h \end{aligned}$$

Opravila brez pogojev:

$$y_a, \quad y_b, \quad y_c \geq 0$$

Odvisnosti med opravili:

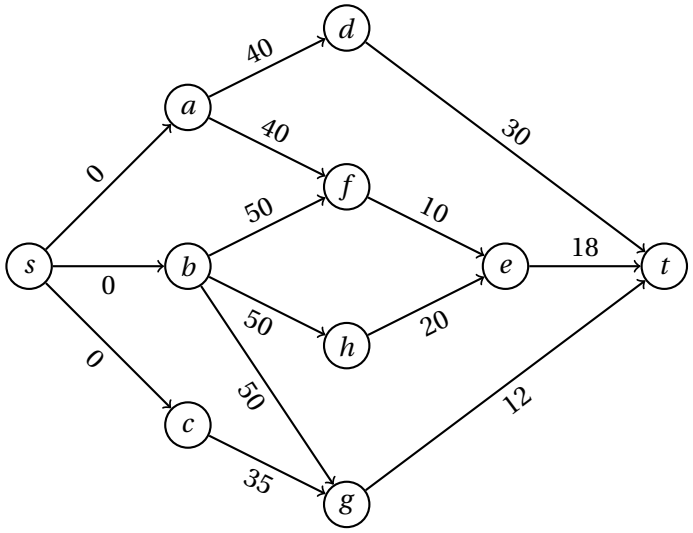
$$\begin{aligned} y_d - y_a + x_a &\geq 40 \\ y_e - y_f + x_f &\geq 10 \\ y_e - y_g + x_g &\geq 12 \\ y_e - y_c + x_c &\geq 90 \\ y_f - y_a + x_a &\geq 40 \\ y_f - y_b + x_b &\geq 50 \\ y_g - y_b + x_b &\geq 50 \\ y_g - y_c + x_c &\geq 35 \\ y_h - y_b + x_b &\geq 50 \end{aligned}$$

Želeni čas trajanja:

$$\begin{aligned} y_d - x_d &\leq 45 \\ y_e - x_e &\leq 57 \\ y_g - x_g &\leq 63 \end{aligned}$$

Minimalno trajanje opravil:

$$\begin{aligned} 0 \leq x_a &\leq 4 \\ 0 \leq x_b &\leq 2 \\ 0 \leq x_c &\leq 4 \\ 0 \leq x_d &\leq 2 \\ 0 \leq x_e &\leq 2 \end{aligned}$$



Slika 50: Graf odvisnosti med opravili in kritična pot za nalogo 6.8.

	s	a	b	c	d	f	g	h	e	t
najzgodnejši začetek	0	0_s	0_s	0_s	40_a	50_b	50_b	50_b	70_h	88_e
najpoznejši začetek	0	18_d	0_h	41_g	58_t	60_e	76_t	50_e	70_t	88
razlika	0	18	0^*	41	18	10	26	0^*	0^*	0

Tabela 29: Razporejanje opravil za nalogo 6.8.

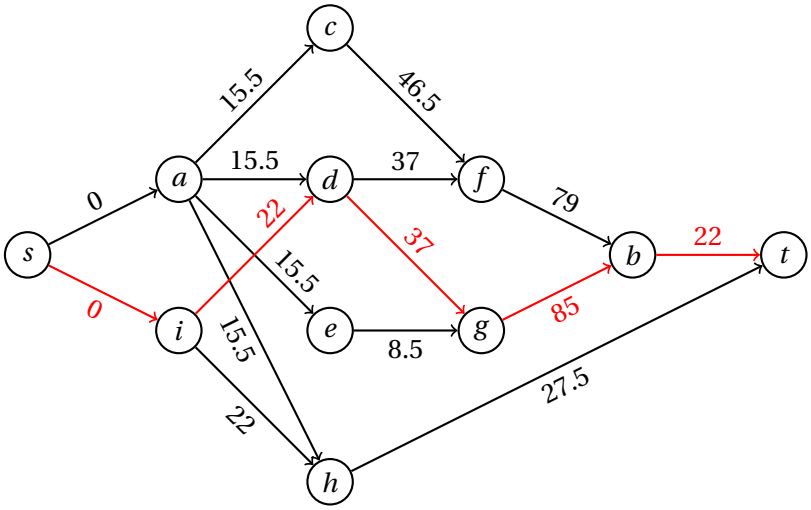
$$0 \leq x_f \leq 1$$
$$0 \leq x_g \leq 2$$
$$0 \leq x_h \leq 2$$

Naloga 6.9.

(a) Projekt lahko predstavimo z uteženim grafom s slike 51, iz katerega je razvidna topološka ureditev $s, a, i, c, d, e, h, f, g, b, t$. Uteži povezav ustrezajo pričakovanim trajanjem opravil, ki so prikazana v tabeli 30.

(b) V tabeli 30 so podani najzgodnejši začetki opravil in najpoznejši začetki, da se celotno trajanje gradnje ne podaljša. Pričakovano trajanje gradnje je torej $\mu = 166$ dni, edina kritična pot pa je $s - i - d - g - b - t$, ki tako vsebuje tudi vsa kritična opravila.

(c) Iz tabele 30 je razvidno, da je najmanj kritično opravilo h, saj lahko njegovo trajanje podaljšamo za 116.5 dni v primerjavi s pričakovanim, ne da bi vplivali na trajanje celotnega projekta.



Slika 51: Graf odvisnosti med opravili in kritična pot za nalogo 6.9.

	s	a	i	c	d	e	h	f	g	b	t
pričakovano		15.5	22	46.5	37	8.5	27.5	79	85	22	
varianca		2.25	1	10.03	9	0.69	2.25	36	16	4	
najprej	0	0_s	0_s	15.5_a	22_i	15.5_a	22_i	62_c	59_d	144_g	166_b
najkasneje	0_i	3_c	0_d	18.5_f	22_g	50.5_g	138.5_t	65_b	59_b	144_t	166
razlika	0	3	0^*	3	0^*	35	116.5	3	0^*	0^*	0

Tabela 30: Razporejanje opravil za nalogo 6.9.

(d) Izračunajmo standardni odklon trajanja pričakovane kritične poti.

$$\sigma = \sqrt{1+9+16+4} = 5.477$$

Naj bo X slučajna spremenljivka, ki meri čas izvajanja pričakovane kritične poti. Izračunajmo verjetnost končanja v roku $T = 180$ dni.

$$P(X \leq 180) = \Phi\left(\frac{T-\mu}{\sigma}\right) = \Phi(2.556) = 0.9947$$

Naloga 6.10.

(a) Projekt lahko predstavimo z uteženim grafom s slike 52, iz katerega je razvidna topološka ureditev $s, a, b, d, c, e, g, f, h, t$. V tabeli 31 so podani najzgodnejši začetki opravil in najpoznejši začetki, da se celotno trajanje gradnje ne podaljša. Gradnja bo trajala najmanj 215 dni, edina kritična pot pa je $s-a-c-g-t$, ki tako vsebuje vsa kritična opravila.

(b) Iz tabele 31 je razvidno, da je najmanj kritično opravilo d, saj lahko njegovo trajanje podaljšamo za 100 dni, ne da bi vplivali na trajanje celotnega projekta.

(c) Zapišimo linearni program, ki bo za vsako opravilo imel spremenljivki x_u in y_u, ki določata število dni, za katerega skrajšamo trajanje opravila u, oziroma čas začetka izvajanja opravila u.

$$\begin{aligned} \min \quad & 300x_a + 200x_b + 700x_c + 1\,100x_d \\ & + 1\,500x_e + 900x_f + 400x_g + 150x_h \end{aligned}$$

Opravila brez pogojev:

$$y_a, \quad y_b \geq 0$$

Odvisnosti med opravili:

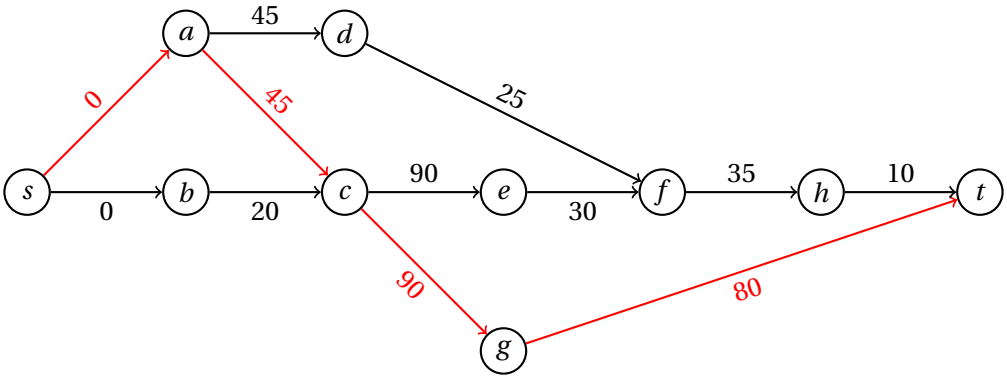
$$\begin{aligned} y_c - y_a + x_a &\geq 45 \\ y_c - y_b + x_b &\geq 20 \\ y_d - y_a + x_a &\geq 45 \\ y_e - y_c + x_c &\geq 90 \\ y_f - y_d + x_d &\geq 25 \\ y_f - y_e + x_e &\geq 30 \\ y_g - y_c + x_c &\geq 90 \\ y_h - y_f + x_f &\geq 35 \end{aligned}$$

Želeni čas trajanja:

$$\begin{aligned} y_g - x_g &\leq 120 \\ y_h - x_h &\leq 190 \end{aligned}$$

Minimalno trajanje opravil:

$$\begin{aligned} 0 \leq x_a &\leq 5 \\ 0 \leq x_b &\leq 5 \\ 0 \leq x_c &\leq 20 \\ 0 \leq x_d &\leq 3 \\ 0 \leq x_e &\leq 4 \\ 0 \leq x_f &\leq 7 \\ 0 \leq x_g &\leq 15 \\ 0 \leq x_h &\leq 2 \end{aligned}$$



Slika 52: Graf odvisnosti med opravili in kritična pot za nalogo 6.10.

	s	a	b	d	c	e	g	f	h	t
najzgodnejši začetek	0	0_s	0_s	45_a	45_a	135_c	135_c	165_e	200_f	215_g
najpoznejši začetek	0	0_c	25_c	145_f	45_g	140_f	135_t	170_h	205_t	215
razlika	0	0^*	25	100	0^*	5	0^*	5	5	0

Tabela 31: Razporejanje opravil za nalogo 6.10.

2.7 Upravljanje zalog

Naloga 7.1.
Imamo sledeče podatke (pri časovni enoti 1 teden).

$$\begin{aligned} \nu &= 10 \\ s &= 0.2€ \\ K &= 150€ \\ \lambda &= 12.5 \\ p &= 0.8€ \end{aligned}$$

Izračunajmo optimalno dolžino cikla τ^* in največjo zalogo M^*.

$$\begin{aligned} \alpha &= \nu\left(1-\frac{\nu}{\lambda}\right) = 10\cdot(1-0.8) = 2 \\ \beta &= 1+\frac{s}{p} &&= 1+\frac{0.2}{0.8} &&= 1.25 \\ \tau^* &= \sqrt{\frac{2K\beta}{s\alpha}} &&= \sqrt{\frac{375}{0.4}} &&\approx 30.619 \\ M^* &= \sqrt{\frac{2K\alpha}{s\beta}} &&= \sqrt{\frac{600}{0.25}} &&\approx 48.990 \end{aligned}$$

Optimalna dolžina cikla je torej približno 30.619 tednov, pri tem pa Marta potrebuje skladišče za 49 kosov nakita.

Izračunajmo še število izdelanih kosov nakita q^*, trajanje proizvodnje t^* in največji primanjkljaj m^* v posameznem ciklu.

$$\begin{aligned} q^* &= \tau^*\nu &&\approx 30.619\cdot 10 &&\approx 306.186 \\ t^* &= \frac{q^*}{\lambda} &&\approx \frac{306.186}{12.5} &&\approx 24.495 \\ m^* &= \tau^*\alpha - M^* &&\approx 30.619\cdot 2 - 48.990 &&\approx 12.247 \end{aligned}$$

V vsakem intervalu torej Marta izdela okoli 306 kosov nakita, pri čemer izdelovanje traja približno 24.495 tednov, primanjkljaj pa ne preseže 13 kosov nakita.

Naloga 7.2.
Imamo sledeče podatke (pri časovni enoti 1 teden).

$$\begin{aligned} \nu &= 60 & \lambda &= 90 \\ K &= 180€ & s &= 0.5€ \end{aligned}$$

Pri teh podatkih izračunajmo še

$$\alpha = \nu\left(1-\frac{\nu}{\lambda}\right) = 60\cdot\left(1-\frac{2}{3}\right) = 20.$$

(a) Če primanjkljaja ne dovolimo, imamo $p = \infty$ in torej $\beta = 1$. Izračunajmo optimalno dolžino cikla τ^*, največjo zalogo M^* in enotske stroške S^*.

$$\tau^* = \sqrt{\frac{2K\beta}{s\alpha}} = \sqrt{\frac{360}{10}} = 6$$
$$M^* = \sqrt{\frac{2K\alpha}{s\beta}} = \sqrt{\frac{7200}{0.5}} = 120$$
$$S^* = \sqrt{\frac{2Ks\alpha}{\beta}} = \sqrt{3600}€ = 60€$$

Optimalna dolžina cikla je torej 6 tednov, pri tem pa mora vulkanizer imeti skladišče za 120 pnevmatik. Tedenski stroški proizvodnje in skladiščenja so 60€.

(b) Izračunajmo še trajanje proizvodnje t^* in število izdelanih pnevmatik q^* v posameznem ciklu.

$$t^* = \frac{M^*}{\lambda - \nu} = \frac{120}{30} = 4$$
$$q^* = \tau^* \nu = t^* \lambda = 6 \cdot 60 = 360$$

Proizvodnja torej traja 4 tedne, skupno pa v tem času izdela 360 pnevmatik.

(c) Vzamemo $p = 2€$ in torej $\beta = 1 + s/p = 1.25$. Izračunajmo optimalno dolžino cikla, potrebno velikost skladišča in enotske stroške še v tem primeru.

$$\tau^* = \sqrt{\frac{2K\beta}{s\alpha}} = \sqrt{\frac{450}{10}} \approx 6.708$$
$$M^* = \sqrt{\frac{2K\alpha}{s\beta}} = \sqrt{\frac{7200}{0.625}} \approx 107.331$$
$$S^* = \sqrt{\frac{2Ks\alpha}{\beta}} = \sqrt{\frac{3600}{1.25}}€ \approx 53.666€$$

Optimalni interval naročanja je v tem primeru torej 6.708 tednov, pri tem pa mora vulkanizer imeti skladišče za 108 pnevmatik. Tedenski stroški proizvodnje in skladiščenja so 53.666€.

Izračunajmo še največji primanjkljaj m^* v posameznem ciklu.

$$m^* = \tau^* \alpha - M^* \approx 6.708 \cdot 20 - 107.331 \approx 26.833$$

Največji primanjkljaj torej ne preseže 27 pnevmatik.

Naloga 7.3.
Imamo sledeče podatke (pri časovni enoti 1 mesec).

$$\nu = 20 \qquad K = 750€$$
$$s = 10€ \qquad p = 50€$$

Pri teh podatkih izračunajmo še

$$\beta = 1 + \frac{s}{p} = 1 + \frac{10}{50} = 1.2.$$

Ker artiklov ne proizvajamo (tj., ob dostavi imamo na voljo celotno količino naročila), vzamemo $\lambda = \infty$ in torej $\alpha = \nu$.

(a) Izračunajmo optimalno dolžino cikla τ^*, največjo zalogo M^* in enotske stroške S^*.

$$\tau^* = \sqrt{\frac{2K\beta}{s\alpha}} = \sqrt{\frac{1\,800}{200}} = 3$$
$$M^* = \sqrt{\frac{2K\alpha}{s\beta}} = \sqrt{\frac{30\,000}{12}} = 50$$
$$S^* = \sqrt{\frac{2Ks\alpha}{\beta}} = \sqrt{\frac{300\,000}{1.2}}€ = 500€$$

Optimalna dolžina cikla je torej 3 mesece, pri tem pa mora trgovina imeti skladišče za 50 sedežnih garnitur. Mesečni stroški nabave in skladiščenja so 60€.

(b) Izračunajmo še največji primanjkljaj m^* in velikost naročila q^*.

$$m^* = \tau^*\alpha - M^* = 3 \cdot 20 - 50 = 10$$
$$q^* = \tau^*\nu = 3 \cdot 20 = 60$$

(c) Drugo skladišče lahko hrani $n' = 40$ sedežnih garnitur, mesečni strošek hrambe enega kosa pa je $s' = 6€$. Najprej preverimo, kakšna bi bila optimalna velikost skladišča pri takih stroških.

$$\beta' = 1 + \frac{s'}{p} = 1 + \frac{6}{50} = 1.12$$
$$M' = \sqrt{\frac{2K\alpha}{s'\beta'}} = \sqrt{\frac{30\,000}{6.72}} \approx 66.815$$

Ker je $M' > n'$, zapišimo formulo za enotske stroške za primer, ko imamo v skladišču največ n' kosov.

$$S' = \frac{2K\alpha + (s'+p)n'^2}{2\alpha\tau'} + p\left(\frac{\alpha\tau'}{2} - n'\right) = 500€ \cdot \tau' - 2\,000€ + \frac{2\,990€}{\tau'}$$

Ker iščemo dolžino intervala, kjer dosežemo minimalne stroške, poiščimo, kje ima odvod zgornjega izraza po τ' ničlo za $\tau' > 0$.

$$0€ = 500€ - \frac{2\,990€}{\tau'^2}$$
$$\tau' = \sqrt{\frac{2\,990}{500}} \approx 2.445$$

Optimalna dolžina cikla ob uporabi drugega skladišča je torej 2.445 meseca. Vstavimo v zgornjo formulo, da dobimo mesečne stroške.

$$S' = 500€ \cdot 2.445 - 2\,000€ + \frac{2\,990€}{2.445} \approx 445.404€$$

Ker so stroški nižji kot pri prvem skladišču, se trgovini izplača sprejeti ponudbo.

Naloga 7.4.
Imamo sledeče podatke (pri časovni enoti 1 dan).

$$\nu = 300 \qquad K = 240€$$
$$\lambda = 400 \qquad s = 0.1€$$

Pri teh podatkih izračunajmo še

$$\alpha = \nu\left(1 - \frac{\nu}{\lambda}\right) = 300 \cdot \frac{1}{4} = 75.$$

(a) Ker primanjkljaj ni dovoljen, vzamemo $p = \infty$ in torej $\beta = 1$. Izračunajmo optimalno dolžino cikla τ^*, največjo zalogo M^*, trajanje proizvodnje t^* ter število izdelanih mask q^* v posameznem ciklu.

$$\begin{aligned}
\tau^* &= \sqrt{\frac{2K\beta}{s\alpha}} &&= \sqrt{\frac{480}{7.5}} &&= 8\\
M^* &= \sqrt{\frac{2K\alpha}{s\beta}} &&= \sqrt{\frac{36\,000}{0.1}} &&= 600\\
t^* &= \frac{M^*}{\lambda - \nu} &&= \frac{600}{100} &&= 6\\
q^* &= \tau^*\nu = t^*\lambda = 8 \cdot 300 &&&&= 2\,400
\end{aligned}$$

Optimalna dolžina cikla je torej 8 dni, od tega proizvodnja teče 6 dni. V tem času proizvedejo 2 400 mask, v skladišču pa mora biti prostora za 600 mask.

(b) Sedaj imamo $p = 0.4€$ in torej $\beta = 1 + s/p = 1.25$. Izračunajmo optimalno dolžino cikla τ^*, največjo zalogo M^*, največji primanjkljaj m^*, trajanje proizvodnje t^* ter število izdelanih mask q^* v posameznem ciklu.

$$\begin{aligned}
\tau^* &= \sqrt{\frac{2K\beta}{s\alpha}} &&= \sqrt{\frac{600}{7.5}} && \approx 8.944 \\
M^* &= \sqrt{\frac{2K\alpha}{s\beta}} &&= \sqrt{\frac{36\,000}{0.125}} && \approx 536.656 \\
m^* &= \tau^*\alpha - M^* && \approx 670.820 - 536.656 && \approx 134.164 \\
t^* &= \frac{M^* + m^*}{\lambda - \nu} && \approx \frac{670.820}{100} && = 6.708 \\
q^* &= \tau^*\nu && \approx 8.944 \cdot 300 && = 2\,683.282
\end{aligned}$$

V tem primeru je torej optimalna dolžina cikla 8.944 dni, od tega proizvodnja teče 6.708 dni. V tem času proizvedejo približno 2 683 mask, v skladišču pa mora biti prostora za 537 mask. Največji primanjkljaj ne preseže 135 mask.

Literatura

[1] S. Dasgupta, C.H. Papadimitriou in U.V. Vazirani, *Algorithms.* McGraw-Hill Education, 2006.