Elfde college algoritmiek

27 april 2007

Branch & Bound

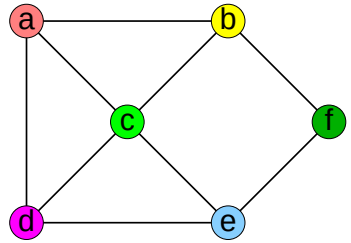
Backtracking

- bouwt een oplossing component voor component op
- kijkt tijdens de constructie of de deeloplossing nog aan de gestelde restricties/eisen voldoet (kan voldoen)
- zo niet, breidt dan de deeloplossing niet verder uit
- spaart zo soms veel werk uit vergeleken met exhaustive search
- bij een minimalisatieprobleem kun je soms ook stoppen met uitbreiden wanneer je deeloplossing al een grotere waarde van de te optimaliseren functie heeft dan de huidige beste oplossing en alleen maar nog groter kan worden (analoog maximalisatieproblemen)

Voorbeeldprobleem

Vind een Hamiltonkring in een gegeven ongerichte graaf.

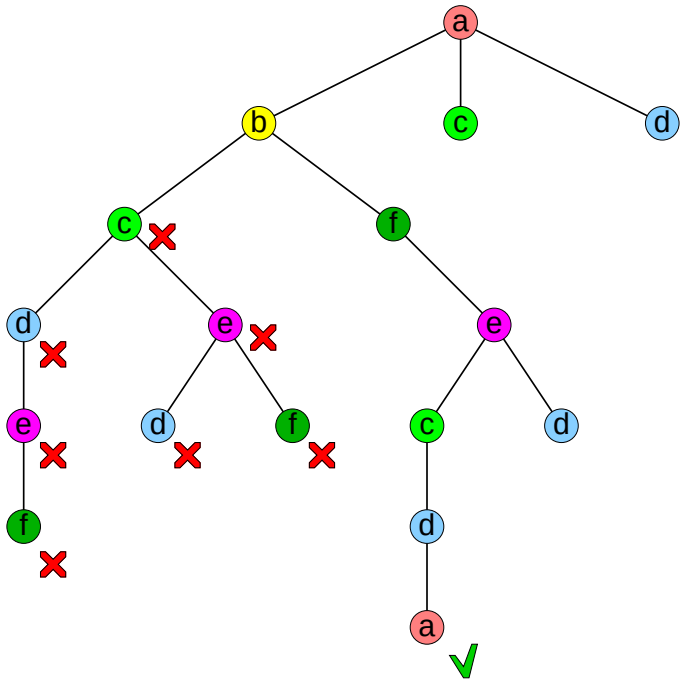
Voorbeeld: a b f e c d is een Hamiltonkring in nevenstaande graaf, echter a b c d e f is geen Hamiltonkring.



Backtracking: genereer de mogelijke Hamiltonkringen door stap voor stap de knopen te kiezen en controleer tijdens de constructie of de deelpermutatie nog wel een pad/kring voorstelt (de restrictie).

State space tree HC

Gebruik van backtracking voor het vinden van een Hamiltonkring in de voorbeeldgraaf leidt tot de volgende state space tree:



- breid het pad telkens met één knoop uit (keuzes in alfabetisch volgorde)
- in de knopen met een rood kruis backtrackt het algoritme zodra blijkt dat die niet tot een oplossing leiden

Optimalisatieprobleem: er wordt een oplossing gezocht met minimale of maximale .....

Terminologie:

- De functie/waarde die ge-optimaliseeerd moet worden heet wel de objectfunctie (bijvoorbeeld de lengte van een Hamiltonkring)

- Een oplossing die voldoet aan de restricties van het probleem heet feasible (toelaatbaar)

- Een optimale oplossing is een/de toelaatbare oplossing met de beste waarde van de objectfunctie

Branch & bound

- is een versnelling/verbetering van backtracking
- is toepasbaar op optimalisatieproblemen
- gebruikt voor elke deeloplossing (=knoop) een ondergrens (resp. bovengrens) op de te verwachten waarde van de objectfunctie, met als doel
  - van deeloplossingen (knopen) —eerder dan bij gewoon backtracking— te kunnen bepalen dat ze niet verder bekeken hoeven te worden: snoeien
  - de zoekvolgorde in de zoekruimte (state space tree) te leiden

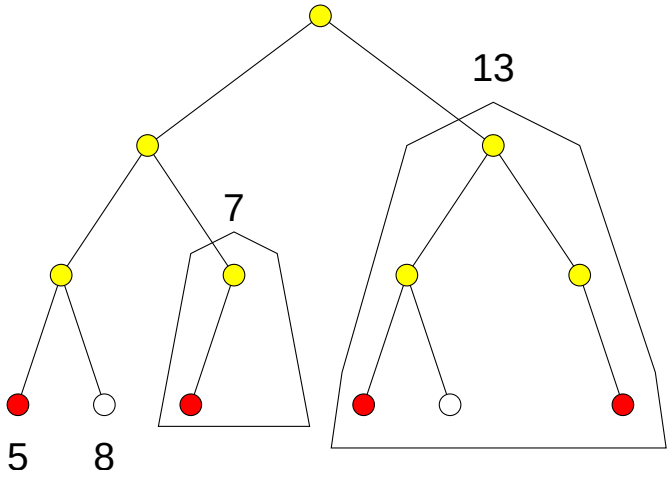
Een branch and bound algoritme breidt een knoop (deeloplossing) dus niet verder uit als

- de waarde van de ondergrens (bovengrens) bij die knoop niet beter is dan de waarde van de tot dusver gevonden beste oplossing

- de deeloplossing niet meer voldoet aan de restricties (of niet meer uit te breiden is tot een toelaatbare oplossing)

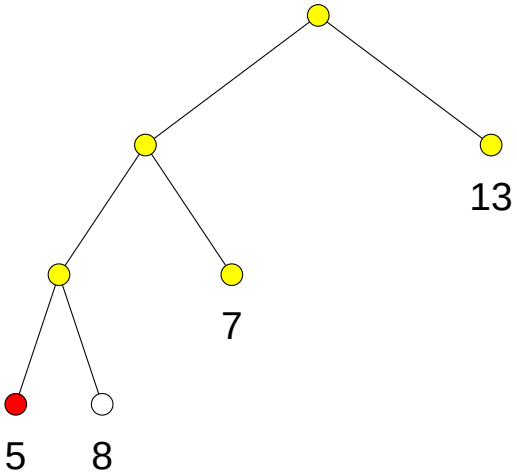
- er geen verdere keuzes meer mogelijk zijn; dan is er een toelaatbare oplossing gevonden: udate beste oplossing

Branch & bound snoeien



state space tree

witte knopen corresponderen met feasible solutions; rode knopen met niet-feasible solutions; de waarden bij de bladeren geven de waarde van de objectfunctie van de bijbehorende oplossing



gesnoeide boom

de waarden bij de andere knopen geven een ondergrens aan voor de te verwachten waarde van de objectfunctie; bij de knoop met ondergrens 13 kan dus gesnoeid worden

Assignmentprobleem (toewijzingsprobleem)

Gegeven n personen en n taken (jobs). Persoon i kan taak j doen voor `kosten`$[i][j]$ euro. **Gevraagd**: de/een toewijzing van de personen aan de jobs (één persoon per job en één job per persoon) met minimale kosten.

Voorbeeld:

	W	X	Y	Z
Alice	9	2	7	8
Bob	6	4	3	7
Carol	5	8	1	8
David	7	6	9	4

	W	X	Y	Z
Alice	9	2	7	8
Bob	6	4	3	7
Carol	5	8	1	8
David	7	6	9	4

Een ondergrens voor de kosten van een optimale oplossing:

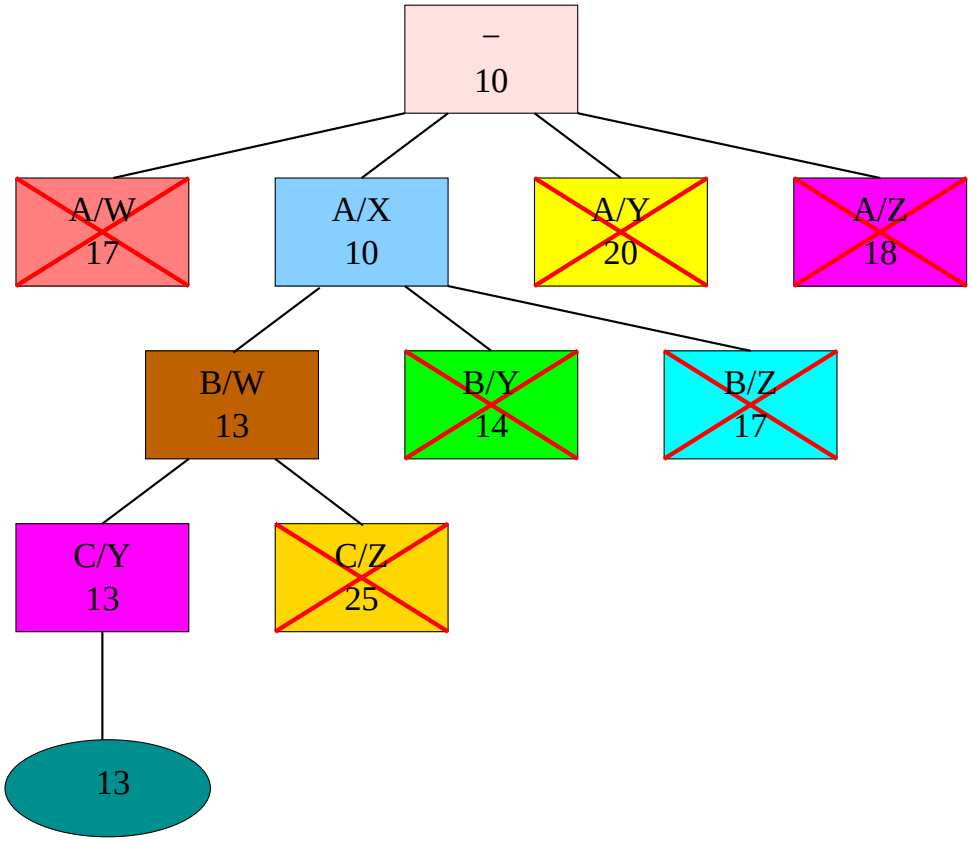
- neem uit elke rij de kleinste waarde en tel die bij elkaar op: $2 + 3 + 1 + 4 = 10$
- neem uit elke kolom de kleinste waarde en tel die bij elkaar op: $5 + 2 + 1 + 4 = 12$

De optimale oplossing heeft totale kosten 13:
Alice job X; Bob job W; Carol job Y; David job Z

Voor een willekeurige knoop/deeloplossing berekenen we de ondergrens op de te verwachten waarde van de object-functie door uit elke verdere rij de kleinste waarde van de nog beschikbare jobs te nemen en deze op te tellen. Voor de deeloplossing(en) die Alice aan job W koppelt zal die ondergrens bijvoorbeeld $9 + 3 + 1 + 4 = 17$ zijn. (Analoog voor kolommen: kies uit elke kolom de kleinste waarde van de nog beschikbare personen.(*))

De volgorde waarin de knopen van de state space tree worden uitgebreid laten we afhangen van de berekende ondergrens. We evalueren de knopen met de beste (=laagste) ondergrens als eerste: dit lijkt de meest veelbelovende knoop. Deze strategie heet wel de best-fit-first branch-and-bound.

B&B state space tree



	W	X	Y	Z
Alice	9	2	7	8
Bob	6	4	3	7
Carol	5	8	1	8
David	7	6	9	4

Opgave: los het probleem op met de ondergrenzen berekend volgens (*).

Los het toewijzingsprobleem op voor onderstaand voorbeeld met behulp van

1. backtracking
2. branch and bound

en vergelijk de hoeveelheid snoeiwerk bij beide methoden.

	W	X	Y	Z
Alice	4	7	3	5
Bob	6	2	9	1
Carol	3	9	5	3
David	1	1	1	8

Gegeven n objecten, met gewicht $w_1, \ldots, w_n$ en waarde $v_1, \ldots, v_n$, en een knapzak met capaciteit W. **Gevraagd**: de meest waardevolle deelverzameling der objecten die in de knapzak past (dus met totaalgewicht $\leq W$).

Voorbeeld:

item	w	v	v/w
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4

Maximaal gewicht $W = 10$

item	w	v	v/w
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4

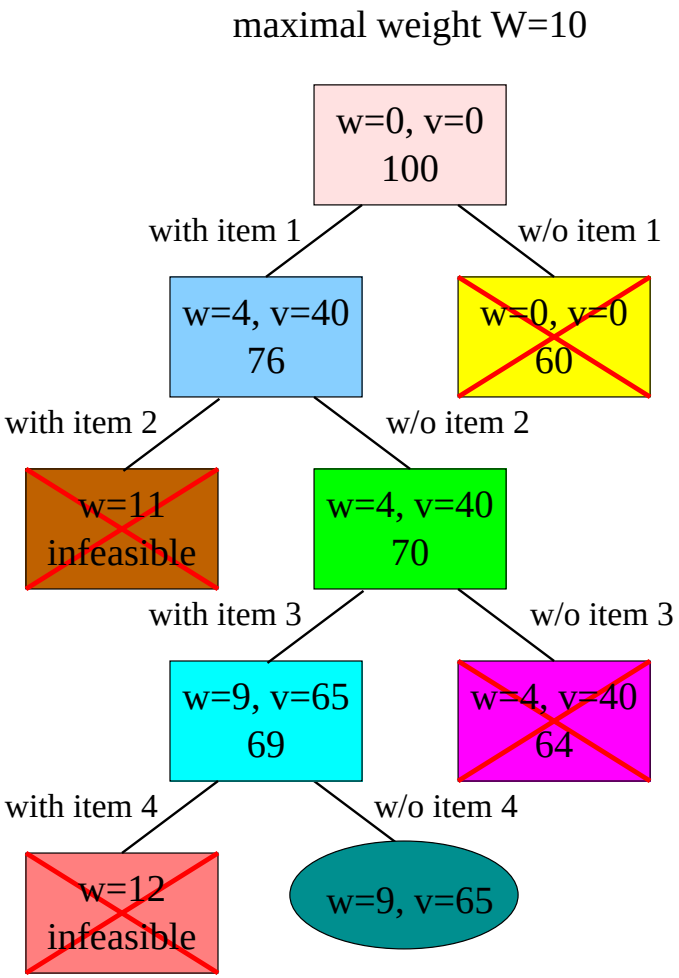
Opbouwen van de oplossing: in de i-de stap wordt object i wel of niet gekozen,

Een bovengrens voor de kosten van een optimale oplossing:

- Bij aanvang: $W * (v_1/w_1) = 100$;
- Na de i-de stap: $v + (W - w) * (v_{i+1}/w_{i+1})$, met v de totaalwaarde van de reeds gekozen objecten en w het totaalgewicht daarvan.

De optimale oplossing heeft gewicht 9 en waarde 65: $\{1, 3\}$.

B&B state space tree

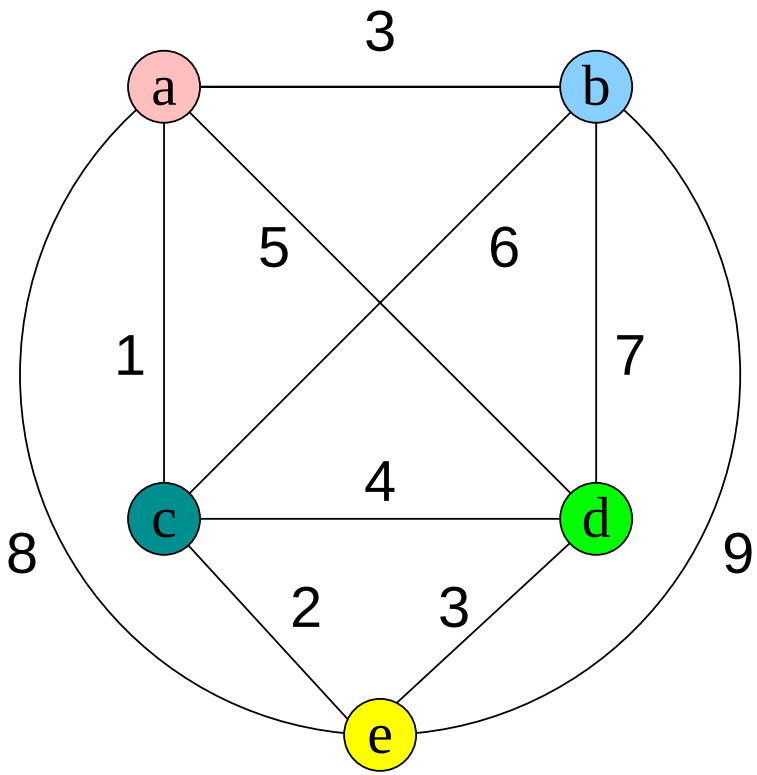


item	w	v	v/w
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4

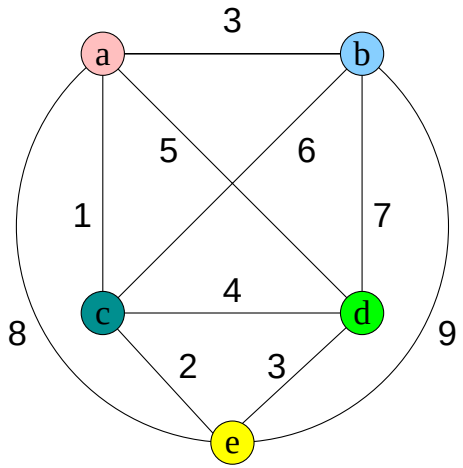
Zie ook exercise 12.2.5, Levitin.

Traveling Salesman Problem (handelsreizigersprobleem)

Gegeven n steden waarvan alle onderlinge afstanden bekend zijn. **Gevraagd**: de/een kortste route die elke stad precies één keer aandoet, en weer terugkeert in het vertrekpunt.
Ofwel: vind de/een kortste Hamiltonkring in een samenhangende gewogen (complete) graaf. Het gaat hier om een ongerichte graaf.



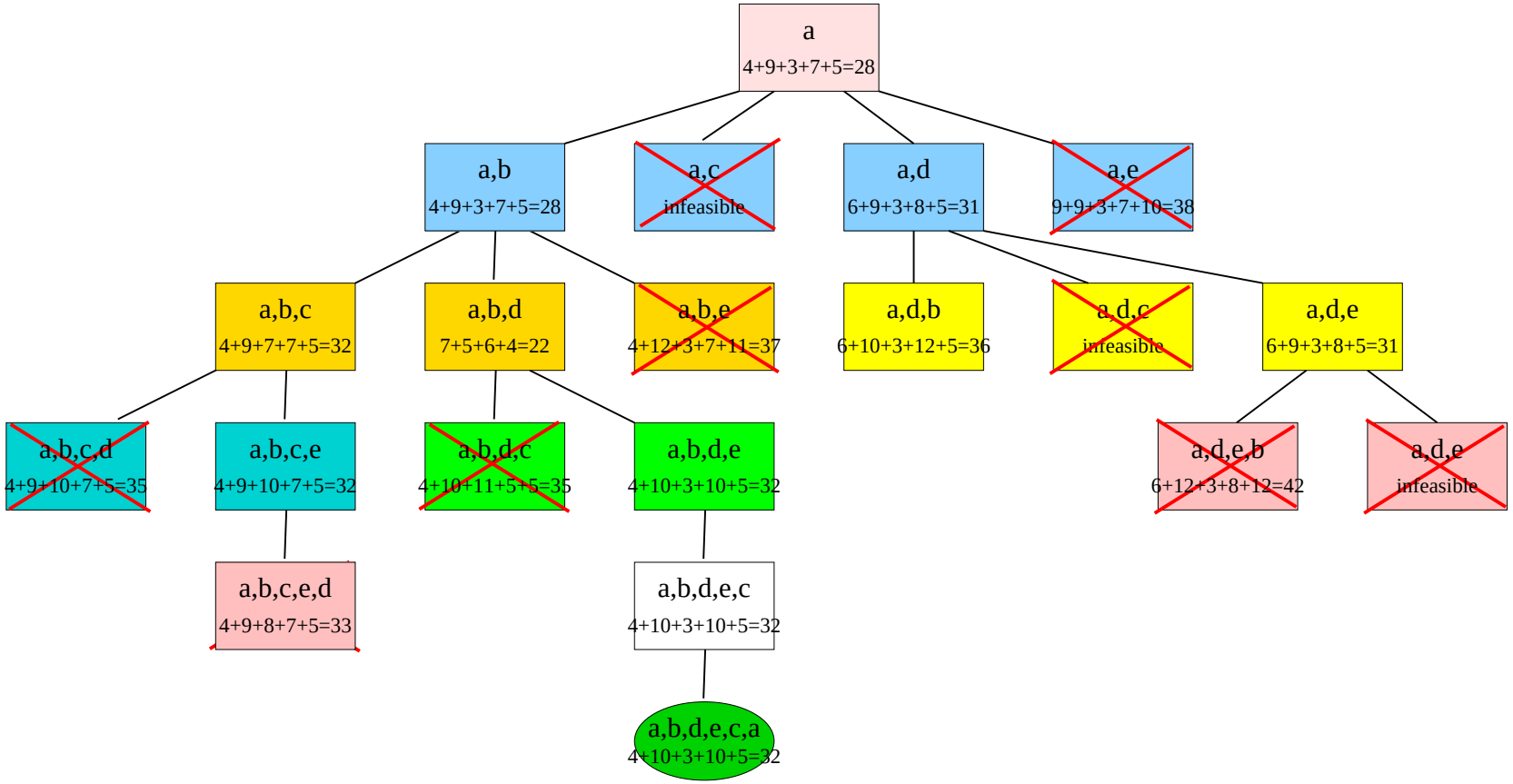
De optimale oplossing heeft lengte 16: a, b, d, e, c



Een ondergrens voor de kosten van een optimale oplossing:

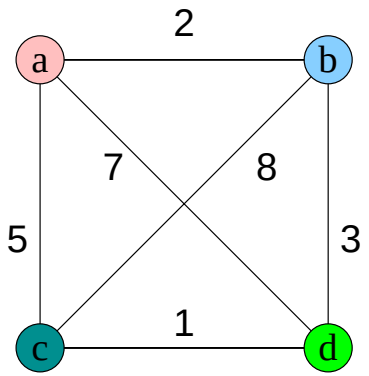
- eenvoudig: $n\times$ kortste afstand tussen twee knopen; $5\times 1 = 5$ bij aanvang (analoog als reeds takken gekozen zijn).
- beter: som over alle knopen i van de afstanden van knoop i tot de twee dichtstbijzijnde knopen (inclusief al gekozen takken), en dat gedeeld door 2(*); $(1+3+3+6+1+2+3+4+3+2)/2 = 14$ bij aanvang.

B&B state space tree



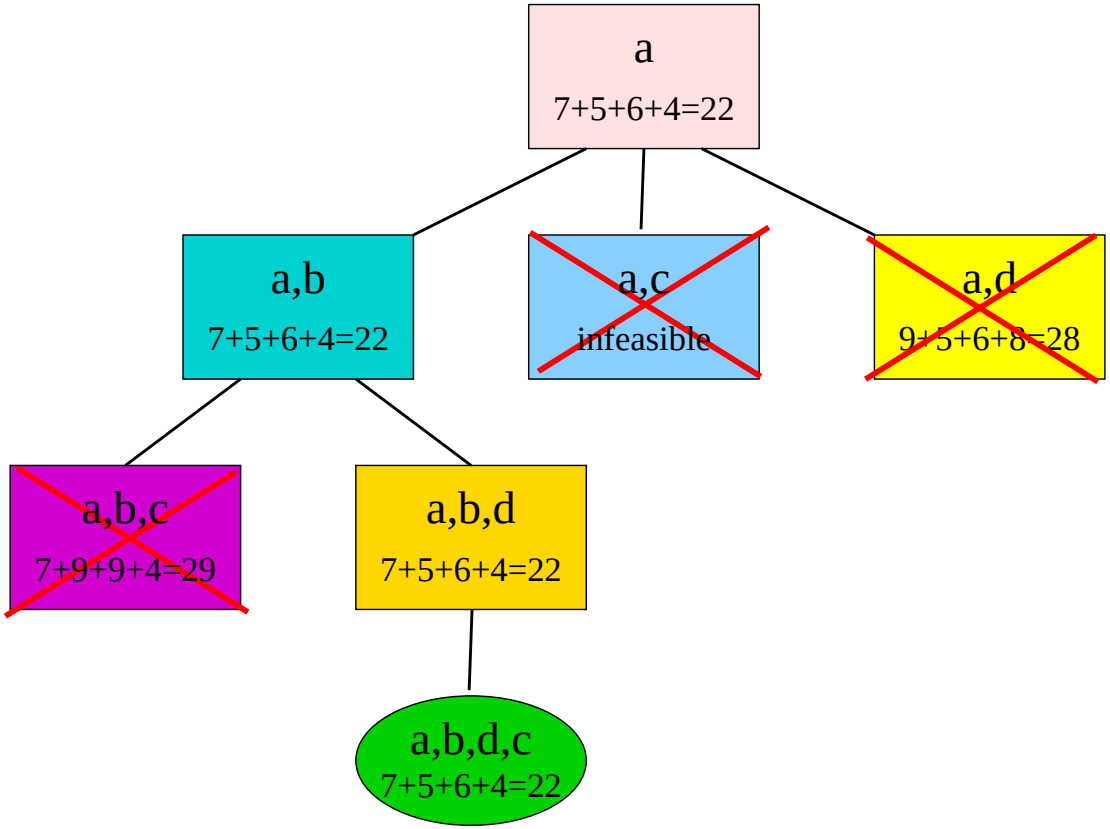
Opmerking: delen door 2 is niet nodig; je kunt ook 2× lengte minimaliseren (*). Opm. Er staan een paar kleine foutjes in bovenstaand plaatje: bijv. de ondergrens bij de knoop a,b,d is 34 en niet 22.

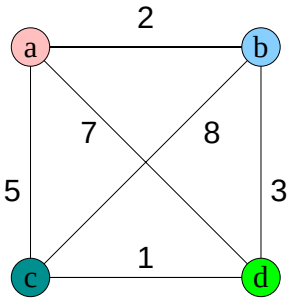
Nog een voorbeeld, met als ondergrens wederom de som van de twee kortste takken per knoop (gedeeld door 2).



De optimale oplossing heeft lengte 11: a, b, d, c, a.

Exercise 12.2.9, vervolg





- donderdag 10 mei 2007 in zaal 306/308

- **Opgaven:**

  zie http://www.liacs.nl/home/graaf/ALGO/algo2007.html

- **Volgende colleges:**

  vrijdag 11 mei 2007
  vrijdag 25 mei 2007