T–79.161 Kombinatoriset algoritmit

Harri Haanpää

Tietojenkäsittelyteorian laboratorio, TKK

Kevät 2005

T–79.161 Kombinatoriset algoritmit

Combinatorial:
1: of, relating to, or involving combinations
2: of or relating to the arrangement of, operation on, and selection of discrete mathematical elements belonging to finite sets or making up geometric configurations

Algorithm:
a procedure for solving a mathematical problem (as of finding the greatest common divisor) in a finite number of steps that frequently involves repetition of an operation; broadly : a step-by-step procedure for solving a problem or accomplishing some end especially by a computer

Kombinatorisia rakenteita

Lista: kokoelma alkioita tietyssä järjestyksessä, esim. $X = [0, 1, 3, 0]$

Joukko: kokoelma eri alkioita, joita ei ole järjestetty, esim. $X = \{1, 3, 4\}$. $|X|$ on X:n alkioiden lkm. Karteesinen tulo $X \times Y = \{[x, y] \,|\, x \in X \wedge y \in Y\}$.

Osajoukko: Joukko X on joukon Y osajoukko, jos kaikille $x \in X$ myös $x \in Y$. Jos $|X| = k$, X on Y:n k-osajoukko.

Graafi: $G = (\mathcal{V}, \mathcal{E})$, missä $\mathcal{V}$ on solmujen ja $\mathcal{E}$ kaarien joukko. Kukin kaari on kahden solmun joukko.

Esimerkki: Latinalainen neliö

Joukkojärjestelmä: $(\mathcal{X}, \mathcal{B})$, missä $\mathcal{X}$ on perusjoukko ja $\mathcal{B}$ joukko $\mathcal{X}$:n osajoukkoja. (esim. $\mathcal{X}$:n ositus)

Latinalainen neliö: $n \times n$-taulukko, jonka joka rivissä ja sarakkeessa esiintyy kukin luvuista $\mathcal{Y} = \{1, \ldots, n\}$ tasan kerran, esim.

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

joukkojärjestelmänä: $\mathcal{X} = \mathcal{Y} \times \{1, 2, 3\}$,
$\mathcal{B} = \{\{(y_1, 1), (y_2, 2), (A_{y_1 y_2}, 3)\} \,|y_1, y_2 \in \mathcal{Y}\}$

Transversaalisommitelma $TD(n)$: $(\mathcal{X}, \mathcal{B})$, missä $|\mathcal{X}| = 3n$; $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2 \cup \mathcal{X}_3$; $|\mathcal{X}_i| = n$; $|B \cap \mathcal{X}_i| = 1$ kaikilla $B \in \mathcal{B}$ ja $1 \leq i \leq 3$; kaikilla $x \in \mathcal{X}_i$, $y \in \mathcal{X}_j$, $i \neq j$ on yksi $B \in \mathcal{B}$, jolle $\{x, y\} \subset B$.

Ongelmatyyppejä

- luettele tietynlaiset kombinatoriset rakenteet
  - luettele mahdolliset pokerikädet
- laske, montako tietynlaista rakennetta on
  - laske n bitin binäärisanat, joissa ei ole kahta peräkkäistä ykköstä
- etsi tietynlainen kombinatorinen rakenne
  - väritä graafin solmut 3 värillä siten, että naapurit väritetään eri väreillä

Hakuongelman variantteja

Selkäreppuongelma: Annetaan n esinettä, joiden painot ovat $w_1, \ldots w_n$ ja hyödyt $p_1, \ldots, p_n$. Reppuun voidaan pakata osajoukko $S \subseteq \{1, \ldots, n\}$, jos $\sum_{i \in S} w_i \leq M$, missä M on repun kapasiteetti. Tällöin saavutetaan hyöty $P(S) = \sum_{i \in S} p_i$.

1. Voidaanko reppuun pakata jokin esinevalikoima S, jolle $P(S) = P$?
   (NP-täydellinen päätösongelma!)
2. Konstruoi reppuun mahtuva esinevalikoima, jolle $P(S) = P$.
3. Mikä on suurin mahdollinen $P(S)$:n arvo?
4. Millä esinevalikoimalla saavutetaan suurin mahdollinen $P(S)$:n arvo?

Ratkaisustrategioita

Ahne algoritmi: rakennetaan ratkaisu valitsemalla joka askeleella “lyhytnäköisesti paras” vaihtoehto. Esim. selkäreppuongelmassa laitetaan reppuun tavaroita paino-hyöty-suhteen mukaisessa järjestyksessä kunnes reppu täyttyy.

Dynaaminen ohjelmointi: Kun optimiratkaisun osat ovat vastaavien osaongelmien optimaalisia ratkaisuja, ratkaistaan ensin osaongelmat ja yhdistellään niiden ratkaisut koko ongelman ratkaisuksi.

Hajoita ja hallitse: Jaetaan ongelma osaongelmiksi, ratkaistaan ne ja yhdistetään ratkaisut.

Peräytyvä haku: Käydään läpi koko kaikkien mahdollisten ratkaisujen vaihtoehtopuu.

Paikallinen haku: Koetetaan jotakin ratkaisua pienin askelin parantamalla löytää lopulta hyvä ratkaisu.

Tietorakenteita osajoukoille

Perusjoukon koko? Tarvittavat operaatiot? Alkion lisäys/poisto, jäsenyyden testaus, unioni, leikkaus, alkioiden lkm, alkioiden luetteleminen?

- (järjestetty) linkitetty lista alkioita
  - kun perusjoukko on suuri ja osajoukko pieni
- binääripuut
  - kun perusjoukko on suuri ja osajoukko pienehkö
- bittikarttaesitys
  - kun perusjoukko on pieni

  esim. $S = \{1, 3, 11, 16\} \subset \{0, \ldots, 16\}$ voidaan esittää bittijonona 01010000000100001, joka voidaan pilkkoa esim. 8 bitin sanoiksi taulukkoon:
  $A[0] = 01010000_2$, $A[1] = 00010000_2$,
  $A[2] = 10000000_2$

Tietorakenteita graafeille ja joukkojärjestelmille

1. Kaarien joukko
2. Insidenssimatriisi: matriisi, jonka rivit vastaavat solmuja ja sarakkeet kaaria; alkio on 1, jos vastaava solmu on vastaavan kaaren päätepiste
3. Naapuruusmatriisi: matriisi, jonka rivit ja sarakkeet vastaavat solmuja; alkio on 1, jos vastaavien solmujen välillä on kaari
4. Naapuriluettelo: Annetaan kunkin solmun naapurien joukko

1. ja 2. soveltuvat myös joukkojärjestelmille.

rank- ja unrank-funktiot

Järjestetään lottorivit. Monesko lottorivi 3, 8, 12, 14, 15, 32, 38 on? Mikä lottorivi on rivi numero 3937483?

Olkoon S joukko joitakin kombinatorisia rakenteita. Numeroidaan ne $0 \ldots |S| - 1$:

$$\mathrm{rank}: \ S \mapsto \{0, \ldots, |S| - 1\}$$

$$\mathrm{unrank}: \{0, \ldots, |S| - 1\} \mapsto S$$

$$\mathrm{rank}(s) = i \Leftrightarrow \mathrm{unrank}(i) = s$$

- satunnainen $s = \mathrm{unrank}(\mathrm{random}(0 \ldots |S| - 1))$
- kokonaisluku on kompakti esitystapa

$$\mathrm{successor}(s) = t \Leftrightarrow \mathrm{rank}(t) = \mathrm{rank}(s) + 1$$

$$\mathrm{successor}(s) = \mathrm{unrank}(\mathrm{rank}(s) + 1),$$

kun $\mathrm{rank}(s) < |S| - 1$

Rakenteet voidaan käydä läpi successor-funktiolla unrank (0):sta lähtien.

Listan leksikografinen järjestys

Määritellään järjestys listoille $l = [s_1, s_2, \ldots, s_n]$ ja $l' = [s'_1, s'_2, \ldots, s'_{n'}]$: Jos toinen lista on toisen alku, lyhempi edeltää pidempää. Muutoin etsitään pienin i, jolla $s_i \neq s'_i$. Jos $s_i \prec s'_i$, niin $l \prec l'$, ja kääntäen.

Esim. Tarkastellaan kolmen kirjaimen muodostamia listoja, merk. ['A','B','C'] ='ABC'.

Olkoon $S = \{$'A','B','C',...,'Ö'$\}$. Määritellään järjestys tavalliseen tapaan: $A \prec B$, jne.

rank_S('A') $= 0$, rank_S('N') $= 13$; $\mathrm{unrank}_S(7)$ ='H'.

Nyt järjestys on 'AAA'≺'AAB'≺ ... ≺'ÖÖÄ'≺'ÖÖÖ', ja tässä erikoistapauksessa saadaan $\mathrm{rank}([s_1 s_2 s_3]) = |S|^2\, \mathrm{rank}(s_1) + |S|\, \mathrm{rank}(s_2) + \mathrm{rank}(s_3)$. (vrt. lukujärjestelmä)

Osajoukkojen leksikografinen järjestys

T	$\chi(T)$	rank(T)
$\emptyset$	$[0,0,0]$	0
$\{3\}$	$[0,0,1]$	1
$\{2\}$	$[0,1,0]$	2
$\{2,3\}$	$[0,1,1]$	3
$\{1\}$	$[1,0,0]$	4
$\{1,3\}$	$[1,0,1]$	5
$\{1,2\}$	$[1,1,0]$	6
$\{1,2,3\}$	$[1,1,1]$	7

Tutkitaan joukon $S = \{1, \ldots, n\}$ osajoukkoja. Kun $T \subseteq S$, olkoon T:n karakteristinen vektori $\chi(T) = [x_{n-1}, \ldots, x_0]$, missä $x_i = 1$, jos $n - i \in T$, ja $x_i = 0$, jos $n - i \notin T$. (vrt. bittikarttaesitys!) Järjestetään osajoukot karakterististen vektorien mukaisesti leksikografiseen järjestykseen.

$$\text{rank}(T) = \sum_{i=0}^{n-1} x_i 2^i$$

Osajoukon leksikografinen rank-funktio

```
SUBSETLEXRANK(n, T)
r ← 0
for i ← 1 to n
  if i ∈ T
    r ← r + 2^(n−i)
return r
```

Esim. rank$(\{1,3,4,6\})$:

i	$i \in T$	2^{n-i}	r
1	true	128	128
2	false	64	128
3	true	32	160
4	true	16	176
5	false	8	176
6	true	4	180
7	false	2	180
8	false	1	180

Osajoukon leksikografinen unrank -funktio

```
SUBSETLEXUNRANK(n, r)
T ← ∅
for i ← n downto 1
  if r mod 2 = 1
    T ← T ∪ {i}
  r ← ⌊r/2⌋
return T
```

Esim. unrank(180):

i	r	mod 2	T
8	180	0	$\emptyset$
7	90	0	$\emptyset$
6	45	1	{6}
5	22	0	{6}
4	11	1	{4,6}
3	5	1	{3,4,6}
2	2	0	{3,4,6}
1	1	1	{1,3,4,6}

Jos perusjoukko ei ole $\{1, \ldots, n\}$ (esim. $\{0, \ldots, n-1\}$), voi kannattaa kuvata perusjoukko $\{1, \ldots, n\}$:lle.

Minimimuutosjärjestys

Toisinaan toivottavaa: kaksi peräkkäistä rakennetta eroavat mahdollisimman vähän. Osajoukoille etäisyys voi olla

$$\text{dist}(T_1, T_2) = |T_1 \Delta T_2|, \text{ missä}$$
$$T_1 \Delta T_2 = (T_1 \setminus T_2) \cup (T_2 \setminus T_1).$$

Esim. joukkojen subsetlexunrank$(n, 3) = \{2, 3\}$ ja subsetlexunrank$(n, 4) = \{1\}$, etäisyys on 3, kun $n = 3$.

Osajoukoilla on olemassa järjestys, jossa peräkkäisten joukkojen dist on aina 1. Sellaisen järjestyksen karakteristiset vektorit muodostavat Gray-koodin.

Gray-koodit

Gray-koodi on 2^n n-bittisen binäärisanan lista, jossa peräkkäisten sanojen Hamming-etäisyys on 1.

Eräs mukava Gray-koodiperhe:
$G_1 = [0, 1]$, G_{i+1} saadaan G_i:stä ottamalla siitä kaksi kopiota, lisäämällä ensimmäisen kopion koodisanojen alkuun 0 ja toisen 1, kääntämällä jälkimmäisen kopion järjestys ja liittämällä kopiot yhteen:

$$G_2 = \begin{array}{|c|c|} \hline 0 & 0 \\ 0 & 1 \\ \hline 1 & 1 \\ 1 & 0 \\ \hline \end{array} \qquad G_3 = \begin{array}{|c|c|} \hline 0 & 00 \\ 0 & 01 \\ 0 & 11 \\ 0 & 10 \\ \hline 1 & 10 \\ 1 & 11 \\ 1 & 01 \\ 1 & 00 \\ \hline \end{array}$$


```
GrayCodeSuccessor(n, T)
if |T| is even
  return T Δ {n}
else if max(T) > 1
  return T Δ {max(T) − 1}
else
  return undefined
```

Olkoon koodisanan rank-luvun binääriesitys $b_{n-1} \dots b_0$ ja vastaavan koodisanan $a_{n-1} \dots a_0$.

$a_j = \left(b_j + b_{j+1}\right) \bmod 2$ ja $b_j = \sum_{i=j}^{n-1} a_i \bmod 2$.

k alkion osajoukkojen leksikografinen järjestys

$S = \{1, \dots, n\}$. Generoidaan kaikki $\binom{n}{k}$ osajoukkoa, joissa on k alkiota.
Esitetään $T \subseteq S$ listana: $\overrightarrow{T} = [t_1, \dots t_k]$, $t_i < t_{i+1}$, ja järjestetään osajoukot näiden listojen leksikografiseen järjestykseen.

T	$\overrightarrow{T}$	rank (T)
$\{1,2,3\}$	$[1,2,3]$	0
$\{1,2,4\}$	$[1,2,4]$	1
$\{1,2,5\}$	$[1,2,5]$	2
$\{1,3,4\}$	$[1,3,4]$	3
$\{1,3,5\}$	$[1,3,5]$	4
$\{1,4,5\}$	$[1,4,5]$	5
$\{2,3,4\}$	$[2,3,4]$	6
$\{2,3,5\}$	$[2,3,5]$	7
$\{2,4,5\}$	$[2,4,5]$	8
$\{3,4,5\}$	$[3,4,5]$	9

Seuraaja: kasvatetaan suurinta alkiota, jota voi kasvattaa, ja asetetaan sitä suuremmat alkiot minimiarvoihinsa.

rank $(T) = \sum_{i=1}^{k} \sum_{j=t_{i-1}+1}^{t_i - 1} \binom{n-j}{k-i}$, missä $t_0 = 0$.

k alkion osajoukkojen co-lex-järjestys

T	$\overleftarrow{T}$	rank (T)
$\{1,2,3\}$	$[3,2,1]$	0
$\{1,2,4\}$	$[4,2,1]$	1
$\{1,3,4\}$	$[4,3,1]$	2
$\{2,3,4\}$	$[4,3,2]$	3
$\{1,2,5\}$	$[5,2,1]$	4
$\{1,3,5\}$	$[5,3,1]$	5
$\{2,3,5\}$	$[5,3,2]$	6
$\{1,4,5\}$	$[5,4,1]$	7
$\{2,4,5\}$	$[5,4,2]$	8
$\{3,4,5\}$	$[5,4,3]$	9

$S = \{1, \dots, n\}$. Generoidaan kaikki $\binom{n}{k}$ osajoukkoa, joissa on k alkiota.
Esitetään $T \subseteq S$ listana: $\overleftarrow{T} = [t_1, \dots t_k]$, $t_i > t_{i+1}$, ja järjestetään osajoukot näiden listojen leksikografiseen järjestykseen.
Seuraaja: kasvatetaan pienintä alkiota, jota voi kasvattaa; minimoidaan sitä pienemmät alkiot.
rank $(T) = \sum_{i=1}^{k} \binom{t_i - 1}{k+1-i}$
rank on riippumaton n:stä!

Lex- ja co-lex-järjestysten yhteys

Kuvataan $T \subseteq \{1, \ldots n\}$ joukoksi $T' = \{n+1-t | t \in T\}$. T:n lex-järjestys on T':n käänteinen colex-järjestys, ja kääntäen!

T	T'	$\text{rank}_L(T)$	$\text{rank}_C(T')$
$\{1,2,3\}$	$\{5,4,3\}$	0	9
$\{1,2,4\}$	$\{5,4,2\}$	1	8
$\{1,2,5\}$	$\{5,4,1\}$	2	7
$\{1,3,4\}$	$\{5,3,2\}$	3	6
$\{1,3,5\}$	$\{5,3,1\}$	4	5
$\{1,4,5\}$	$\{5,2,1\}$	5	4
$\{2,3,4\}$	$\{4,3,2\}$	6	3
$\{2,3,5\}$	$\{4,3,1\}$	7	2
$\{2,4,5\}$	$\{4,2,1\}$	8	1
$\{3,4,5\}$	$\{3,2,1\}$	9	0

Tämän muunnoksen kautta on tehokkaampaa laskea lex-järjestyksen rank ja unrank.

Esimerkki k-osajoukon rank:sta

Järjestetään lottorivit leksikografiseen järjestykseen. Monesko lottorivi 3, 8, 12, 14, 15, 32, 38 on?

$$T = \{3,8,12,14,15,32,38\} \subseteq \{1, \ldots 39\}.$$
$$T' = \{37,32,28,26,25,8,2\}.$$

$$\begin{aligned}\text{rank}_C(T') &= \binom{37-1}{7} + \binom{32-1}{6} + \binom{28-1}{5} \\ &+ \binom{26-1}{4} + \binom{25-1}{3} + \binom{8-1}{2} + \binom{2-1}{1} \\ &= 9179387\end{aligned}$$

$$\begin{aligned}\text{rank}_L(T) &= \binom{39}{7} - 1 - \text{rank}_C(T') \\ &= 6201549\end{aligned}$$

Esimerkki k-osajoukon unrank:sta

Järjestetään lottorivit leksikografiseen järjestykseen. Mikä lottorivi on rivi numero 3937483?

$3937482 = \text{rank}_L(T) = \binom{39}{7} - 1 - \text{rank}_C(T')$

$T' = \text{unrank}_C\left(\binom{39}{7} - 1 - 3937482\right)$

i	r	t_i s.t. $\binom{t_i-1}{i} \le r < \binom{t_i}{i}$	$r - \binom{t_i-1}{i}$
7	11443454	38	1147982
6	1147982	33	40414
5	40414	24	6765
4	6765	22	780
3	780	18	100
2	100	15	9
1	9	10	0

$$T' = \{38,33,24,22,18,15,10\},$$
$$T = \{2,7,16,18,22,25,30\}$$

Permutaatiot

Permutaatio on tapa järjestää alkiot $\{1, \ldots, n\}$ eli bijektio joukolta $\{1, \ldots, n\}$ itselleen.

$$\pi : \{1, \ldots, n\} \mapsto \{1, \ldots, n\}$$

Esim.

x	1	2	3	4	5	6
$\pi(x)$	3	5	1	4	6	2

Permutaatio voidaan esittää listana: $[\pi(1), \pi(2), \ldots, \pi(n)]$

Esim. $[3,5,1,4,6,2]$.

Permutaatio voidaan esittää syklinotaatiossa, jossa suluissa aina edeltävä alkio kuvautuu seuraavalle ja viimeinen ensimmäiselle, esim.

$$\pi = (1\,3)(2\,5\,6)(4) = (1\,3)(2\,5\,6)$$

Permutaatioiden yhdistely

Permutaatiot ovat funktioita ja niitä yhdistellään kuin funktioita. Yhdistely tapahtuu siksi oikealta vasemmalle.

$$(\pi_1\pi_2)(x) = (\pi_1 \circ \pi_2)(x) = \pi_1(\pi_2(x))$$

$$\begin{aligned}(1\,2)(2\,3) &= (1\,2\,3)\\(2\,3)(1\,2) &= (1\,3\,2)\end{aligned}$$

Permutaatioiden parillisuus

Yksinkertaisin permutaatio on transpositio $(i\,j)$, missä $i \neq j$. Permutaatiot voidaan jakaa kahteen luokkaan.

Parilliset permutaatiot voidaan ilmaista vain parillisen määrän transpositioita tulona, esim.

$$(1\,2\,3) = (1\,2)(2\,3)$$

Parittomat permutaatiot voidaan ilmaista vain parittoman määrän transpositioita tulona, esim.

$$(1\,2\,3\,4) = (1\,2)(2\,3)(3\,4)$$

Aputulos

Jos listat $d = [d_1, \ldots, d_n]$, missä $0 \leq d_i < n_i$ järjestetään leksikografiseen järjestykseen, niin

$$\text{rank}(d) = \sum_{i=1}^{n} d_i \prod_{j=i+1}^{n} n_j$$

unrank(r):
for $i = n$ downto 1:
  $d_i \leftarrow r \bmod n_i$
  $r \leftarrow \left\lfloor \frac{r}{n_i} \right\rfloor$

successor(d):
$i = n$
while $d_i = n_i - 1$
  $i \leftarrow i - 1$
$d_i \leftarrow d_i + 1$
for $j = i + 1$ to n
  $d_j = 0$

Permutaatioiden leksikografinen rank

Järjestetään permutaatiot listaesitystensä leksikografiseen järjestykseen. Esim.

$$[1, 2, 3], [1, 3, 2], [2, 1, 3], [2, 3, 1], [3, 1, 2], [3, 2, 1]$$

Valittaessa listan i:nnettä alkiota on $n + 1 - i$ vaihtoehtoa. Merkitään d_i:llä, montako valittua pienempää vaihtoehtoa oli. Nyt $0 \leq d_i < n_i = n + 1 - i$, ja

$$\text{rank}(\pi) = \sum_{i=1}^{n-1} d_i (n - i)!$$

Esim. $\pi = [2, 4, 1, 3] \Rightarrow d = [1, 2, 0, 0]$ ja

$$\text{rank}(\pi) = 1 \cdot 3! + 2 \cdot 2! + 0 \cdot 1! = 10$$

Permutaatioiden leksikografinen unrank ja seuraaja

Vastaavasti unrank (10) antaa ensin $d = [1, 2, 0, 0]$, ja siitä saadaan $\pi = [2, 4, 1, 3]$.

Seuraaja: Pyritään olemaan koskematta alkupään alkioihin; etsitään listan lopusta lyhin osalista, joka ei ole käänteisessä lex. järjestyksessä. Vaihdetaan osalistan ensimmäinen alkio seuraavan suuremman alkion kanssa, ja käännetään osalistan loppu kasvavaan järjestykseen. Esim.
$[3, 6, \underline{2, 7, 5, 4, 1}] \to [3, 6, 4, 7, 5, 2, 1] \to [3, 6, 4, 1, 2, 5, 7]$

Permutaatioiden minimimuutosjärjestys: Trotter–Johnson

Permutaatioiden minimimuutos on vierekkäisten alkioiden transpositio: $[\ldots, i, j, \ldots] \to [\ldots, j, i, \ldots]$.

Trotter (1962): lisätään alkioiden $\{1, \ldots, n-1\}$ minimimuutosjärjestykseen T^{n-1} alkio n seuraavasti. Kopioidaan kukin T^{n-1}:n alkio n-kertaisesti, ja lisätään sopiviin väleihin alkio n niin, että n:n paikat muodostavat siksak-kuvion.
$T^1 = [1]$, $T^2 = [[1, 2], [2, 1]]$

$$T^3 = \begin{bmatrix} & 1 & & 2 & 3 \\ & 1 & 3 & 2 & \\ 3 & 1 & & 2 & \\ 3 & 2 & & 1 & \\ & 2 & 3 & 1 & \\ & 2 & & 1 & 3 \end{bmatrix}$$

Menetelmä on tunnettu jo 1600-luvun Englannissa kirkonkellojen soitossa nimellä plain changes.

$$T^3 = \begin{bmatrix} [1, 2, 3] \\ [1, 3, 2] \\ [3, 1, 2] \\ [3, 2, 1] \\ [2, 3, 1] \\ [2, 1, 3] \end{bmatrix}$$

$$T^4 = \begin{bmatrix}
 & 1 & & 2 & & 3 & 4 \\
 & 1 & & 2 & 4 & 3 & \\
 & 1 & 4 & 2 & & 3 & \\
4 & 1 & & 2 & & 3 & \\
4 & 1 & & 3 & & 2 & \\
 & 1 & 4 & 3 & & 2 & \\
 & 1 & & 3 & 4 & 2 & \\
 & 1 & & 3 & & 2 & 4 \\
 & 3 & & 1 & & 2 & 4 \\
 & 3 & & 1 & 4 & 2 & \\
 & 3 & 4 & 1 & & 2 & \\
4 & 3 & & 1 & & 2 & \\
4 & 3 & & 2 & & 1 & \\
 & 3 & 4 & 2 & & 1 & \\
 & 3 & & 2 & 4 & 1 & \\
 & 3 & & 2 & & 1 & 4 \\
 & 2 & & 3 & & 1 & 4 \\
 & 2 & & 3 & 4 & 1 & \\
 & 2 & 4 & 3 & & 1 & \\
4 & 2 & & 3 & & 1 & \\
4 & 2 & & 1 & & 3 & \\
 & 2 & 4 & 1 & & 3 & \\
 & 2 & & 1 & 4 & 3 & \\
 & 2 & & 1 & & 3 & 4
\end{bmatrix}$$

Trotter–Johnson rank

TJRank(π, n):
$\pi' \leftarrow \pi$ ilman alkiota n
$r \leftarrow$ TJRank $(\pi', n-1)$
jos r parillinen:
  $r \leftarrow nr +$ alkioiden lkm n : n oik. puolella
muuten:
  $r \leftarrow nr +$ alkioiden lkm n : n vas. puolella
return r

Esim. TJRank$([3, 4, 2, 1], 4)$:
  $r =$TJRank$([3, 2, 1], 3)$
    $r =$TJRank$([2, 1], 2) = 1$
    r pariton; $r \leftarrow 3 \cdot 1 + 0 = 3$
  r pariton; $r \leftarrow 4 \cdot 3 + 1 = 13$

Trotter–Johnson unrank

TJUnrank(r,n):
$\pi' \leftarrow$ TJUnrank $\left(\left\lfloor \frac{r}{n} \right\rfloor, n-1\right)$
$r \leftarrow r$ mod n
jos π' parillinen:
  $\pi \leftarrow \pi'$, johon lisätään n s.e. sen oik. puolelle jää r alkiota
muuten:
  $\pi \leftarrow \pi'$, johon lisätään n s.e. sen vas. puolelle jää r alkiota
return π

Esim. TJUnrank($13,4$):
  TJUnrank($3,3$):
    TJUnrank($1,2$) $=[2,1]$
  1 pariton: lisätään alkio 3 s.e. sen vas. puolelle jää
  3 mod $3=0$ alkiota: $[3,2,1]$
3 pariton: lisätään alkio 4 s.e. sen vas. puolelle jää 13 mod $4=1$ alkiota: $[3,4,2,1]$

Trotter–Johnson successor

TJSuccessor(π,n):
$\pi' \leftarrow \pi$ ilman alkiota n
jos π' on parillinen ja n:ä voidaan siirtää vasempaan, tehdään niin
muuten jos π' on pariton ja n:ä voidaan siirtää oikealle, tehdään niin
muuten lasketaan TJSuccessor($\pi', n-1$) pitäen n:ä paikallaan

Esim. TJSuccessor($[4,3,1,2]$):

$\pi'=[3,1,2]$ on parillinen, mutta 4:ä ei voi siirtää vasempaan; lasketaan $[4]+$TJSuccessor($[3,1,2]$):

$\pi'=[1,2]$ on parillinen, mutta 3:a ei voi siirtää vasempaan; lasketaan $[3]+$TJSuccessor($[1,2]$):

$\pi'=[1]$ on parillinen, ja 2:a voidaan siirtää vasempaan: $[2,1]$

saadaan $[4]+[3]+[2,1]=[4,3,2,1]$

Myrvold & Ruskey: unrank

Sen sijaan, että valittaisiin jokin järjestys, jolle laskettaisiin sitten rank- ja unrank-funktioita, Myrvold ja Ruskey valitsivat nopean unrank-funktion ja kehittivät sitä vastaavan rank-funktion.

Perinteinen tapa luoda satunnainen permutaatio:

for $i=n$ downto 1
  swap($\pi(i), \pi(r_i)$)

missä $r_i=$ random $(1,\ldots,i)$. Syntyy permutaatio

$$\pi=(n\ r_n)(n-1\ r_{n-1})\ldots(2\ r_2)(1\ r_1)$$

(tässä merkitään $(i\ i)=(i)$ yksinkertaisuuden vuoksi). Jokaisella permutaatiolla on yksikäsitteinen esitystapa tässä muodossa.

Unrank on yksinkertainen: päätellään rank:sta r_i:ien arvot ja konstruoidaan permutaatio ylläolevalla algoritmilla.

Myrvold & Ruskey: rank

Rank:n laskemiseksi pitää ensin päätellä r_i:t ja sitten laskea

$$\operatorname{rank}(\pi)=\sum_{i=1}^{n}(r_i-1)(i-1)!$$

Koska π:n edellisen sivun esitysmuodossa alkiota n permutoidaan vain vasemmanpuoleisessa transpositiossa, $\pi(n)=r_n$. Näin π:stä voidaan helposti päätellä r_n. Sitten lasketaan $(n\ r_n)\pi$, jossa n kuvautuu itselleen ja oleellisesti jää käsiteltäväksi joukon $\{1,\ldots,n-1\}$ permutaatio, ja iteroidaan.

Järjestys ei ole kovin intuitiivinen:

0 : 2341	6 : 4123	12 : 3241	18 : 1423
1 : 4312	7 : 3124	13 : 3412	19 : 1324
2 : 2413	8 : 2431	14 : 4213	20 : 4231
3 : 2314	9 : 4132	15 : 3214	21 : 1432
4 : 3421	10 : 2143	16 : 4321	22 : 1243
5 : 3142	11 : 2134	17 : 1342	23 : 1234

Kokonaislukujen ositus

$P(m)$: monellako tavalla positiivinen kokonaisluku m voidaan esittää positiivisten kokonaislukujen summana $m = a_1 + \ldots + a_n$, kun summattavien järjestyksellä ei ole merkitystä? (tai vastaavasti $a_1 \geq \ldots \geq a_n$)

$P(5) = 7$: 5, 4+1, 3+2, 3+1+1, 2+2+1
2+1+1+1, 1+1+1+1+1

$P(1) = 1$, $P(2) = 2$, $P(3) = 3$, $P(4) = 5$, $P(5) = 7$, $P(6) = 11$,
$P(m) \sim \Theta\left(e^{\pi\sqrt{2m/3}}/m\right)$

Ositusten generointi

```
GenRecPartition(m, B, N)
    if m = 0
        output ([a_1, ..., a_N])
    else
        for i = 1 to min(B, m):
            a_{N+1} ← i
            GenRecPartition(m − i, i, N + 1)
RecPartition(m, m, 0)
```

Parametri m on ositettava kokonaisluku, parametri B on suurin kokonaisluku, joka voidaan laittaa seuraavaksi kiinnitettävään a_i:hin rikkomatta suuruusjärjestystä, ja N on jo kiinnitettyjen arvojen lkm.

Ferrers-Young-kaaviot

Osituksen Ferrers-Young-kaavio saadaan kirjoittamalla kutakin a_i:tä vastaava määrä pisteitä omalle rivilleen.

$$7 = 4 + 2 + 1 \Rightarrow D = \begin{array}{llll} \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & & \\ \bullet & & & \end{array}$$

Kääntämällä rivit sarakkeiksi saadaan vastaava konjugaattikaavio ja konjugaattiositus:

$$D^* = \begin{array}{lll} \bullet & \bullet & \bullet \\ \bullet & \bullet & \\ \bullet & & \\ \bullet & & \end{array} \Rightarrow 7 = 3 + 2 + 1 + 1$$

$P(m, n)$:
m:n osituksia, joissa on n osaa, on yhtä monta kuin m:n osituksia, joissa suurin osa on n.

Relaatio I

Selvästi $P(m,m) = P(m,1) = 1$, kun $m > 1$; määritellään $P(m,0) = 0$ ja $P(0,0) = 1$.

Teoreema 3.2: Kun $m \geq n > 0$,

$$P(m,n) = P(m-1,n-1) + P(m-n,n).$$

Tod. Merkitään $\mathcal{P}(m,n)$:llä m:n n-ositusten joukkoa. Jaetaan $\mathcal{P}(m,n)$ kahtia ja määritellään kuvaukset:

jos $a_n = 1$, $\Phi_1([a_1,\ldots,a_n]) = [a_1,\ldots,a_{n-1}]$;
jos $a_n > 1$, $\Phi_2([a_1,\ldots,a_n]) = [a_1-1,\ldots,a_n-1]$

Φ_1 ja Φ_2 ovat bijektioita $\mathcal{P}(m,n)$:n osilta joukoille $\mathcal{P}(m-1,n-1)$ ja $\mathcal{P}(m-n,n)$, joten joukkojen alkioiden lukumäärät ovat identtiset.

Relaatioita II

Teoreema 3.3:

$$P(m,n) = \sum_{i=0}^{n} P(m-n,i)$$

Tod: Jaetaan $\mathcal{P}(m,n)$ osiin $\mathcal{P}(m,n)_i$, joista kuhunkin kuuluvat ositukset, joissa on tasan i osaa, jotka ovat suurempia kuin 1.

Kullekin $0 \leq i \leq n$ määritellään bijektio

$$\Phi_i : \mathcal{P}(m,n)_i \mapsto \mathcal{P}(m-n,i)$$

seuraavasti:

$$\Phi_i([a_1,\ldots,a_n]) = [a_1-1,\ldots a_i-1]$$

Rank-funktio $\mathcal{P}(m,n)$:lle

Järjestetään ositukset $[a_1,\ldots,a_n]$ käänteisen standardimuodon $[a_n,\ldots,a_1]$ mukaiseen leksikografiseen järjestykseen. Esim. $\mathcal{P}(10,4)$:

std. muoto	käänt. std. muoto
$[7,1,1,1]$	$[1,1,1,7]$
$[6,2,1,1]$	$[1,1,2,6]$
$[5,3,1,1]$	$[1,1,3,5]$
$[4,4,1,1]$	$[1,1,4,4]$
$[5,2,2,1]$	$[1,2,2,5]$
$[4,3,2,1]$	$[1,2,3,4]$
$[3,3,3,1]$	$[1,3,3,3]$
$[4,2,2,2]$	$[2,2,2,4]$
$[3,3,2,2]$	$[2,2,3,3]$

Ositukset, joissa $a_n = 1$, tulevat tässä järjestyksessä ennen niitä, joissa $a_n > 1$. Saadaan

$$\operatorname{rank}([a_1,\ldots,a_n]) = \begin{cases} \operatorname{rank}([a_1,\ldots,a_{n-1}]) & \text{jos } a_n = 1 \\ \operatorname{rank}([a_1-1,\ldots,a_n-1]) + P(m-1,n-1) & \text{jos } a_n > 1 \end{cases}$$

Seuraajafunktio $\mathcal{P}(m,n)$:ssä

Partitio $[a_1,\ldots,a_n]$ on viimeinen, kun $a_1 \leq a_n + 1$. Tällöin m on jaettu mahdollisimman tasan n:llä.

Valitussa järjestyksessä koetetaan pitää viimeiset alkiot mahdollisimman muuttumattomina.

Seuraajafunktio:

1. etsitään listan ensimmäinen osalista, joka ei ole tasan jaettu, ts. pienin i, jolle $a_1 > a_i + 1$.
2. Kasvatetaan a_i:tä yhdellä ja asetetaan $a_2,\ldots,a_{i-1}$ minimiarvoonsa ($= a_i$)
3. Täsmätään summa asettamalla $a_1 = m - \sum_{i=2}^{n} a_i$.

Esim.:
$[5,5,4,2,1]$:lle $i = 4$. Asetetaan $a_4 = a_4 + 1 = 3$, $a_3 = 3$, $a_2 = 3$ ja $a_1 = 17 - 3 - 3 - 3 - 1 = 7$; saadaan $[7,3,3,3,1]$.

Nimetyt puut

Graafi $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ on puu, kun se on yhtenäinen ja syklitön. Solmun v asteluku on niiden kaarien määrä, joissa solmu esiintyy. Olkoon $\mathcal{V} = \{1, 2, \dots, n\}$. Tällöin on n^{n-2} eri puuta, joiden solmujoukko on $\mathcal{V}$.

Olkoon $\mathcal{T}_n$ niiden puiden joukko, joiden solmujoukko on $\mathcal{V}$. Prüferin vastaavuus:

$$\text{Prüfer} : \mathcal{T}_n \mapsto \mathcal{V}^{n-2}$$

$$\text{Prüfer}^{-1} : \mathcal{V}^{n-2} \mapsto \mathcal{T}_n$$

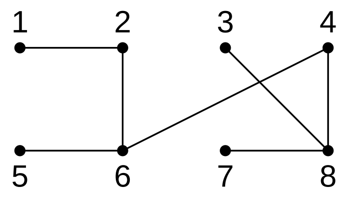
Prüfer

PRÜFER$(n, \mathcal{E})$:
for $i = 1$ to $n - 2$:
  olkoon v korkeanumeroisin solmu, jonka asteluku=1
  etsitään kaari $\{v, v'\} \in \mathcal{E}$ ja asetetaan $L_i \leftarrow v'$
  poistetaan graafista kaari $\{v, v'\}$

Kukin solmu v esiintyy listassa L deg$(v) - 1$ kertaa. Graafiin jää kaari $\{1, v\}$, missä v:n asteluku on 1 algoritmin ajamisen jälkeen.

INVPRÜFER(n, L):
lasketaan listasta L solmujen asteluvut
lisätään listan jatkoksi ykkönen: $L_{n-1} \leftarrow 1$
for $i = 1$ to $n - 1$:
  olkoon v korkeanumeroisin solmu, jonka asteluku=1
  lisätään graafiin kaari $\{v, L_i\}$
  vähennetään v:n ja L_i:n astelukua yhdellä

Prüfer-esimerkki



Solmujen asteluvut	L_i	Kaari
[1,2,1,2,1,3,1,3]	8	{7,8}
[1,2,1,2,1,3,0,2]	6	{5,6}
[1,2,1,2,0,2,0,2]	8	{3,8}
[1,2,0,2,0,2,0,1]	4	{4,8}
[1,2,0,1,0,2,0,0]	6	{4,6}
[1,2,0,0,0,1,0,0]	2	{2,6}
[1,1,0,0,0,0,0,0]		

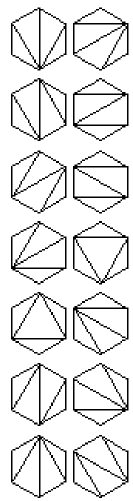
Listaesityksestä saadaan helposti rank- ja unrank-funktiot tulkitsemalla lista n-kantaisena lukuna.

Catalanin luvut

$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

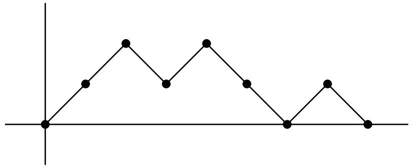
Catalanin luvut esiintyvät monessa yhteydessä:

- monellako eri tavalla matriisitulolauseke voidaan suluttaa niin, että aina kaksi tekijää kerrotaan kerrallaan: $((M_1\,(M_2M_3))\,(M_4M_5))$
- monellako eri tavalla konveksi $n+2$-kulmio voidaan kolmioida
- montako sellaista $2n$ bitin jonoa on, joissa on n ykköstä ja n nollaa, ja joissa minkään kohdan vasemmalla puolella ei ole enemmän ykkösiä kuin nollia: 000111, 001011, 001101, 010011, 010101

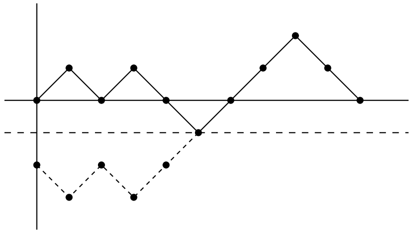


Catalanin luvuista

Em. kaltaiset binääriluvut voidaan esittää vuoristona, joka ei laske 0-tason alle. Esim. $a = 00101101$ vastaa vuoristoa



Peilataan ne vuoristot, jotka laskevat 0-tason alle, akselin $y = -1$ yli alusta siihen asti, jossa ne ensimmäisen kerran laskevat 0-tason alle. Saadaan bijektio 0-tason alle laskevien ja $(-2, 0)$:sta $(2n, 0)$:aan kulkevien vuoristojen välille.



$$C_n = \binom{2n}{n} - \binom{2n}{n+2} = \left(1 - \frac{n}{n+1}\right)\binom{2n}{n} = \frac{1}{n+1}\binom{2n}{n}$$

Catalan-rank ja -unrank

Lasketaan, monellako tavalla vuoriston voi saattaa loppuun kustakin kohdasta, esim. tapaus C_5:

5						1					
4					5		1				
3				14		4		1			
2			28		9		3		1		
1		42		14		5		2		1	
0	42		14		5		2		1		1
	0	1	2	3	4	5	6	7	8	9	10

rank: seurataan vuoristoa; kun se menee oikealle alas, lisätään rank:iin luku, joka olisi ollut oikealla ylhäällä.

unrank: jos rank $\geq$ oikealla ylhäällä oleva luku, mennään oikealle alas ja vähennetään oikealla ylhäällä ollut luku rank:sta; muuten mennään oikealle ylös

Esim. rank $(0010110101) = 22$

Peräytyvä haku

- yleinen menetelmä kombinatorisen haku-, optimointi-, generointi- tai enumerointiongelman ratkaisemiseen.
- rekursiivinen: toteutetaan tyypillisesti itseään kutsuvilla aliohjelmilla, jotka rakentavat ratkaisun askel askeleelta
- täydellinen haku: koko ratkaisuavaruus käydään läpi
- karsinta säästää tarkastelemasta epäoleellisia hakuavaruuden osia

Esimerkki peräytyvästä hausta

Selkäreppuongelma: Annetaan n esinettä, joiden painot ovat $w_1, \ldots w_n$ ja hyödyt $p_1, \ldots, p_n$. Repun kapasiteetti on M. Maksimoidaan $P(x) = \sum p_i x_i$ rajoitusehdoilla $x_i \in \{0, 1\}$ ja $\sum w_i x_i \leq M$.

Konstruoidaan rekursiivisesti lista $[x_0, \ldots, x_{n-1}]$. Tässä l on jo kiinnitettyjen muuttujien x_i lukumäärä; rekursio käynnistetään kutsulla KNAPSACK1(0).

KNAPSACK1(l):
if $l = n$:
  if $\sum w_i x_i \leq M$:
    $CurP \leftarrow \sum p_i x_i$
    if $CurP > OptP$:
      $OptP \leftarrow CurP$
      $OptX \leftarrow$
$[x_0, \ldots x_{n-1}]$
else:
  $x_l = 1$
  KNAPSACK1($l + 1$)
  $x_l = 0$
  KNAPSACK1($l + 1$)

Peräytyvä haku yleisesti

Monen kombinatorisen ongelman ratkaisut voidaan esittää listana $X = [x_0, \ldots, x_{n-1}]$, missä $x_i \in P_i$, missä P_i on äärellinen x_i:n mahdollisten arvojen joukko. Peräytyvä haku konstruoi joukon $P_0 \times P_1 \times \ldots \times P_{n-1}$ kaikki alkiot. Haun aikana konstruoitavan listan pituus vastaa hakusolmun syvyyttä hakupuussa.

Osittainen ratkaisu $[x_0, \ldots, x_{l-1}]$ voi rajoittaa hakua; joskus voidaan päätellä, etteivät jotkin $x_l \in P_l$ voi johtaa käypiin ratkaisuihin. Tällöin voidaan karsia hakupuuta ja rajoittaa haku vaihtoehtojoukkoon $C_l \subseteq P_l$.

BACKTRACK(l):
jos $[x_0, \ldots, x_{l-1}]$ on käypä ratkaisu:
    käsittele se
laske C_l
kullekin $x \in C_l$:
    $x_l \leftarrow x$
    BACKTRACK($l+1$)

KNAPSACK2(l):
if $l = n$:
    if $CurP > OptP$:
        $OptP \leftarrow CurP$
        $OptX \leftarrow [x_0, \ldots x_{n-1}]$
if $l = n$:
    $C_l \leftarrow \emptyset$
else if $\sum_{i=0}^{l-1} w_i x_i + w_l \leq M$:
    $C_l \leftarrow \{0, 1\}$
else:
    $C_l \leftarrow \{0\}$
for $x \in C_l$:
    $x_{l+1} \leftarrow x$
    KNAPSACK2($l+1$)

Klikkien generointi

Klikki on graafin $G = (\mathcal{V}, \mathcal{E})$ solmujoukon $\mathcal{V}$ sellainen osajoukko $S \subseteq \mathcal{V}$, että kaikkien S:n solmuparien $x, y \in S$, $x \neq y$ välillä on kaari: $\{x, y\} \in \mathcal{E}$.

Maksimaalinen klikki on klikki, joka ei ole suuremman klikin osajoukko.

Määritellään peräytyvä haku:
$[x_0, \ldots, x_{l-1}]$ vastaa klikkiä $S_l = \{x_0, \ldots, x_{l-1}\}$

$$\begin{aligned} C_l &= \{v \in \mathcal{V} \setminus S_{l-1} : \{v, x\} \in \mathcal{E} \text{ kaikilla } x \in S_{l-1}\} \\ &= \{v \in C_{l-1} \setminus \{x_{l-1}\} : \{v, x_{l-1}\} \in \mathcal{E}\} \end{aligned}$$

Ongelma: algoritmi generoi kaikki k solmun klikit $k!$ kertaa, kerran kussakin järjestyksessä! Ratkaisu: määritellään solmuille järjestys $v_0 < \ldots < v_{n-1}$ ja valitaan

$$C_l = \{v \in C_{l-1} : \{v, x_{l-1}\} \in \mathcal{E} \wedge v > x_{l-1}\}$$

Klikkien generointi II

Esilasketaan aluksi kullekin solmulle v apujoukot $N_v = \{u \in \mathcal{V} : \{u, v\} \in E\}$ ja $G_v = \{u \in \mathcal{V} : u > v\}$. N_v on solmun v naapurien joukko ja G_v on valitussa järjestyksessä solmun v jälkeen tulevien solmujen joukko.

Haun aikana X on lista solmuja, jotka muodostavat klikin; N on X:n solmujen yhteisten naapurien joukko; ja C on yhteisten naapurien joukko, jotka ovat järjestyksessä viimeisen X:ään lisätyn solmun jälkeen.

ALLCLIQUES(X, N, C):
output X
if $N = \emptyset$:
    X on maksimaalinen
for $v \in C$:
    ALLCLIQUES($X + [v], N \cap N_v, C \cap N_v \cap G_v$)

ALLCLIQUES($[], \mathcal{V}, \mathcal{V}$)

Hakupuun koon arviointi

Jos hakupuun joka kohdassa $|C_i| = c_i$, puun koko on $|T| = 1 + c_0 + c_0c_1 + c_0c_1c_2 + \ldots + c_0c_1 \ldots c_{n-1}$. Yleensä näin ei ole. Kutsutaan hakupuun solmuja nimillä $[x_0, \ldots x_{l-1}]$ sen mukaan, mitä valintoja tekemällä niihin päästään. Puun kokoa voidaan arvioida valitsemalla joka askeleella tasajakautuneesti satunnainen vaihtoehto, jolloin todennäköisyys, että polku kulkee solmun X kautta, on

$$p(X) = \begin{cases} 1 & \text{kun } l = 0 \\ \frac{p(f(X))}{|C_{l-1}(f(X))|} & \text{kun } l > 0, \end{cases}$$

missä $f([x_0, \ldots x_{l-1}])$=$[x_0, \ldots, x_{l-2}]$ (solmun isä). Merkitään $m(X) = 1$, jos X kuuluu polkuun, ja $m(X) = 0$, jos ei. Arvioidaan puun kokoa laskemalla

$$N = \sum_{X \in P} \frac{1}{p(X)} = \sum_{X \in T} \frac{m(X)}{p(X)}.$$

Väite: $E(N) = |T|$. Todistus:

$$E(N) = E \sum_{X \in T} \frac{m(X)}{p(X)} = \sum_{X \in T} \frac{E(m(X))}{p(X)}$$

$$= \sum_{X \in T} \frac{p(X)}{p(X)} = \sum_{X \in T} 1 = |T|.$$

Täsmällinen peitto (Exact Cover)

Kun annetaan joukko R ja joukko S sen osajoukkoja, voidaanko valita osajoukkojen osajoukko, joka osittaa R:n?

$R = \{0, \ldots, n-1\}$. $S = \{S_0, \ldots, S_{m-1}\}$, missä $S_i \subseteq R$ kaikilla i. Onko olemassa $S' \subseteq S$, jolle $\bigcup_{X \in S'} X = S$ ja $S_i \cap S_j = \emptyset$, kun $S_i, S_j \in S'$?

Graafin $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, missä $\mathcal{V} = \{0, \ldots, m-1\}$ ja $\mathcal{E} = \{\{i, j\} : S_i \cap S_j = \emptyset\}$, klikit vastaavat ongelman osittaisratkaisuja. Voitaisiin suoraan soveltaa AllCliques-algoritmia ja tarkastaa, onko jokin löydetyistä maksimaalisista klikeistä ratkaisu.

Täsmällinen peitto II

AllCliques-algoritmissa valitaan solmuille järjestys. Valitaan osajoukoille laskeva leksikografinen järjestys. Merkitään H_i:llä niiden osajoukkojen joukkoa, joiden pienin alkio on i. H_i:n joukot edeltävät H_{i+1}:n joukkoja.

AllCliques-algoritmissa C_l muodostuu solmuista, jotka ovat järjestyksessä listassa jo olevien jälkeen ja kaikkien listassa olevien solmujen naapureita. (tässä: eivät sisällä yhteisiä alkioita ko. osajoukkojen kanssa)

Parannus: jos r on pienin alkio, jota listassa olevat joukot eivät peitä, suppeampi valintajoukko $C'_l = C_l \cap H_r$ riittää – muuten ei alkiota r saada peitettyä!

Rajoitusfunktiot

Optimointitehtävässä hakupuuta voidaan karsia arvioimalla, miten hyviä ratkaisuja jostakin haarasta voi löytyä.

Olkoon profit (X) ratkaisun X hyöty. Olkoon $P(X)$ suurin hyöty, joka voidaan saavuttaa osittaisratkaisun X jälkeläisissä. Olkoon $B(X)$ helposti laskettava funktio, jolla arvioidaan $P(X)$:ää s.e. $B(X) \geq P(X)$.

Jos paras aiemmin löydetty ratkaisu on X', käsitellään osittaisratkaisua X, ja profit $(X') > B(X)$, voidaan haara karsia, sillä profit$(X') > B(X) \geq P(X)$, eikä X:n jälkeläisten joukossa voi olla X':a parempaa ratkaisua.

```
BOUNDING(X):
jos X on käypä ratkaisu:
    P ← profit (X)
        if P > OptP :
            OptP ← P
            OptX ← X
laske C_l
B ← B (X)
kullekin x ∈ C_l:
    if B ≤ OptP: # jos karsitaan myös
        return # yhtäsuuruudella, testin on
            # oltava tässä, sillä
            # OptP voi muuttua
    BOUNDING(X + [x])
```


Rationaaliselkäreppu

Eräs tapa muodostaa rajoitusfunktio on höllentää joitakin alkuperäisen tehtävän rajoituksista siten, että ratkaiseminen helpottuu.

Selkäreppuongelma: Annetaan n esinettä, joiden painot ovat $w_1, \ldots w_n$ ja hyödyt $p_1, \ldots, p_n$. Repun kapasiteetti on M. Maksimoidaan $P(x) = \sum p_i x_i$ rajoitusehdoilla $x_i \in \{0,1\}$ ja $\sum w_i x_i \leq M$.

Rationaaliselkäreppuongelma: kuten edellä, mutta vaaditaan $x_i \in \{0,1\}$:n sijaan, että $0 \leq x_i \leq 1$.

Rationaaliselkäreppu

Annetaan n esinettä, joiden painot ovat $w_1, \ldots w_n$ ja hyödyt $p_1, \ldots, p_n$. Repun kapasiteetti on M. Maksimoidaan $P(x) = \sum p_i x_i$ rajoitusehdoilla $0 \leq x_i \leq 1$ ja $\sum w_i x_i \leq M$.

Kokonaislukuarvoisilla p_i, w_i, M muuttujat x_i tulevat olemaan rationaalisia. Ahne algoritmi antaa optimin:

RKNAP$(p_1, \ldots, p_n, w_1, \ldots, w_n, M)$
järjestä esineet s.e. $p_1/w_1 \geq p_2/w_2 \geq \ldots \geq p_n/w_n$
for $i = 1$ to n:
$\quad x_i \leftarrow \min\left(1, \frac{M - \sum_{j=1}^{i-1} w_j x_j}{w_i}\right)$
return $\sum p_i x_i$

Selkäreppu

```
KNAPSACK3(l, CurW):
if l = n:
    if ∑ p_i x_i > OptP:
        OptP ← ∑ p_i x_i
        OptX ← X
if l = n:
    C_l ← ∅
else if CurW + w_{l+1} ≤ M:
    C_l ← {0, 1}
else:
    C_l ← {0}
B ← ∑_{i=1}^{l} p_i x_i + RKNAP(p_{l+1}, ..., p_n, w_{l+1}, ... w_n, M − CurW)
for x ∈ C_l:
    if B ≤ OptP
        return
    x_{l+1} ← x
    KNAPSACK3(l + 1, CurW + w_{l+1} x_{l+1})
```

Tämä algoritmi olettaa, että $\frac{p_1}{w_1} \geq \ldots \geq \frac{p_n}{w_n}$. Kirjan testeissä tietynlaisilla satunnaisilla ongelmilla tämä karsinta pienensi hakupuuta dramaattisesti (parhaimmillaan muttei epätyypillisesti 18953093→180).

Kauppamatkustajan ongelma

Annetaan $K_n = (\mathcal{V}, \mathcal{E})$, n solmun täydellinen graafi, ja kustannusfunktio $\mathrm{cost} : \mathcal{E} \mapsto \mathbb{Z}^+$. Etsi Hamiltonin kierros X, jolle $\mathrm{cost}(X) = \sum_{e \in E(X)} \mathrm{cost}(e)$ on pienin. (Hamiltonin kierros on polku, joka käy kaikissa solmuissa ja palaa lähtöpisteeseensä.)
Hamiltonin kierros voidaan esittää solmujen permutaationa, ja kierros voidaan valita alkamaan solmusta 0.
Kierros 2 5 1 0 3 4 6 2 voidaan siten esittää listana $[0, 3, 4, 6, 2, 5, 1]$ tai $[0, 1, 5, 2, 6, 4, 3]$.

```
TSP1(l):
if l = n:
    C ← cost(X)
    if C < OptC:
        OptC ← C
        OptX ← X

if l = 0:
    C_l ← {0}
else if l = 1:
    C_l ← {1, ... n − 1}
else
    C_l ← C_{l−1} \ {x_{l−1}}

for each x ∈ C_l:
    x_l ← x
    TSP1(l + 1)
```


Rajoitusfunktioita kauppamatkustajalle

cost-funktio voidaan esittää matriisina M, jossa m_{ij} on kaaren (i, j)(suunnattu!) kustannus.

$$M = \begin{bmatrix} \infty & 3 & 5 & 8 \\ 3 & \infty & 2 & 7 \\ 5 & 2 & \infty & 6 \\ 8 & 7 & 6 & \infty \end{bmatrix}$$

MINEDGEBOUND: Lasketaan kunkin sarakkeen (rivin) pienimmät arvot yhteen; pitäähän jokaiseen solmuun tulla jostakin (jokaisesta mennä jonnekin).

REDUCEBOUND: Jos jonkin rivin (sarakkeen) kaikista alkioista vähennetään k, kierroksen pituus pienenee k:lla. Siis: olkoon c sarakkeiden pienimpien alkioiden summa. Vähennetään kunkin sarakkeen kaikista alkioista pienin alkio. Lasketaan saadusta matriisista vastaavasti riveittäin r. Näin joka riville ja sarakkeeseen tulee ainakin yksi 0. Alaraja: $c + r$

Rajoituksia kauppamatkustajalle II

Osittaisratkaisua $X = [x_0, \ldots, x_{l-1}]$ vastaava alaraja saadaan seuraavasti: käsitellään X:ä kuin se olisi yksi solmu, josta solmuun y siirtyminen aiheuttaa kustannuksen $\mathrm{cost}((x_{l-1}, y))$ ja johon siirtyminen solmusta aiheuttaa kustannuksen $\mathrm{cost}((y, x_0))$.

Korvataan alkuperäisen matriisin rivi x_0 rivillä x_{l-1} ja asetetaan $m_{x_0 x_0} = \infty$. Poistetaan kustannusmatriisista rivit ja sarakkeet $x_1, \ldots, x_{l-1}$.

$$M = \begin{bmatrix} \infty & 3 & 5 & 8 \\ 3 & \infty & 2 & 7 \\ 5 & 2 & \infty & 6 \\ 8 & 7 & 6 & \infty \end{bmatrix}$$

Näin ongelma on saatu redusoitua $n - l + 1$ solmun suunnatuksi kauppamatkustajan ongelmaksi, johon voidaan soveltaa edellisiä alarajoja.

Rajoituksia maksimiklikkiongelmalle

ALLCLIQUES-proseduurissa C_l oli niiden solmujen joukko, jotka ovat osittaisratkaisun yhteisiä naapureita, ja tulevat järjestyksessä osittaisratkaisun solmujen jälkeen.

Rajoitus: $B(X) = |X| + |C_l|$

Rajoja graafinväritysongelmasta: Jos graafin solmut voidaan värittää k värillä ilman, että yhdenkään kaaren päätepisteet ovat samanvärisiä, sen suurin klikki on kooltaan korkeintaan k (klikin kaikki solmut ovat naapureita ja siten erivärisiä).

Rajoitus: väritetään solmujen C_l indusoima graafi. Jos tämä onnistuu k värillä, $B(X) = |X| + k$.

Graafinväritysongelma on laskennallisesti hankala. Raja voidaan hakea ahneella algoritmilla: väritetään kukin solmu vuorollaan mahdollisimman pieninumeroisella värillä. Tai voidaan värittää graafin solmut aluksi, ja tarkastella kustakin C_l:stä, monenko värisiä solmuja se sisältää.

Haaraudu ja rajoita (branch and bound)

Aiemmin vaihtoehdot $x \in C_l$ on käyty läpi mielivaltaisessa järjestyksessä. Parempi voi olla laskea kullekin x:lle $B(X + [x])$ ja tarkastella ensin lupaavimpia vaihtoehtoja. Näin saatetaan löytää hyviä ratkaisuja karsimaan hakua.

```
BranchAndBound(X):
jos X on käypä ratkaisu:
    P ← profit(X)
        if P > OptP :
            OptP ← P
            OptX ← X

laske C_l

vaihtoehdot ← [ ]
kullekin x ∈ C_l:
    laske B_x ← B(X + [x])
    vaihtoehdot ← vaihtoehdot + [(x, B_x)]
järjestä vaihtoehdot B_x:n laskevaan
järjestykseen

kullekin (x, B_x) ∈ vaihtoehdot:
    if B_x ≤ OptP:
        return
    BranchAndBound(X + [x])
```


Dynaaminen ohjelmointi maksimiklikkiongelmassa

Etsitään graafin $G = (\mathcal{V}, \mathcal{E})$, missä $\mathcal{V} = \{1, \ldots, n\}$. Esilasketaan $N_v = \{u \in \mathcal{V} : \{u, v\} \in E\}$ ja $G_v = \{u \in \mathcal{V} : u \geq v\}$. Merkitään c_v:llä suurimman G_v:hen sisältyvän klikin S kokoa. Nyt $c_{i-1} \in \{c_i, c_i + 1\}$ ja $c_{i-1} = c_i + 1$ vain silloin, kun G_{i-1} sisältää $c_i + 1$ solmun klikin (johon väistämättä kuuluu solmu $i - 1$).

```
for i = n downto 1:
    found ← false
    MaxClique([i], G_i ∩ N_i)
    c_i ← OptP
MaxClique(X, N):
if |X| > OptP :
    OptP ← |X|
    OptX ← X
    found ← true
    return
if |X| + |N| ≤ OptP:
    return
for x ∈ N:
    if |X| + c_x ≤ OptP:
        return
    MaxClique(X + [x], N ∩ N_x)
    if found:
        return
```


Heuristiset menetelmät

heuristic: involving or serving as an aid to — problem-solving by experimental and especially trial-and-error methods; also : of or relating to exploratory problem-solving techniques that utilize self-educating techniques (as the evaluation of feedback) to improve performance

- kun ratkaisuavaruus on liian suuri peräytyvälle haulle
- etsivät hyviä ratkaisuja yrityksen ja erehdyksen kautta tekemällä muutoksia aiempiin ratkaisuihin
- sopivat optimointigelmiin (jos hyvä ratkaisu riittää) ja hakuongelmiin, mutta eivät yleensä enumerointi- tai generointiongelmiin

Optimointiongelma

$$
\begin{gathered}
\max P(x) \\
g_j(x) \leq 0, j = 1 \ldots m \\
x \in X \\
X \text{ äärellinen}
\end{gathered}
$$

$P(x)$ on kohdefunktio, ja $g_j(x) \leq 0$:t ovat rajoitusehtoja. Mikä tahansa $x \in X$ on ratkaisu. Jos lisäksi $g_j(x) \leq 0$, x on käypä ratkaisu. Jos $P(x) \geq P(x')$ kaikilla $x' \in X$, $g_j(x') \leq 0$, ratkaisu x on optimaalinen.

Sakkofunktiomenetelmällä saadaan epäkäyvistä ratkaisuista käypiä:

$$
\begin{gathered}
\max P(x) - \mu \sum_j \Phi(g_j(x)) \\
x \in X \\
X \text{ äärellinen,}
\end{gathered}
$$

missä $\Phi(y) = 0$, kun $y \leq 0$, ja $\Phi(y) > 0$, kun $y > 0$. Voidaan valita riittävän suuri μ:n arvo heti aluksi tai kasvattaa μ:tä vähitellen.

Yleinen heuristinen menetelmä

Heuristisissa menetelmissä keskeinen käsite on naapuristo. Ratkaisun x naapuristo on $N(x) \subseteq X$. Heuristiikka $h_N(x)$ palauttaa jonkin käyvän ratkaisun x:n naapuristosta tai *Fail*.

GENERICHEURISTICSEARCH:
$x \leftarrow$ jokin käypä $x \in X$
BestX $\leftarrow x$
while not *lopetusehto*:
    $y \leftarrow h_N(x)$
    if $y \neq$ *Fail*
        $x \leftarrow y$
        if $P(x) > P(\mathit{BestX})$
            BestX $\leftarrow X$
return *BestX*

Mahdollisia heuristiikkoja

1. etsi käypä $y \in N(x) \setminus \{x\}$, jolle $P(y)$ on suurin; palauta y tai *Fail*, jos ainoatakaan käypää ratkaisua ei ole
2. etsi käypä $y \in N(x)$, jolle $P(y)$ on suurin; jos $P(y) > P(x)$, palauta y, muuten *Fail*
3. etsi jokin käypä $y \in N(x)$
4. etsi jokin käypä $y \in N(x)$; jos $P(y) > P(x)$, palauta y, muuten *Fail*

Tasainen graafin ositus

Annettu: $2n$ solmun täydellinen graafi $G = (\mathcal{V}, \mathcal{E})$ ja kustannusfunktio $\mathrm{cost} : \mathcal{E} \mapsto \mathbb{Z}^+ \cup \{0\}$.

$$\min C\{\mathcal{V}_0, \mathcal{V}_1\} = \sum_{v_0 \in \mathcal{V}_0, v_1 \in \mathcal{V}_1} \mathrm{cost}(\{v_0, v_1\})$$

$$\mathcal{V} = \mathcal{V}_0 \cup \mathcal{V}_1, \ |\mathcal{V}_0| = |\mathcal{V}_1| = n$$

Esimerkkiratkaisu: Ratkaisuavaruus X olkoon $\mathcal{V}$:n ositusten $[\mathcal{V}_0, \mathcal{V}_1]$, joille $|\mathcal{V}_0| = |\mathcal{V}_1| = n$. Osituksen x naapuristoksi $N(x)$ valitaan niiden ositusten joukko, jotka saadaan x:stä siirtämällä kummastakin joukosta yksi alkio toiseen. Heuristiikka $h_N(x)$ voisi olla steepest ascent: etsi paras naapuri y; jos kohdefunktio paranee, palauta y, muuten *Fail*.

Naapuristoheuristiikat

Maksimoitaessa kullakin iteraatiolla

Steepest ascent: Valitaan x:n käypä naapuri, jolla kohdefunktion arvo on suurin, kunnes ei enää löydy naapuria, johon siirryttäessä kohdefunktio kasvaisi

Hill-climbing: Valitaan jokin x:n käypä naapuri, jolla kohdefunktion arvo kasvaa, kunnes sellaista ei enää löydy

Molemmat näistä pysähtyvät ensimmäiseen paikalliseen optimiin. Paikallinen optimi on ratkaisu x, jolle pätee, että $P(x) > P(y)$ kaikilla käyvillä $y \in N(x)$.

Lokaaleja optimeja voidaan jossakin määrin vähentää laajentamalla naapuristoa, mutta se harvoin poistaa ongelman; lisäksi naapuriston kasvaessa $h_N(x)$:n laskenta tulee työläämmäksi.

Naapuristoheuristiikat II

Great deluge: Valitaan joka iteraatiolla käypä naapuri $y \in N(x)$, jolle $P(y) \geq W$, missä W on vedenpinnan taso. Kasvatetaan W:tä aika ajoin, kunnes käypiä naapureita ei ole.

Record-to-record travel: Valitaan joka iteraatiolla käypä naapuri $y \in N(x)$, jolle $P(y) \geq P(BestX) - D$, missä D on vakio.

Simuloitu jäähdytys

Simuloitu jäähdytys perustuu analogiaan metallin jäähtymisestä.

$h_N(x)$: Valitaan satunnainen $y \in N(x)$. Olkoon $\Delta P = P(y) - P(x)$. Jos $\Delta P \geq 0$, palautetaan y. Jos $\Delta P < 0$, palautetaan y todennäköisyydellä $e^{\Delta P/T}$, missä T on systeemin senhetkinen lämpötila; muuten *Fail*.

Aluksi T on suhteellisen suuri, joten ratkaisua huonontavia siirtoja hyväksytään usein. Haun mittaan lämpötilaa lasketaan, jolloin huonontavia siirtoja hyväksytään yhä harvemmin.

Yksinkertaisimmillaan voidaan joka iteraatiolla asettaa $T \leftarrow \alpha T$, missä $\alpha < 1$, mutta $a \approx 1$.

Tabu-haku

Heuristiikka, jossa valitaan x:n käypä naapuri, jolla kohdefunktion arvo on suurin, siirtyy kyllä pois paikallisesta optimista, mutta palaa usein heti takaisin. Siksi tabu-haku:

$h_N(x)$: Valitaan x:n käypä naapuri, jolla kohdefunktion arvo on suurin, kuitenkin perumatta mitään viimeisen l iteraation aikana tehtyä muutosta.

change(x, y) kuvaa muutosta, joka tehdään siirryttäessä x:stä sen naapuriin y:hyn.

TabuSearch
TabuList $\leftarrow []$
valitse käypä $x \in X$
while not *lopetusehto*:
  $T = \{y : y \in N(x), \text{change}(x, y) \in \textit{TabuList}\}$
  $N \leftarrow N(x) \setminus T$
  etsi käypä $y \in N$ jolle $P(y)$ on suurin
  $x \leftarrow y$
  lisää *TabuList*:iin change(y, x)
  poista *TabuList*:stä kaikki paitsi l uusinta alkiota
  if $P(X) > \textit{BestP}$
    $\textit{BestP} \leftarrow P(X)$
    $\textit{BestX} \leftarrow X$

Naapuriston valinta

Naapuriston ja ratkaisuavaruuden yhdistelmä voidaan tulkita (suunnattuna) verkkona, jonka solmut ovat ratkaisuja. Solmusta x on kaari solmuun y, jos $y \in N(x)$.

Hyvän naapuriston ominaisuuksia:

1. Jokainen ratkaisu—tai ainakin optimiratkaisu—on saavutettavissa mistä tahansa ratkaisusta.
2. Naapuristo on suhteellisen pieni, ja ratkaisun ja sen naapurien kohdefunktioiden arvot korreloivat edes jossakin määrin.
3. Naapuristo on riittävän suuri, jotta jokainen ratkaisu—tai ainakin optimiratkaisu—on saavutettavissa mistä tahansa ratkaisusta pienehköllä määrällä siirtoja.

Kohdefunktion sileys

On huomattavaksi eduksi, jos kohdefunktion arvo on suuri ratkaisuilla, jotka ovat lähellä optimiratkaisua. Olkoon $x_i \in \{0,1\}$ ja $N(x) = \{y \in \{0,1\}^n : \mathrm{dist}(x,y) = 1\}$.

Vrt.
1. max $\sum_i x_i$
2. max $\prod_i x_i$
3. max $\sum_i x_i - 3\left(\sum_i x_i \mod 4\right)$
4. max $2n \prod_i x_i - \sum_i x_i$

Laajat ”laaksot” tai ”tasangot” kohdefunktiossa ovat usein hankalia. Valittu ongelman esitystapa ja naapuristo ovat avainasemassa.

Geneettiset algoritmit

Sen sijaan, että ylläpidettäisiin yhtä senhetkistä ratkaisua, voidaan ylläpitää kokonaista populaatiota. Uusia ratkaisuja saadaan risteyttämällä vanhoja ja mutatoimalla niitä.

Risteytyksessä kahdesta ratkaisusta saadaan kaksi uutta ratkaisua. Mutaatiossa ratkaisu x korvataan jollakin naapurillaan; $x \leftarrow h_N(x)$.

GENETICALGORITHM
valitse alkupopulaatio P
while not *lopetusehto*:
    $Q \leftarrow P$
    laske risteytettävien parien lista R
    for $(w,x) \in R$
        $(y,z) = \text{risteytys}(w,x)$
        $y \leftarrow h_N(y)$
        $z \leftarrow h_N(z)$
        $Q \leftarrow Q \cup \{y,z\}$
        $P \leftarrow Q$:n *popsize* parasta yksilöä
    $b \leftarrow Q$:n paras yksilö
    if $P(b) > P(BestX)$
        $BestX \leftarrow b$

Risteytys ja parinmuodostus

Listat $A = [a_1, \ldots, a_n]$ ja $B = [b_1, \ldots, b_n]$ voidaan risteyttää esim. seuraavasti:

single-point crossover Valitaan $1 \le j < n$. Jälkeläiset ovat
$C = [a_1, \ldots a_j, b_{j+1}, \ldots b_n]$ ja
$D = [b_1, \ldots b_j, a_{j+1}, \ldots a_n]$.

two-point crossover Valitaan $1 \le j < k \le n$. Jälkeläiset ovat
$C = [a_1, \ldots, a_j, b_{j+1}, \ldots, b_k, a_{k+1}, \ldots, a_n]$ ja
$D = [b_1, \ldots, b_j, a_{j+1}, \ldots, a_k, b_{k+1}, \ldots, b_n]$.

uniform crossover Valitaan $S \subseteq \{1, \ldots, n\}$. Jälkeläisillä $c_i = a_i$ ja
$d_i = b_i$, jos $i \in S$; muuten $c_i = b_i$ ja $d_i = a_i$.

Permutaatioiden α ja β risteyttämiseksi voidaan valita $1 \le j < k \le n$, ja

PartiallyMatchedCrossover(n, α, β, j, k)
$\gamma \leftarrow a$
$\delta \leftarrow \beta$
for $i = j$ to k:
    $\gamma \leftarrow (a_i\ \beta_i)\,\gamma$
    $\delta \leftarrow (a_i\ \beta_i)\,\delta$

Risteytyksen tulokset eivät aina ole käypiä ratkaisuja. Voidaan
1) soveltaa sakkofunktiomenetelmää
2) räätälöidä ongelmalle risteytysfunktio, jolla jälkeläiset ovat aina käypiä.

Parinmuodostus: populaation ratkaisut voidaan parittaa eri kriteerein, esim. paremmuusjärjestyksessä. Voidaan myös generoida hyvistä ratkaisuista enemmän jälkeläisiä.

Steinerin kolmikkojärjestelmät

Steinerin kolmikkojärjestelmä on joukkojärjestelmä $(\mathcal{V}, \mathcal{B})$, missä $\mathcal{B}$ koostuu $\mathcal{V}$:n 3-osajoukoista, ja kukin $\mathcal{V}$:n 2-osajoukko on täsmälleen yhden $b \in B$:n osajoukko. STS on täydellisen graafin kaarien ositus kolmioiksi. Esim.

$$\begin{aligned} \mathcal{V} &= \{1, \ldots 7\} \\ \mathcal{B} &= \left\{ \begin{array}{lll} \{1,2,4\}, & \{2,3,5\}, & \{3,4,6\}, \\ \{4,5,7\}, & \{1,5,6\}, & \{2,6,7\}, \\ \{1,3,7\} & & \end{array} \right\} \end{aligned}$$

Hill-climbing ja Steinerin kolmikkojärjestelmät

Piste $v \in V$ on *elävä*, jos siitä lähtee kaaria, jotka eivät vielä ole missään kolmiossa $b \in B$. Kaari $\{u, v\}$ on elävä, jos se ei esiinny missään kolmiossa $b \in B$.

StinsonsAlgorithm(v):
$\mathcal{V} \leftarrow \{1, \ldots, n\}$
$\mathcal{B} \leftarrow \emptyset$
while $|\mathcal{B}| < v\,(v-1)\,/6$:
    valitse elävä piste x
    valitse elävät kaaret $\{x, y\}$ ja $\{x, z\}$
    if $\{y, z\}$ on elävä:
        $\mathcal{B} \leftarrow B \cup \{\{x, y, z\}\}$
    else
        etsi $\mathcal{B}$:stä blokki $\{w, y, z\}$
        $\mathcal{B} \leftarrow B \cup \{\{x, y, z\}\} \setminus \{\{w, y, z\}\}$

Selkäreppu ja simuloitu jäähdytys

Selkäreppuongelma: Annetaan n esinettä, joiden painot ovat $w_1, \ldots w_n$ ja hyödyt $p_1, \ldots, p_n$. Repun kapasiteetti on M. Maksimoidaan $P(x) = \sum p_i x_i$ rajoitusehdoilla $x_i \in \{0,1\}$ ja $\sum w_i x_i \leq M$.

Valitaan naapuristo:
$N(x) = \{y \in \{0,1\}^n : \mathrm{dist}(x,y) = 1\}$.
Satunnainen naapuri y saadaan kääntämällä satunnainen x_j.

Jos $x_j = 0$, kohdefunktion muutos $\Delta P = +p_j$, ja uusi naapuri hyväksytään, jos se on käypä. Jos $x_j = 1$, $\Delta P = -p_j$, ja uusi naapuri hyväksytään todennäköisyydellä $e^{p_j/T}$.

Valitaan aluksi T siten, että suuri osa huonontavistakin siirroista hyväksytään; esim. $4 \max_i p_i$, ja asetetaan joka iteraation jälkeen $T \leftarrow \alpha T$. Kirjassa parhaat tulokset saatiin $\alpha = 0.9999$:llä.

Selkäreppu ja tabu-haku

Valitaan naapuristo:
$N(x) = \{y \in \{0,1\}^n : \mathrm{dist}(x,y) = 1\}$.

Ei maksimoidakaan hyötyfunktiota(?!!), vaan

1. lisätään reppuun esine i, joka ei ole tabu-listalla ja jolla on suurin p_i/w_i-suhde niistä esineistä, joilla $x_i = 0$ ja jotka mahtuvat reppuun

2. ellei sellaista ole, poistetaan repusta esine i, joka ei ole tabu-listalla ja jolla on pienin p_i/w_i-suhde niistä esineistä, joilla $x_i = 1$.

3. Lisätään tabulistaan i.

Graafin väritys

Mikä on pienin määrä värejä, joilla graafi $G = (\mathcal{V}, \mathcal{E})$ voidaan värittää siten, että u ja v ovat erivärisiä, jos $\{u,v\} \in \mathcal{E}$?

Ositetaan solmut väriluokkiin $\mathcal{V}_1 \ldots \mathcal{V}_k$ esim. ahneella algoritmilla. Kun on löydetty k-väritys, etsitään heuristisella haulla $k-1$-väritystä seuraavasti.

Otetaan kohdefunktioksi $\max \sum_i |\mathcal{V}_i|^2$. Tämä ohjaa hakua sellaiseen suuntaan, että joillakin väreillä väritetään paljon solmuja (ja toisilla taas vähän). Jos jonkun osan koko menee nollaan, on löytynyt $k-1$-väritys, ja voidaan alkaa etsiä $k-2$-väritystä jne.

Schurin luvut

Schurin luku $s(m)$ on suurin kokonaisluku s, jolla kokonaisluvut $X = \{1, \ldots, s\}$ voidaan osittaa m osaan siten, että missään joukossa ei ole kahta (ei välttämättä eri) alkiota ja niiden summaa.

Esim. $s(3) = 13$: $\left\{ \begin{array}{l} \{1,4,7,10,13\} \\ \{2,3,11,12\} \\ \{5,6,8,9\} \end{array} \right\}$

Kun etsitään lukujen $1 \ldots s$ summatonta ositusta, ilmeisin valinta kohdefunktioksi olisi eri osissa esiintyvien summien lukumäärä, mutta parempi on maksimoida $\max c_1 f_1 + c_2 f_2$, missä $c_1 \gg c_2$ ja
$f_1 = \max t$, jolla missään osassa ei ole summaa t
$f_2 = \sum_{i=1}^{m} \sum_{t,u \in X_i} g(t,u)$

missä $g(t,u) = \begin{cases} 0 & \text{jos } t+u \leq s \\ 2s - t - u & \text{jos } t+u > s \end{cases}$

Tässä f_2:n tarkoitus on vain ohjata hakua lupaavaan suuntaan.

Kauppamatkustajan ongelma

Ratkotaan kauppamatkustajan ongelmaa geneettisillä algoritmeilla.

n-opt-naapuristo: poistetaan syklistä n kaarta ja lisätään siihen n kaarta niin, että saadaan taas sykli.

Voidaan määritellä risteytysfunktio seuraavasti: Risteytetään kaksi ratkaisua jollakin permutaatioita yhdistelevällä risteytysfunktiolla, ja sovelletaan tämän jälkeen saatuun tulokseen steepest descent -menetelmää esim. 2-opt-naapuristolla.

Voidaan myös yksinkertaisesti risteyttää permutaatiot ja lisätä steepest descent kohdefunktioon. Tällöin permutaation hyvyyttä arvioitaisiin soveltamalla siihen ensin steepest descent -hakua ja laskemalla kohdefunktion arvo tämän jälkeen.

Isomorfismi

Nimettyjä rakenteita ei useinkaan pidetä oleellisesti eri rakenteina, jos ne eroavat toisistaan vain nimeämisen osalta.

Isomorfismi on bijektiivinen kuvaus, joka kuvaa yhden rakenteen osat toisen rakenteen osille niin, että itse rakenne säilyy. Rakenteet ovat isomorfiset, jos niiden välillä on isomorfismi.

Esimerkki: Kaksi graafia $G_1 = (V_1, E_1)$ ja $G_2 = (V_2, E_2)$ ovat isomorfiset, jos on olemassa bijektio $f : V_1 \mapsto V_2$ s.e. $\{u, v\} \in E_1$ jos ja vain jos $\{f(u), f(v)\} \in E_2$.

Automorfismi

Automorfismi on isomorfismi rakenteelta itselleen. Automorfismit ovat tietyssä mielessä rakenteen symmetrioita.

Esimerkki: Graafin $G = (V, E)$ automorfismi on bijektio (permutaatio) $\pi : V \mapsto V$, jolla $\{u, v\} \in E$ jos ja vain jos $\{\pi(u), \pi(v)\} \in E$.

Minkä tahansa graafin automorfismit muodostavat *ryhmän*.

Ryhmä

Joukko-operaatiopari $(G, *)$ on ryhmä, jos

1. Binäärinen operaatio $*$ on suljettu: $g_1 * g_2 \in G$ kaikilla $g_1, g_2 \in G$, ts. $* : G \times G \mapsto G$
2. Ryhmässä on yksikköalkio $\mathbf{I}$, jolla $g * \mathbf{I} = g = \mathbf{I} * g$ kaikilla $g \in G$
3. Jokaisella $g \in G$ on käänteisalkio $g^{-1} \in G$, jolle $g^{-1} * g = \mathbf{I} = g * g^{-1}$
4. Binäärioperaatio $*$ on liitännäinen: $(g_1 * g_2) * g_3 = g_1 * (g_2 * g_3)$ kaikilla $g_1, g_2, g_3 \in G$

Esimerkkejä:
kokonaisluvut modulo n ja yhteenlasku
$m \times m$-matriisit, joiden determinantti $\neq 0$
kolmiulotteisen kappaleen rotaatiot

Kun operaatio on ilmeinen, puhutaan ryhmästä G.

Kertotaulu

Äärellinen ryhmä voidaan esittää tauluna:

$*$	$\mathbf{I}$	a	b	c	d	e	f	g
$\mathbf{I}$	$\mathbf{I}$	a	b	c	d	e	f	g
a	a	b	c	$\mathbf{I}$	e	f	g	d
b	b	c	$\mathbf{I}$	a	f	g	d	e
c	c	$\mathbf{I}$	a	b	g	d	e	f
d	d	g	f	e	$\mathbf{I}$	c	b	a
e	e	d	g	f	a	$\mathbf{I}$	c	b
f	f	e	d	g	b	a	$\mathbf{I}$	c
g	g	f	e	d	c	b	a	$\mathbf{I}$

Kun 1. riviin ja sarakkeeseen laitetaan $\mathbf{I}$, kertotaulu on redusoitu latinalainen neliö, jossa pätee liitännäisyys $M\left[M\left[g_i, g_j\right], g_k\right] = M\left[g_i, M\left[g_j, g_k\right]\right]$.

Ryhmän aliryhmä:
H on G:n aliryhmä, jos H on ryhmä ja $H \subseteq G$.
Esim. edellisessä taulussa $\{\mathbf{I}, a, b, c\}$.

Aliryhmistä

Äärellisen ryhmän kertaluku $|G|$ on joukon alkioiden lkm.

Jos H on ryhmän G epätyhjä osajoukko, H on G:n aliryhmä, joss H on suljettu (G:n operaatiolla):

Tod. Jos $H = \{\mathbf{I}\}$, tilanne on selvä. Oletetaan, että $h_1 h_2 \in H$ kaikilla $h_1, h_2 \in H$. Valitaan jokin $h \in H$. Kaikilla $n \in \mathbb{Z}^+$ pätee:
$h^n = hh \ldots h \in H$. Koska H on äärellinen, täytyy olla jotkin $m < n$ s.e. $h^m = h^n$. Nyt $h^n h^{n-m} = h^m h^{n-m} = h^n$, ja $h^{n-m} = \mathbf{I} \in H$ sekä $h^{-1} = h^{n-m-1} \in H$.

Sivuluokat

Kun $g \in G$ ja $H \subseteq G$, merkitään $gH = \{gh : h \in H\}$. Kun $A, B \subseteq G$, merkitään $AB = \{ab : a \in A, b \in B\}$.

Olkoon H äärellisen ryhmän G aliryhmä. gH on vasen sivuluokka (left coset), johon $g \in G$ kuuluu.

Lagrange: jos H on G:n aliryhmä, G voidaan esittää pistevieraiden vasempien sivuluokkien unionina:

$$G = g_1 H \cup g_2 H \cup \ldots \cup g_n H,$$

missä $g_i \in G$, ja $g_i H \cap g_j H = \emptyset$, kun $i \neq j$.

Tod. $|gH| = |H|$ kaikilla $g \in G$, sillä $f(x) = gx$ on bijektio $H \mapsto gH$. Jos $g_1 H \cap g_2 H \neq \emptyset$, missä $g_1, g_2 \in G$, täytyy olla olemassa $h_1, h_2 \in H$, joilla $g_1 h_1 = g_2 h_2$ ja $g_1 = g_2 h_2 h_1^{-1}$. Nyt mille tahansa $h \in H$ saadaan $g_1 h = g_2 \left(h_2 h_1^{-1} h\right) \in g_2 H$, ja $g_1 H \subseteq g_2 H$. Koska $|g_1 H| = |g_2 H| = |H|$, $g_1 H = g_2 H$. Lisäksi jokainen $g \in G$ kuuluu sivuluokkaan gH; sivuluokat siis osittavat G:n, ja $|G|$ on jaollinen $|H|$:lla.

Transversaalit

Kun G esitetään pistevieraiden vasempien sivuluokkien unionina

$$G = g_1 H \cup g_2 H \cup \ldots \cup g_n H,$$

joukko $T = \{g_1, \ldots, g_n\}$ on H:n vasen transversaali. Se voidaan muodostaa valitsemalla $n = \frac{|G|}{|H|}$ sivuluokan edustajaa $g_i \in G$ siten, että mikään valittu g_i ei kuulu mihinkään aiempaan sivuluokkaan.

Transversaalit: esimerkki

$*$	$\mathbf{I}$	a	b	c	d	e	f	g
$\mathbf{I}$	$\mathbf{I}$	a	b	c	d	e	f	g
a	a	b	c	$\mathbf{I}$	e	f	g	d
b	b	c	$\mathbf{I}$	a	f	g	d	e
c	c	$\mathbf{I}$	a	b	g	d	e	f
d	d	g	f	e	$\mathbf{I}$	c	b	a
e	e	d	g	f	a	$\mathbf{I}$	c	b
f	f	e	d	g	b	a	$\mathbf{I}$	c
g	g	f	e	d	c	b	a	$\mathbf{I}$

Aliryhmällä $H = \{\mathbf{I}, a, b, c\}$ on sivuluokat $\{\mathbf{I}, a, b, c\}$ ja $\{d, e, f, g\}$.
Transversaali saadaan valitsemalla kustakin sivuluokasta yksi alkio; esim. $T = \{\mathbf{I}, d\}$ tai $T = \{b, f\}$.
$TH = \mathbf{I}\{\mathbf{I}, a, b, c\} \cup d\{\mathbf{I}, a, b, c\} = \{\mathbf{I}, a, b, c\} \cup \{d, e, f, g\} = G$.

Permutaatioryhmät

Tarkastellaan jonkin perusjoukon $\mathcal{X}$ permutaatioita.

Permutaatiot (bijektiot $\pi : \mathcal{X} \mapsto \mathcal{X}$) muodostavat ryhmän, kun operaatioksi valitaan funktioiden yhdistely $(\pi_1\pi_2)(x) = (\pi_1 \circ \pi_2)(x) = \pi_1(\pi_2(x))$, sillä

1. Kahden permutaation yhdiste on permutaatio
2. on olemassa identiteettialkio ($\mathbf{I}(x) = x$)
3. permutaatioilla on käänteispermutaatio
4. funktioiden yhdistely on liitännäistä

Olkoon $\mathcal{X}$ epätyhjä joukko ja $\mathrm{Sym}(\mathcal{X})$ sen permutaatioiden joukko. $\mathrm{Sym}(\mathcal{X})$ operaationa funktioiden yhdistäminen on $\mathcal{X}$:n symmetrinen ryhmä. Jos $|\mathcal{X}| = n$, niin $\mathrm{Sym}\{X\}$:ssä on $n!$ alkiota.

Permutaatioryhmät ovat symmetrisen ryhmän aliryhmiä. Esim. Kun $\mathcal{X} = \{0, 1, 2, 3, 4\}$, permutaatiot $\{\mathbf{I}, (0, 1, 2)(3, 4), (0, 2, 1), (3, 4), (0, 1, 2), (0, 2, 1)(3, 4)\}$ muodostavat permutaatioryhmän.

Graafin automorfismit

Merkitään $\alpha(\{u, v\}) = \{\alpha(u), \alpha(v)\}$

Graafin $G = (\mathcal{V}, \mathcal{E})$ automorfismi α on sellainen $\mathcal{V}$:n permutaatio, että $\alpha(\{u, v\}) \in \mathcal{E}$ kaikille graafin kaarille $\{u, v\} \in \mathcal{E}$.

Automorfismit muodostavat ryhmän $\mathrm{Aut}(G)$:

$\mathrm{Aut}(G)$ on epätyhjä, sillä selvästi $\mathbf{I} \in \mathrm{Aut}(G)$

Jos $\alpha, \beta \in \mathrm{Aut}(G)$, niin $\alpha\beta \in \mathrm{Aut}(G)$: oletetaan, että $\{u, v\} \in \mathcal{E}$. $\beta(\{u, v\}) \in \mathcal{E}$, ja $(\alpha\beta)(\{u, v\}) = \alpha(\beta(\{u, v\})) \in \mathcal{E}$, joten $\mathrm{Aut}(G)$ on suljettu.

$\mathrm{Aut}(G)$ on $\mathrm{Sym}(\mathcal{V})$:n epätyhjä osajoukko, joka on suljettu samalla operaatiolla, joten $\mathrm{Aut}(G)$ on $\mathrm{Sym}(\mathcal{V})$:n aliryhmä ja siis ryhmä.

Generaattorit

Alkiot $\alpha_1, \ldots, \alpha_r$ generoivat ryhmän G, jos kaikki alkiot $g \in G$ voidaan esittää alkioiden $\alpha_1, \ldots, \alpha_r$ äärellisenä tulona

$$g = \alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_m},$$

missä $1 \le i_j \le r$ kaikilla j. Alkiot $\alpha_1, \ldots, \alpha_r$ ovat ryhmän G generaattorit, ja merkitään $G = \langle \alpha_1, \ldots, \alpha_r \rangle$.

Esimerkki: Rubikin kuutio

Ideal Toy Company stated on the package of the original Rubik cube that there were more than three billion possible states the cube could attain. It's analogous to MacDonald's proudly announcing that they've sold more than 120 hamburgers.
(J. A. Paulos, Innumeracy)

```
                +--------------+
                |  1    2    3 |
                |  4  top    5 |
                |  6    7    8 |
 +--------------+--------------+--------------+--------------+
 |  9   10   11 | 17   18   19 | 25   26   27 | 33   34   35 |
 | 12  left  13 | 20 front  21 | 28 right  29 | 36  rear  37 |
 | 14   15   16 | 22   23   24 | 30   31   32 | 38   39   40 |
 +--------------+--------------+--------------+--------------+
                | 41   42   43 |
                | 44 bottom 45 |
                | 46   47   48 |
                +--------------+
gap> cube := Group(
( 1, 3, 8, 6)( 2, 5, 7, 4)( 9,33,25,17)(10,34,26,18)(11,35,27,19),
( 9,11,16,14)(10,13,15,12)( 1,17,41,40)( 4,20,44,37)( 6,22,46,35),
(17,19,24,22)(18,21,23,20)( 6,25,43,16)( 7,28,42,13)( 8,30,41,11),
(25,27,32,30)(26,29,31,28)( 3,38,43,19)( 5,36,45,21)( 8,33,48,24),
(33,35,40,38)(34,37,39,36)( 3, 9,46,32)( 2,12,47,29)( 1,14,48,27),
(41,43,48,46)(42,45,47,44)(14,22,30,38)(15,23,31,39)(16,24,32,40));;
gap> Size( cube );
43252003274489856000
```


Ryhmän vaikutus (action)

Ryhmän G vaikutus joukkoon $\mathcal{X}$ on funktio $\alpha : G \times \mathcal{X} \mapsto \mathcal{X}$, merkitään $\alpha : (g, x) \mapsto gx$, jolle pätee:

1. $\mathbf{I}x = x$ kaikille $x \in \mathcal{X}$
2. $g\,(hx) = (gh)\,x$ kaikilla $g, h \in G$ ja $x \in \mathcal{X}$.

Huomaa, että jos $gx_1 = gx_2$, niin
$g^{-1}\,(gx_1) = \left(g^{-1}g\right)x_1 = \mathbf{I}x_1 = x_1$
$= g^{-1}\,(gx_2) = \left(g^{-1}g\right)x_2 = \mathbf{I}x_2 = x_2$.

Itse asiassa kukin g määrittää joukon $\mathcal{X}$ permutaation.

Ryhmän vaikutus. Esimerkki: graafi

Symmetrisestä ryhmästä puhuttaessa puhutaan yleensä ryhmästä S_n, jonka rakenne on sama kuin joukon $\{1, \ldots, n\}$ permutaatioiden muodostaman ryhmän rakenne. Kun S_n vaikuttaa johonkin n-alkioiseen joukkoon V aivan kuin joukon V permutaatiot, sanotaan, että S_n vaikuttaa V:hen *luonnollisella tavalla*.

Jos nyt jokin ryhmä vaikuttaa esimerkiksi graafin $G = (V, E)$ solmuihin jollakin tavalla, ryhmä vaikuttaa *indusoidulla tavalla* graafin kaariin. Itse asiassa tämä indusoi myös vaikutuksen graafien joukkoon.

Ryhmän vaikutus. Esimerkki: binäärikoodi

Kahta binäärikoodia (joukkoa samanmittaisia binäärisanoja) voidaan pitää ekvivalentteina, jos toinen saadaan toisesta komplementoimalla kaikkien sanojen tietyissä positioissa olevat bitit ja permutoimalla positioita.

Tätä vastaa ryhmä, joka on seppeltulo (wreath product) $S_2 \wr S_n$, missä S_n:n vaikutus vastaa positioiden permutointia, ja kunkin S_2:n (joita on n) vaikutus vastaa yksittäisen position bittien komplementointia.

(Seppeltuloa ei käsitellä tässä täsmällisemmin.)

Ryhmän vaikutus. Esimerkki: diedriryhmä

Diedriryhmän D_n alkiot vastaavat n-kulmaisen säännöllisen monikulmion symmetrioita. Diedriryhmä voidaan määrittää seuraavasti: $D_n = \langle r, s \rangle$, missä $r^n = s^2 = (rs)^2 = \mathbf{I}$; ryhmällä on kaksi generaattoria, ja nuo annetut rajoitteet määrittävät yksikäsitteisesti ryhmän rakenteen (kun ajatellaan nuo mainitut potenssit pienimmiksi, joilla saadaan identiteetti). Tässä r vastaa $1/n$ kierroksen kiertoa myötäpäivään ja s peilausta jonkin akselin suhteen.

D_n voi vaikuttaa joukkoon $V = \{0, \ldots, n-1\}$ esim. seuraavasti: $rv = (v+1) \bmod n$, $sv = (n-v) \bmod n$.

D_n voi vaikuttaa joukkoon $\mathbf{R}^2$ seuraavasti:

$$rx = \begin{pmatrix} \cos 2\pi/n & -\sin 2\pi/n \\ \sin 2\pi/n & \cos 2\pi/n \end{pmatrix} x,$$

$$sx = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} x$$

Alkion rata (orbit)

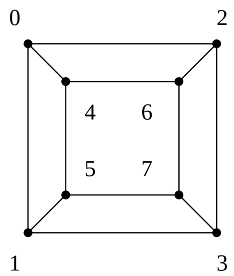
Alkion x rata on $\mathrm{orb}(x) = \{gx : g \in G\} \subset \mathcal{X}$ ja alkion x stabiloija on $G_x = \{g \in G : gx = x\} \subseteq G$. Koska G_x on epätyhjä ($\mathbf{I} \in G_x$) ja suljettu, se on aliryhmä, ja sille voidaan etsiä transversaali. Jos kahdelle transversaalin alkiolle g_i ja g_j pätee $g_i x = g_j x$, niin $g_i^{-1} g_i x = g_i^{-1} g_j x = x$, ja $g_i^{-1} g_j \in G_x$. Nyt $g_i^{-1} g_j G_x = G_x$ ja $g_i G_x = g_i g_i^{-1} g_j G_x = g_j G_x$ – mutta transversaalissa on vain yksi alkio kustakin sivuluokasta, joten $g_i = g_j$. Näin siis $|\mathrm{orb}(x)| = |G| / |G_x|$.

Permutaatioryhmän tietokone-esitykset

Permutaatioryhmän esityksellä olisi hyvä olla seuraavat ominaisuudet:

1. Voidaan tarkistaa, kuuluuko jokin g ryhmään G
2. Voidaan luetella ryhmän alkiot
3. On tilavaatimuksiltaan kohtuullinen

Esim. kuutiograafi:



$\mathrm{Aut}(G) = \langle \alpha, \beta \rangle$, missä
$\alpha = (0, 1, 3, 7, 6, 4)(2, 5)$ ja
$\beta = (0, 1, 3, 2)(4, 5, 7, 6)$.
$|G| = 48$.

Permutaatioryhmän tietokone-esitykset

Voitaisiin tallettaa ryhmän permutaatiot esim. leksikografisessa järjestyksessä.

1. Selviää binäärisellä haulla, kuuluuko g G:hen
2. Alkiot voidaan luetella helposti
3. Tilaa tarvitaan paljon; $\mathrm{Sym}(n)$:llä ryhmässä on $n!$ permutaatiota

Permutaatioryhmän tietokone-esitykset

Voitaisiin tallettaa vain ryhmän generaattorit.

3. Tallennustilaa tarvitaan vähän, mutta
1. & 2. Joudutaan tekemään breadth-first-haku kaikkien alkioiden generoimiseksi, ja saadaan duplikaatteja; esimerkissä $\alpha\alpha\alpha\alpha\alpha\beta\beta\beta = \alpha\beta\alpha\alpha$.

SIMPLEGEN(Γ):
$G \leftarrow \emptyset$
$N \leftarrow \{\mathbf{I}\}$
while $N \neq \emptyset$:
  $G \leftarrow G \cup N$
  $N \leftarrow N\Gamma \setminus G$

missä Γ on ryhmän generaattorien joukko ja $N\Gamma$ on $\{ng : n \in N, g \in \Gamma\}$.

Schreier-Sims

Olkoon G permutaatioryhmä joukossa $\mathcal{X} = \{0, \ldots, n-1\}$.
Merkitään

$$G_0 = \{g \in G : g(0) = 0\}.$$

G_0 on aliryhmä, joka *stabiloi* pisteen 0.
Alkion 0 rata G:ssä on

$$\mathrm{orb}(0) = \{g(0) : g \in G\} = \{x_{0,1}, x_{0,2}, \ldots, x_{0,n_0}\}.$$

Muodostetaan $\mathcal{U}_0$ valitsemalla kutakin 0:n radan alkiota $x_{0,i}$ kohti ryhmän alkio $h_{0,i}$, jolla $h0, i(0) = x_{0,i}$.
$\mathcal{U}_0$ on G_0:n vasen transversaali ($G = U_0 G_0$): Jokainen $g \in G$ kuvaa 0:n jollekin $x_{0,i}$:lle. $g = h_{0,i}\left(h_{0,i}^{-1} g\right)$, ja $h_{0,i}^{-1} g \in G_0$. Siten $g \in h_{0,i} G_0$.
$\mathcal{U}$:ssa ei ole kahta saman sivuluokan edustajaa: kunkin sivuluokan $h_{0,i} G_0$ alkiot kuvaavat 0:n eri $x_{0,i}$:lle.

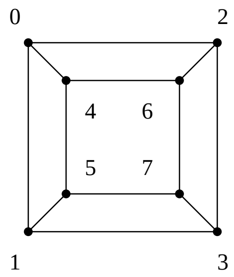
Schreier-Sims

Sovelletaan ideaa rekursiivisesti:

$$\begin{aligned}
G_0 &= \{g \in G : g(0) = 0\} \\
G_1 &= \{g \in G_0 : g(1) = 1\} \\
G_2 &= \{g \in G_1 : g(2) = 2\} \\
&\vdots \\
G_{n-1} &= \{g \in G_{n-2} : g(n-1) = n-1\} = \{\mathbf{I}\}
\end{aligned}$$

Nyt $G \supseteq G_0 \supseteq \ldots \supseteq G_{n-1} = \{\mathbf{I}\}$.
Ryhmän G:n Schreier-Sims-esitys on $G = \mathcal{U}_0 \mathcal{U}_1 \ldots \mathcal{U}_{n-1}$.

Schreier-Sims-esimerkki



Kuutiograafille voidaan valita esim.

$$\mathcal{U}_0 = \left\{ \begin{array}{l} (0)(1)(2)(3)(4)(5)(6)(7), \\ (0,1,3,7,6,4)(2,5), \\ (0,2,6,4)(1,3,7,5), \\ (0,3,6)(1,7,4)(2)(5), \\ (0,4,6,7,3,1)(2,5), \\ (0,5,3,6)(1,7,2,4), \\ (0,6,3)(1,4,7)(2)(5), \\ (0,7)(1,6)(2,5)(3,4) \end{array} \right\}$$

$$\mathcal{U}_1 = \left\{ \begin{array}{l} (0)(1)(2)(3)(4)(5)(6)(7), \\ (0)(1,2)(3)(4)(5,6)(7), \\ (0)(1,4,2)(3,5,6)(7) \end{array} \right\}$$

$$\mathcal{U}_2 = \left\{ \begin{array}{l} (0)(1)(2)(3)(4)(5)(6)(7), \\ (0)(1)(2,4)(3,5)(6)(7) \end{array} \right\}$$

ja $\mathcal{U}_3, \ldots, \mathcal{U}_7 = \{(0)(1)(2)(3)(4)(5)(6)(7)\}$.

Schreier-Sims

Kun tunnetaan G:n Schreier-Sims-esitys $\mathcal{U}_0\mathcal{U}_1 \ldots \mathcal{U}_{n-1}$, kaikki elementit on helppo käydä läpi rekursiivisesti: lasketaan kaikki $g = u_0u_1 \ldots u_{n-1}$, missä $u_i \in \mathcal{U}_i$.

Tarkistaminen, onko $g \in G$ tapahtuu seuraavasti. Yritetään kirjoittaa $g \in G$ voidaan kirjoittaa muodossa $u_0u_1 \ldots u_{n-1}$. Ensin $g(0)$:sta päätellään, mikä $u_0 \in \mathcal{U}_0$ on valittava (se, jolle $u_0(0) = g(0)$). Tämän jälkeen ongelma redusoituu tarkistamiseen, onko $u_0^{-1}g \in G_0$, ts. yritetään kirjoittaa $u_0^{-1}g$ muotoon $u_1 \ldots u_n$, jne.

TEST$(n, g, G = [\mathcal{U}_0, \ldots, \mathcal{U}_{n-1}])$:
for $i \leftarrow 0$ to $n-1$:
  jos löytyy $h \in \mathcal{U}_i$, jolle $h(i) = g(i)$:
    $g \leftarrow h^{-1}g$
  muuten:
    return i
return n

Schreier-Sims-esityksen laskeminen

ENTER$(n, g, G = [\mathcal{U}_0, \mathcal{U}_1, \ldots, \mathcal{U}_{n-1}])$:
$i \leftarrow$TEST$(n, g, G = [\mathcal{U}_0, \mathcal{U}_1, \ldots, \mathcal{U}_{n-1}])$
if $i = n$
  return
$\mathcal{U}_i \leftarrow \mathcal{U}_i \cup \{g\}$
for $j = 0$ to i:
  for $h \in \mathcal{U}_j$:
    ENTER(n, gh, G)

MAIN:
for $i \leftarrow 0$ to $n-1$:
  $\mathcal{U}_i \leftarrow \{\mathbf{I}\}$
for $\alpha \in \Gamma$:
  ENTER$(n, \alpha, G = [\mathcal{U}_0, \ldots, \mathcal{U}_{n-1}])$
return G

Enter-funktio tarkastaa, kuuluuko g Schreier-Sims-muodossa esitetyyn ryhmään G, ja jollei kuulu, lisää generaattoreihin g:n.

Schreier-Sims-kannanvaihdos

Edellä Schreier-Simsissä kiinnitettiin alkiot järjestyksessä $0, \ldots, n-1$. Voidaan toki kiinnittää alkiot mielivaltaisessa järjestyksessä. Valitaan alkioiden $\{0, \ldots, n-1\}$ permutaatio β, ja

$$G_0 = \{g \in G : g(\beta(0)) = \beta(0)\},$$

ja

$$G_i = \{g \in G_{i-1} : g(\beta(i)) = \beta(i)\}.$$

Kaikki operaatiot tehdään aivan vastaavasti kuin aiemmin.

Uutena operaationa on kannanvaihto: muunna kannassa β esitetty ryhmä kantaan β'. Tämä tehdään lisäämällä kannan β permutaatiot enter-proseduurilla kannan β' mukaiseen Schreier-Sims-esitykseen.

Rataedustajien etsintä

Kun tunnetaan $k+1$-osajoukkon ratojen lukumäärä N_{k+1}, voidaan etsiä yksi joukko kultakin radalta. Jos $\mathcal{R}$ on joukko k-osajoukkojen rataedustajia,

$$\mathcal{S} = \{A \cup \{x\} : A \in \mathcal{R}, x \in X \setminus A\}$$

sisältää edustajan kultakin $k+1$-osajoukkojen radalta; joiltakin useampiakin. Samojen ratojen edustajia pitää poistaa, kunnes jäljellä on edustaja kultakin radalta.

Yksinkertainen idea:

kaikille $g \in G$:
  kaikille $A \in \mathcal{S}$ laskevassa leks. järjestyksessä:
    jos $\mathrm{rank}(g(A)) < \mathrm{rank}(A)$:
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{g(A)\} \setminus \{A\}$
      jos $|\mathcal{S}| = N_{k+1}$:
        return

k-permutaation radan minimiedustaja

Olkoon meillä k-permutaatio $t = (t_1, \ldots, t_k)$, jonka alkiot $t_i \in X$. Kun ryhmä G permutoi järjestettyä joukkoa X, se indusoi vaikutuksen k-permutaatioiden joukossa. Etsitään radan $G(t)$ leksikografinen minimiedustaja $\min G(t)$.

Ensinnäkin t_1 tulee kuvata järjestyksessä mahdollisimman aikaisin tulevalle alkiolle. Lasketaan $t'_1 = \min G(t_1)$ esim. soveltamalla G:n generaattoreita ja leveyshakua; löydetään myös g, jolle $t'_1 = g(t_1)$. Lasketaan $t' = g(t)$ ja etsitään edelleen $\min G_{t'_1}(t')$, jne. Tarvittavat stabiloija-aliryhmät voi laskea vaikkapa soveltamalla Schreier-Sims-kannanvaihtoja.

k-osajoukon radan minimiedustaja

Olkoon meillä järjestetyn joukon X k-osajoukko $T = \{t_1, \ldots, t_k\}$. Kun ryhmä G permutoi X:ä, se indusoi vaikutuksen k-osajoukkojen joukossa. Etsitään radan $G(T)$ leksikografinen minimiedustaja $\min G(T)$.

Etsitään kullekin t_i ratansa minimialkio $\min G(t_i)$ ja vastaava ryhmän alkio g_i. Olkoon $t' = \min g_i(t_i)$ ja vastaava ryhmän alkio g'. Lasketaan $T' = g'(T)$ ja sovelletaan menettelyä rekursiivisesti $\min G_{t'_1}(T')$:n etsimiseen.

Jos jossakin vaiheessa on useita t_i, joilla $t' = g_i(t_i)$, kokeillaan kaikki vaihtoehdot rekursiossa vuorollaan.

Tarvittavat stabiloija-aliryhmät voi laskea vaikkapa soveltamalla Schreier-Sims-kannanvaihtoja.

Rataedustajien etsintä

$\mathcal{R}$ on k-osajoukkojen rataedustajien joukko. Lasketaan

$$\mathcal{S} = \{A \cup \{x\} : A \in \mathcal{R}, x \in X \setminus A\},$$

korvataan jokainen S:n joukko ratansa minimiedustajalla ja poistetaan duplikaatit.

Optimointi: kun $A \cup \{x\}$ on tarkistettu, ei tarvitse tarkistaa enää osajoukkoja $G(A \cup \{x\})$. Erityisesti kun g:n Schreier-Sims-kannalle $\{\beta(0), \ldots, \beta(k-1)\} = A$ ja $\beta(k) = x$, ei tarvitse tarkastella osajoukkoja muotoa $A \cup \{c\}$, missä $c \in \mathcal{U}_k(x)$.

Orderly algorithm

Kun ryhmä G vaikuttaa joukkoon X, jonka alkioille on määritelty järjestys, ja joukon X osajoukkoihin indusoidulla tavalla, voidaan osajoukot järjestää seuraavasti: $S \prec T$, jos on olemassa $s \in S$, jolle $s \notin T$, ja kaikilla $x \prec s$ joko $x \in S$ ja $x \in T$ tai $x \notin S$ ja $x \notin T$.

Nyt ratojen minimiedustajat saadaan seuraavalla algoritmilla tyhjästä joukosta lähtien:

```
ORDERLY(S):
    process S
    C = {x : x ∈ X ∧ x > s ∀s ∈ S}
    for x in C:
        if CANONICAL(S ∪ {x}):
            ORDERLY(S ∪ {x})
```

Tod: Merk. $F(S) = S \setminus \{\max S\}$. F on heikosti monotoninen: $S_1 < S_2 \Rightarrow F(S_1) \leq F(S_2)$.

Induktion perustapaus: Kun $n = 0$, kaikki n alkion joukot tulevat käsitellyksi.

Induktioaskel: Jos kaikki n alkion kanoniset osajoukot tulevat käsitellyiksi, myös kaikki $n+1$ alkion kanoniset osajoukot tulevat käsitellyiksi: Olkoon S kanoninen $n+1$-osajoukko. Koska S on kanoninen, $S \leq g(S)$ kaikilla $g \in G$, ja $F(S) \leq F(g(S))$ kaikilla $g \in G$. Todetaan, että $F(g(S)) \leq g(F(S))$ kaikilla $g \in G$ — molemmat saadaan poistamalla $g(S)$:stä yksi alkio, $F(g(S))$:n tapauksessa $\max g(S)$. Koska $F(S) \leq g(F(S))$ kaikilla $g \in G$, $F(S)$ on kanoninen. Nyt S tulee induktion nojalla käsitellyksi, koska $F(S)$ on kanoninen.

Orderly algorithm -esimerkki I

Summapakkaus mod$n$: Annetulla n etsitään suurin joukko $S \subseteq \mathbb{Z}_n$, jolle pätee, että mitään $x \in \mathbb{Z}_n$ ei voida esittää kahdella eri tavalla kahden S:n alkion summana. On helppo määritellä $\mathbb{Z}_n$ alkioille järjestys, ja tämä määrää osajoukoille leksikografisen järjestyksen.

Funktiot muotoa $f(x) = ax + b \pmod{n}$ säilyttävät samat summat samoina ja eri summat eri summina, kunhan $\gcd(a, n) = 1$. Nämä funktiot muodostavat ryhmän. Kanonisuustesti joukolle S voidaan tehdä esim. kokeilemalla kaikille ryhmän alkioille, olisiko $f(S) < S$.

Orderly algorithm -esimerkki II

Binäärikoodi on joukko n-bittisiä binäärisanoja. Minimietäisyyskoodissa kaikkien koodisanaparien tulee erota vähintään d kohdassa. Ekvivalenssi: bittipositioita koodisanoissa voi permutoida vapaasti; yksittäisissä positioissa olevat bitit voi kääntää. Nämä etäisyydet säilyttävät operaatiot määrittelevät ryhmän, joka vaikuttaa koodisanojen joukkoon; kun koodisanoille on määritelty järjestys, koodeille voidaan määritellä lex-järjestys.

Voidaan perustella, että kanonisuustesti voidaan suorittaa seuraavasti: ajatellaan koodia 0/1-matriisina, jonka rivit ovat koodisanoja ja sarakkeet vastaavat positioita. Käydään peräytyvällä haulla läpi kaikki mahdollisuudet järjestää rivit. Kullakin rivijärjestyksellä käännetään bitit niissä sarakkeissa, joissa 1. koodisanassa on 1-bittejä. Lopuksi järjestetään sarakkeet kasvavaan järjestykseen. Pienin eri haaroissa saatu tulos on kanoninen muoto.

Invariantit

Funktio Φ on graafi-invariantti, jos sen arvo ei riipu solmujen järjestyksestä graafin esityksessä:

$$\phi(G) = \phi(\pi(G)) \text{ kaikilla } \pi \in \mathrm{Sym}(V).$$

Esim. kun $\mathcal{V} = \{v_1, \ldots, v_n\}$,

$$\phi(G) = [\deg(v_1), \ldots, \deg(v_n)]$$

ei ole invariantti, mutta monijoukko

$$\phi(G) = \{\deg(v_1), \ldots, \deg(v_n)\}$$

on; astelistasta saadaan siis graafi-invariantti järjestämällä se kasvavaan järjestykseen. Jos $\phi(G_1) \neq \phi(G_2)$, niin G_1 ja G_2 eivät ole isomorfisia.

Solmuinvariantit

Olkoon F perhe graafeja, joiden solmujoukko on V. Funktio $D : F \times V \mapsto R$ on solmuinvariantti, jos sen arvo ei riipu graafin solmujen järjestyksestä.

$$D(G, v) = D(\pi(G), \pi(v)) \text{ kaikilla } \pi \in \mathrm{Sym}(V).$$

Esim. $\deg(v)$ tai v:n sisältävien kolmioiden lukumäärä. Myöhempää varten oletamme, että joukolle R on määritelty järjestys.

Invarianteista

Solmuinvarianteista voidaan konstruoida graafi-invariantteja. Esim. solmuinvariantista $D : F \times V \mapsto R$ saadaan graafi-invariantti

$\phi_{D,r}(G) = |B_D[r]|$, missä $B_D[r] = \{v \in V : D(G, v) = r\}$.

Solmu- ja graafi-invariantteja voidaan yhdistellä uusiksi invarianteiksi:

$$\phi(G) = [\phi_1(G), \ldots, \phi_n(G)]$$

ja

$$D(G, v) = [D_1(G, v), \ldots, D_n(G, v)].$$

$D(G, v)$:n arvojen järjestys voidaan valita esim. listojen leksikografiseksi järjestykseksi.

Solmuinvarianteista saadaan uusia solmuinvariantteja, esim. montako kaarta v:stä menee $B_D[r]$:n solmuihin:

$$D'_r(G, v) = |\{v, v'\} \in E : v' \in B_D[r]|.$$

Sertifikaatti

Kahdella ei-isomorfisella graafilla saattaa olla sama invariantti. Graafiperheelle $\mathcal{F}$ sertifikaatti c on funktio, jolla $c(\mathcal{G}_1) = c(\mathcal{G}_2)$ jos ja vain jos $\mathcal{G}_1, \mathcal{G}_2 \in \mathcal{F}$ ovat isomorfiset.

Sertifikaatti on myös invariantti.

Solmun eksentrisyys ja puun keskusta

Olkoon $d(v_1, v_2)$ solmujen v_1 ja v_2 välisen polun pituus. Olkoon $e(v) = \max_{v' \in V} d(v, v')$ solmun v eksentrisyys.

Yhtenäisen verkon keskusta koostuu solmuista, joiden eksentrisyys on pienin. Puun keskustassa on enintään 2 solmua, jotka ovat naapurit.

Tod: Olkoon $e(v_1) = e(v_2) \leq e(v')$ kaikille $v' \in V$, olkoon $\{v_1, v_2\} \notin E$ ja olkoon v_3 jokin solmu polulta v_1:stä v_2:een. Olkoon v_4 jokin solmu, jolla $d(v_3, v_4) = e(v_3)$. Joko polku v_1:stä v_4:ään tai v_2:sta v_4:ään kulkee v_3:n kautta; siis joko $e(v_1) > e(v_3)$ tai $e(v_2) > e(v_3)$ — ristiriita. Koska keskustan solmut ovat naapureita, ne muodostavat klikin, mutta puussa suurin mahdollinen klikki on kooltaan 2.

Jos puussa on muitakin kuin lehtisolmuja, lehtisolmu ei voi kuulua keskustaan, sillä sen naapurin eksentrisyys on pienempi. Jos tällaisesta puusta poistetaan lehtisolmut, jäljelle jääneiden solmujen eksentrisyys pienenee yhdellä ja keskusta säilyy samana.

Sertifikaatti juurrutetuille puille

Juurrutettu puu on puu, jonka solmuista yksi on nimetty juureksi. Lasketaan sertifikaatti: poistetaan puusta juuri v, jonka jälkeen meillä on yksi tai useampi alipuu. Lasketaan kunkin alipuun sertifikaatti pitäen juurena solmua, joka oli v:n naapuri. Sertifikaatti saadaan lopulta liittämällä yhteen 0, alipuiden sertifikaatit leksikografisessa järjestyksessä, ja 1.

Sertifikaatti puille

Jos puun keskustaan kuuluu vain yksi solmu, juurrutetaan puu siitä ja lasketaan sertifikaatti juurrutetulle puulle.

Jos puun keskustaan kuuluu kaksi solmua, nimetään nämä solmut juuriksi, poistetaan niiden välinen kaari, lasketaan syntyneiden kahden juurrutetun puun sertifikaatit ja liitetään ne leksikografisessa järjestyksessä.

Sertifikaatti puille

Edellisen kalvon sertifikaatin saa laskettua esim. seuraavasti. Tässä etsitään keskipistettä samalla kun sertifikaattia lasketaan, ja sertifikaatin osat menevät hieman eri järjestykseen kuin edellä.

Nimeä kaikki solmut 01:llä.
Niin kauan kuin on vähintään 2 solmua:

aseta $T \leftarrow$ ei-lehtisolmut (deg > 1)

jokaiselle $x \in T$:

- poista x:n nimestä alusta 0 ja lopusta 1
- muodosta joukko Y x:n ja sen naapurilehtisolmujen nimistä
- liitä Y:n alkiot yhteen leksikografisessa järjestyksessä, lisää 0 alkuun ja 1 loppuun, ja kirjoita tulos x:n nimeksi

poista graafista x:n naapurilehtisolmut

Jos jäljellä on 1 solmu, sen nimi on sertifikaatti; jos jäljellä on kaksi solmua, sertifikaatti on niiden nimet leksikografisessa järjestyksessä yhdistettynä.

Sertifikaatti graafeille

Permutoitaessa graafin $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ solmuja permutaatiolla $\pi \in \mathrm{Sym}(\mathcal{V})$ saadaan insidenssimatriisi

$$A_\pi(\mathcal{G})[u,v] = \begin{cases} 1, \text{ jos } \{\pi(u), \pi(v)\} \in \mathcal{E} \\ 0 \text{ muuten} \end{cases}$$

$\mathrm{Num}_\pi(\mathcal{G})$ saadaan lukemalla $A_\pi(\mathcal{G})$:n diagonaalin alapuoliset alkiot binäärilukuna:
$a_{21}\, a_{31} a_{32}\, a_{41} \ldots a_{43}\, a_{51} \ldots a_{54}\, \ldots\, a_{n1} \ldots a_{n\,n-1}$
Yksinkertainen sertifikaatti:

$$\min \{\mathrm{Num}_\pi(G) : \pi \in \mathrm{Sym}(V)\}$$

tulee etsineeksi samalla suurimman riippumattoman joukon, mikä on päätösongelmana $\mathbf{NP}$-täydellinen ongelma. Graafi-isomorfismin ei kuitenkaan uskota olevan niin vaikea ongelma.

Idea: järjestetään solmut solmuinvarianttien määräämään järjestykseen. Ositetaan graafin solmut solmuinvarianttien avulla järjestetyksi ositukseksi B. Olkoon Π_G niiden permutaatioiden joukko, jotka säilyttävät osituksen mukaisen järjestyksen: jos $u \in B_i$ ja $v \in B_j$, niin $\pi(u) < \pi(v)$, jos $i < j$. Nyt

$$\mathrm{cert}(G) = \min \{\mathrm{Num}_\pi(G) : \pi \in \Pi_\mathcal{G}\}.$$

Sertifikaatti graafeille / osituksen parannus

Olkoon $B = [B_{r_0}, \ldots, B_{r_{k-1}}]$ graafin G solmujen (solmuinvariantteihin perustuva) järjestetty ositus. Jos B on diskreetti (kussakin epätyhjässä $B[i]$:ssä tasan yksi alkio), olemme valmiit; muuten pyritään muodostamaan solmut yhä paremmin erottelevia solmuinvariantteja, jolloin myös Π_G pienenee.

Merkitään $D_T(\mathcal{G}, v) = |v' : \{v, v'\} \in \mathcal{E}, v' \in T|$. Invariantti ilmaisee, montako naapuria v:llä on T:ssä.

Parannetaan ositusta: jos on solmut $u, v \in B_{r_i}$ ja jokin $T = B_{r_j}$ siten, että $D_T(\mathcal{G}, u) \neq D_T(\mathcal{G}, v)$, jaetaan B_{r_i} pienemmiksi osiksi D_T:n indusoimalla tavalla D_T:n arvojen mukaiseen kasvavaan järjestykseen. Kun $D_{B_{r_j}}(\mathcal{G}, u) = D_{B_{r_j}}(\mathcal{G}, v)$ kaikilla i ja $u, v \in B_i$, ositus B on tasasuhtainen. On tärkeää, että järjestys, jossa ositusta parannetaan, on invariantti!

Esimerkiksi:

PARANNA(A):
$B \leftarrow A$
olkoon S lista B:n alkioita
while $S \neq \emptyset$:
    poista S:stä sen ensimmäinen joukko T
    jokaiselle $B[i] \in B$ (järjestyksessä):
        jokaiselle h: $L[h] \leftarrow \{v \in B[i] : D_T(\mathcal{G}, v) = h\}$
        jos on useampia kuin yksi epätyhjä $L[h]$:
            korvaa $B[i]$ joukoilla $L[h_1] \ldots L[h_n]$ (järjestyksessä)
            lisää joukot $L[h]$ listan S jatkoksi (järjestyksessä)

Sertifikaatti graafeille

Lasketaan graafin $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ sertifikaatti. Aloitetaan osituksesta $B = \{B_0\}$, missä $B_0 = \mathcal{V}$. Hienonnetaan ositusta, kunnes se on tasasuhtainen, jolloin kuvataan graafin solmut osituksen määräämään järjestykseen ja lasketaan sertifikaatin arvo.

Jos ositus on tasasuhtainen, mutta ei diskreetti, etsitään ensimmäinen joukko, jossa on enemmän kuin 1 alkio, erotetaan joukosta kukin alkio vuorollaan omaksi joukokseen, sovelletaan rekursiota ja valitaan pienin saatu sertifikatti.

CERT($B, \mathcal{G}$):
paranna(B)
jos B diskreetti:
    laske B:stä π; return $\mathrm{Num}_\pi(\mathcal{G})$
muuten:
    etsi pienin i, jolle $|B_i| > 1$
    best $\leftarrow \infty$
    kullekin $x \in B_i$:
        $B' = [B_0, \ldots, B_{i-1}, \{x\}, B_i \setminus \{x\}, B_{i+1}, \ldots]$
        $t \leftarrow \mathrm{cert}(B', \mathcal{G})$
        jos $t < best$: $best \leftarrow t$
    return *best*

Symmetrioiden hyväksikäyttö

Jos kahdessa haun haarassa saadaan sama $\mathsf{Num}_\pi(\mathcal{G}) = \mathsf{Num}_\mu(\mathcal{G})$, niin $\pi(\mathcal{G}) = \mu(\mathcal{G})$, ja $\pi^{-1}\mu(\mathcal{G}) = \mathcal{G}$, joten $\pi^{-1}\mu$ on $\mathcal{G}$:n automorfismi.

Löytyneet automorfismit voidaan kerätä ryhmäksi ja esittää Schreier-Sims-muodossa.

Kun haussa on edetty pisteeseen, jossa B_i on ensimmäinen joukko, jolla $|B_i| > 1$, valitaan ensin jokin $x \in B_i$ ja tutkitaan haara kuten edellä. Tämän jälkeen tehdään kannanvaihto tunnetuille automorfismeille niin, että β_k on B_k:n sisältämä alkio, kun $k < i$, ja $\beta_i = x$. Nyt $\mathcal{U}_i(x)$ on x:n rata tunnetuilla automorfismeilla; muita radan alkioita kuin x ei tarvitse automorfismin perusteella tarkastella.

Tietenkin, jos automorfismiryhmä tunnetaan (edes osaksi) ennalta, voidaan se syöttää valmiiksi.

Joukkojärjestelmien isomorfisuus

Joukkojärjestelmien $(\mathcal{X}, \mathcal{B})$ isomorfisuutta voidaan tarkastella graafi-isomorfismina seuraavasti:

Esitetään joukkojärjestelmä graafina $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, missä $\mathcal{V} = \mathcal{X} \cup \mathcal{B}$, ja $\mathcal{E} = \{\{x, B\} : x \in \mathcal{X}, B \in \mathcal{B}, x \in B\}$. Tämän jälkeen pitää vain pitää huolta siitä, että $\mathcal{X}$-solmut ja $\mathcal{B}$-solmut eivät mene sekaisin; voidaan alustaa sertifikaatin laskenta solmujen osituksesta $[\mathcal{X}, \mathcal{B}]$.

Yleisempi orderly algorithm

Rakenteiden isomorfiaedustajia voidaan generoida seuraavasti: jaetaan rakenteet tasoihin. Kun on konstruoitu tason n isomorfialuokat (isät), konstruoidaan niistä tason $n+1$ isomorfialuokat (lapset) seuraavasti, missä luokka konstruoidaan konstruoimalla sen edustaja:

Konstruoidaan kustakin isästä jokin joukko lapsia. Ongelma on, että jotkin lapset voivat tulla konstruoiduiksi useita kertoja, 1) eri tai 2) samoista isistä.

1. määritellään lapsille kanoninen isä eli se tason n rakenne, josta ne pitää konstruoida, ja tarkistetaan haussa, että lapsi on konstruoitu kanonisesta isästään. Jotta menettely toimisi, on varmistuttava, että jokainen mahdollinen lapsi voidaan generoida kanonisesta isästään.
2. tehdään samasta isästä generoitujen lasten isomorfiakarsinta.

Kustakin tason n rakenteesta p konstruoidaan vuorollaan joukko Q tason $n+1$ rakenteita. Jokaiselle $q \in Q$ lasketaan $F(q) = p'$, jokin tason n rakenne. F:n pitää olla sellainen, että se säilyttää isomorfismin: jos $q_1 \cong q_2$, niin $F(q_1) \cong F(q_2)$. Hylätään ne $q \in Q$, joilla p ja p' eivät ole isomorfiset ($p \ncong p'$). Karsitaan vielä Q:n jäljelle jääneitä alkioita siten, että jäljelle jää vain yksi kunkin isomorfialuokan edustaja.

Orderly algorithm graafeille

Kaikki ei-isomorfiset graafit voidaan konstruoida äskeisellä idealla: tason n rakenteet ovat graafit, joissa on n solmua. Olkoon f funktio, joka poistaa graafista suurinumeroisimman solmun. Olkoon c funktio, joka laskee graafin kanonisen muodon. Nyt $F(G) = f(c(G))$.

Ainoa tason 1 graafi on yksisolmuinen ja kaareton. Tason n graafeista saadaan tason $n+1$ graafit seuraavasti: Tarkastellaan vuorollaan kutakin tason n graafia G. Muodostetaan kaikki graafit, jotka saadaan G:stä lisäämällä uusi solmu $v = |V| + 1$ ja kaaria, joiden toinen päätepiste on v. Lasketaan jokaiselle näin saadulle graafille H kanoninen isä: $G' = F(H)$. Jos $G \not\cong G'$ — voidaan esim. vertailla onko $c(G) \neq c(G')$ — hylätään H. Poistetaan mahdolliset ylimääräiset isomorfiset graafit ja valitaan seuraava n solmun graafi.

Jos kukin n-solmuisten graafien ekvivalenssiluokka on edustettuna, niin kukin $n+1$-solmuisten graafien isomorfiaekvivalenssiluokka on edustettuna, sillä jokaiselle $n+1$-solmuiselle graafille H pätee, että H:n kanssa isomorfinen graafi on mahdollista generoida $f(c(H))$:n kanssa isomorfisesta graafista.

Osajoukon radat

G olkoon $\mathcal{X}$:n permutaatioryhmä ja $S \subseteq \mathcal{X}$. Ryhmän alkio g vaikuttaa S:ään tavallisesti siten, että $g(S) = \{g(s) : s \in S\}$. Näin G siis permutoi myös $\mathcal{X}$:n osajoukkoja.

S:n rata on $G(S) = \{g(S) : g \in G\}$. Jos joukkojärjestelmällä on epätriviaali automorfismiryhmä, sen blokkien joukko on unioni osajoukkojen radoista: jos $S \in B$, on oltava myös $g(S) \in B$ kaikilla $g \in G$.

S:n stabiloija G:ssä on $G_S = \{g \in G : g(S) = S\}$. G_S on G:n aliryhmä, sillä se on suljettu ja epätyhjä.

Lemma: $|G| = |G(S)| \cdot |G_S|$.

Tod: Kuten kalvolla Alkion rata; ajatellaan ryhmän vaikuttavan X:n osajoukkoihin.

Vasempia sivuluokkia on $|G| / |G_S|$, ja jokainen kuvaa S:n eri joukolle, joten $|G(S)| = |G| / |G_S|$.

Esimerkki: Ramseyn luku

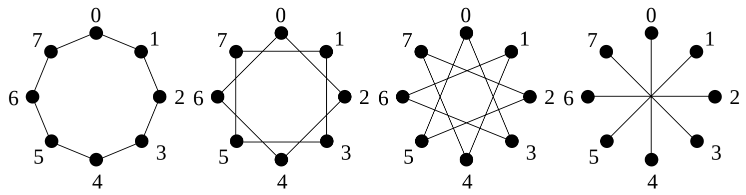
Ramseyn luku $R(k, l)$ on pienin kokonaisluku n, jolla kaikki n solmun graafit sisältävät k solmun klikin tai l solmun riippumattoman joukon. Osoitetaan, että $R(3, 4) > 8$ etsimällä 8 solmun graafi, jossa ei ole 3 solmun klikkiä eikä 4 solmun riippumatonta joukkoa.

Rajoitetaan hakuavaruutta arvaamalla, että voimme löytää ehdot täyttävän graafin $G = (\mathcal{V} = \{0, 1, \ldots, 7\}, \mathcal{E})$, jonka automorfismiryhmä on syklinen:
$\mathrm{Aut}(G) = \langle (0, 1, 2, 3, 4, 5, 6, 7) \rangle$.
Tällöin $\mathcal{E}$ on unioni $\mathcal{V}$:n 2-osajoukkojen ratoja.

Esimerkki: Ramseyn luku

$\mathcal{V}$:n 2-osajoukkojen radat ovat

$$\begin{aligned}
O_1 &= \{\{0,1\},\{1,2\},\{2,3\},\{3,4\},\{4,5\},\{5,6\},\{6,7\},\{0,7\}\} \\
O_2 &= \{\{0,2\},\{1,3\},\{2,4\},\{3,5\},\{4,6\},\{5,7\},\{0,6\},\{1,7\}\} \\
O_3 &= \{\{0,3\},\{1,4\},\{2,5\},\{3,6\},\{4,7\},\{0,5\},\{1,6\},\{2,7\}\} \\
O_4 &= \{\{0,4\},\{1,5\},\{2,6\},\{3,7\}\}.
\end{aligned}$$



Kokeilemalla huomataan, että kaarijoukot $\mathcal{E} = O_3 \cup O_4$ ja $\mathcal{E} = O_1 \cup O_4$ täyttävät ehdot.

Esimerkki: Ramseyn luku $R(5,9) > 120$

On olemassa 120 solmun verkko, joka ei sisällä 5-klikkiä eikä 9-riippumatonta joukkoa. Se voidaan löytää tabu-haulla:

- Jaetaan kaaret ratoihin kuten edellä (syklinen automorfismiryhmä).
- Valitaan satunnainen osajoukko ratoja.
- Lisätään verkkoon tai poistetaan siitä toistuvasti sellainen osajoukko, että muutos johtaa verkkoon, jossa on mahdollisimman vähän 5-klikkejä tai 9-riippumattomia osajoukkoja.
- Ei kuitenkaan koskaan lisätä tai poisteta kaarijoukkoa, joka on lisätty tai poistettu viimeisen 12 siirron aikana.

$(\mathcal{V}, \mathcal{E})$, missä $\mathcal{E} = \{\{v, v+d \pmod{120}\} : d \in S, v \in \mathcal{V}\}$, $\mathcal{V} = \{0, \ldots, 119\}$ ja $S = \{2, 3, 6, 7, 13, 15, 17, 18, 19, 20, 22, 23, 28, 29, 31, 33, 41, 42, 43, 45, 48, 52, 53, 54, 60\}$, toteuttaa ehdot.

Symmetristen objektien generointi

Jos kombinatorista objektia etsittäessä hakuavaruus on liian suuri, voidaan rajoittaa hakuavaruutta rajoittamalla tarkastelu objekteihin, joilla on (vähintään) tietty automorfismiryhmä.

Esimerkki: $\mathcal{X} = \{0, \ldots, 24\}$ ja $G = \langle(0, 1, \ldots, 24)\rangle$. Kun $\mathcal{B}$ on joukkojen $\{0, 8, 13\}$, $\{0, 2, 3\}$, $\{0, 4, 11\}$ ja $\{0, 6, 15\}$ ratojen unioni, $(\mathcal{X}, \mathcal{B})$ on STS (25). (Steinerin kolmikkojärjestelmä, jossa $|\mathcal{X}| = 25$; jokainen $\mathcal{X}$:n pari esiintyy tasan yhdessä $\mathcal{B}$:n osajoukossa).

Voitaisiin tietysti myös luetella kaikki sata 3-osajoukkoa.

Ratainsidenssimatriisit

Ratainsidenssimatriisi: kun G on $\mathcal{X}$:n permutaatioryhmä ja $0 \leq t \leq k \leq |\mathcal{X}|$, ratainsidenssimatriisi A_{tk} on $N_t \times N_k$-matriisi, jossa rivi i vastaa t-osajoukkojen rataa Δ_i, sarake j vastaa k-osajoukkojen rataa Γ_k, ja $a_{ij} = \left|\left\{K \in \Gamma_j\right\} : K \supset T_0\right|$, missä $T_0 \in \Delta_i$.

Osoittautuu, että $a_{ij} = \left|\left\{K \in \Gamma_j\right\} : K \supset T_0\right|$ ei riipu valitusta $T_0 \in \Delta$:sta:

Jos $T_0, T_0' \in \Delta$, on olemassa $g \in G$, jolla $g(T_0) = T_0'$. Jos $T_0 \subseteq K \in \Gamma$, niin $T_0' \subseteq g(K)$.

Ratainsidenssimatriisiesimerkki

Olkoon $\mathcal{X} = \{0, \ldots, 4\}$ ja $G = \langle(0, 1, 2, 3, 4)\rangle$.
2-osajoukkojen radat ovat

$$\begin{aligned} \Delta_1 &= \{\{0,1\}, \{1,2\}, \{2,3\}, \{3,4\}, \{4,0\}\} \text{ ja} \\ \Delta_2 &= \{\{0,2\}, \{1,3\}, \{2,4\}, \{3,0\}, \{4,1\}\}. \end{aligned}$$

3-osajoukkojen radat ovat

$$\begin{aligned} \Gamma_1 &= \{\{0,1,2\}, \{1,2,3\}, \{2,3,4\}, \{3,4,0\}, \{4,0,1\}\} \text{ ja} \\ \Gamma_2 &= \{\{0,1,3\}, \{1,2,4\}, \{2,3,0\}, \{3,4,1\}, \{4,0,2\}\}. \end{aligned}$$

Ratainsidenssimatriisi $A_{23} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$. Esimerkiksi a_{22} voidaan laskea valitsemalla $T_0 = \{0, 2\} \in \Delta_2$ ja toteamalla, että T_0 sisältyy Γ_2:n joukoista kahteen ($\{2, 3, 0\}$ ja $\{4, 0, 2\}$).

Ratainsidenssimatriisin laskeminen

Seuraava algoritmi laskee ratainsidenssimatriisin naiivisti. R ja S ovat t- ja k-osajoukkojen ($t \leq k$) rataedustajien joukot.

```
for g ∈ G:
    for T ∈ R:
        for K ∈ S:
            if T ⊆ g(K):
                A[T,K] ← A[T,K] + 1
for K ∈ R:
    stab ← 0
    for g ∈ G:
        if g(K) = K:
            stab ← stab + 1
    for T ∈ R:
        A[T,K] ← A[T,K]/stab
```


Burnsiden lemma

(Frobenius, 1887) Jos äärellinen ryhmä G vaikuttaa äärelliseen joukkoon X, ja N on ratojen lukumäärä, niin

$$N = \frac{1}{|G|} \sum_{g \in G} F(g),$$

missä $F(g)$ on niiden $x \in \mathcal{X}$ lukumäärä, joille $gx = x$.

Tod: yo. summassa kukin $x \in \mathcal{X}$ lasketaan $|G_x|$ kertaa (G_x:n määritelmän mukaan). Jos x ja y ovat samalla radalla, niin $|G_x| = |G_y|$, joten radan kaikki $|G|/|G_x|$ alkiota lasketaan $|G_x|$ kertaa, yhteensä siis $|G|$ kertaa. Joka radasta summaan tulee $|G|$, jolla jakamalla saadaan ratojen määrä.

Burnsiden lemma

Tarkastellaan k-osajoukkoja, joihin permutaatioryhmä vaikuttaa luonnollisella tavalla.

Merkitään $F(g)$:llä, montako k-osajoukkoa g kiinnittää:

$$F(g) = |\{S \subseteq \mathcal{X}\} : |S| = k \text{ ja } g(S) = S|.$$

$F(g)$:n laskemiseksi lasketaan ensin g:n syklien pituusjakauma ja merkitään

$$\text{type}(g) = [t_1, \ldots, t_n],$$

missä t_i on i-pituisten syklien lukumäärä. Jos $g(S) = S$, niin S on unioni joidenkin g:n syklien alkioista.

Jos S:ssä on c_i sykliä, joissa on i solmua, niin on oltava $c_i \leq t_i$, ja $k = \sum_i ic_i$. Annetuille c_i:n arvoille sellaisia joukkoja on $\prod_i \binom{t_i}{c_i}$.

Burnsiden lemma

Lasketaan annetulle k ja annetulle $[t_1, \ldots, t_n]$ rekursiolla kaikki mahdolliset c_i-yhdistelmät, joille $c_i \leq t_i$ ja $k = \sum_i ic_i$:

```
CHIG(n,k,i,t):
if i = 1: χ ← 0
if i = n + 1:
    if k = 0:
        χ ← χ + ∏_i (t_i choose c_i)
    return
C_i ← {0,...,min(t_i, ⌊k/i⌋)}
for x ∈ C_i:
    c_i ← x
    CHIG(n, k − ic_i, i + 1, t)
return χ
```

k-permutaatioiden radat

σ on $\mathcal{X}$:n k-monikko, jos $\sigma = [x_1, \ldots, x_k]$, missä $x_i \in \mathcal{X}$. Jos $x_i \neq x_j$, kun $i \neq j$, σ on $\mathcal{X}$:n k-permutaatio. $\mathcal{X}$:n k-permutaatioiden joukko on $\Pi_k(\mathcal{X})$.

Vastaavasti kuin osajoukoille permutaatioryhmän luonnollinen vaikutus on,

$$g(\sigma) = [g(x_1), \ldots, g(x_k)]$$

ja

$$G(\sigma) = \{g(\sigma) : g \in G\}.$$

Burnsiden lemman tapaan

$$\vec{N}_k = \frac{1}{|G|} \sum_{g \in G} \vec{\chi}_k(g),$$

missä

$$\vec{\chi}_k = |\{\sigma \in \Pi_k(\mathcal{X}) : g(\sigma) = \sigma\}|$$

Burnsiden lemma – esimerkki

Montako oleellisesti erilaista viisiraitaista lippua on, kun kukin raita on joko sininen, valkoinen tai punainen? Lippuja ei pidetä oleellisesti erilaisina, jos toinen saadaan toisesta peilaamalla.

Lippuja voidaan tarkastella listoina $[c_1, c_2, \ldots, c_n]$, missä kukin $c_i \in C$. Vaihtoehtoja on siis $|C|^n$. Ryhmä koostuu kahdesta permutaatiosta, identiteetistä ja peilauksesta τ, joka vaikuttaa lippuun s.e. $\tau[c_1, \ldots, c_n] = [c_n, \ldots, c_1]$. Lasketaan kummallekin permutaatiolle π, kuinka monta lippua se kiinnittää. $F(\mathbf{I}) = |C|^n$ ja $F(\tau) = |C|^{\lceil n/2 \rceil}$ ja lippujen määrä on $N = \frac{1}{2}\left(|C|^n + |C|^{\lceil n/2 \rceil}\right)$ eli tässä esimerkissä $\frac{1}{2}(243 + 27) = 135$.

Sivuluokat, transversaalit ja radat

Olkoon G joukon $\mathcal{X}$ permutaatioryhmä ja $(\mathcal{X}, \mathcal{B})$ joukkojärjestelmä. Kun $g \in G$, merkitään

$$g(\mathcal{B}) = \{g(S) : S \in \mathcal{B}\}.$$

Nyt $\mathcal{B}$:n stabiloija on aliryhmä

$$H = \{g \in G : g(\mathcal{B}) = \mathcal{B}\} \subseteq G.$$

On siis olemassa H:n vasen transversaali $T = \{g_1, \ldots, g_r\}$.

Teoreema: $\mathcal{B}$:n radassa on $r = |G| / |H|$ alkiota, ja $G(\mathcal{B}) = T(\mathcal{B}) = \{g_1(\mathcal{B}), \ldots, g_r(\mathcal{B})\}$.

Todistus: Edelleen kalvon Ryhmän vaikutus mukaan. Nyt ryhmä G vaikuttaa joukkojärjestelmiin. Vasemmassa transversaalissa on $|G| / |H|$ alkiota, joista jokainen kuvaa $\mathcal{B}$:n eri joukkojärjestelmälle – jos olisi $g_i(\mathcal{B}) = g_j(\mathcal{B})$, niin $g_i^{-1} g_j(\mathcal{B}) = g_i^{-1} g_i(\mathcal{B}) = \mathcal{B}$, ja $g_i^{-1} g_j \in H$. Nyt g_i ja g_j kuuluvat samaan sivuluokkaan, sillä $g_i H = g_i\left(g_i^{-1} g_j H\right) = g_j H$, ja on oltava $g_i = g_j$; transversaaliinhan kuuluu vain yksi edustaja kustakin sivuluokasta.

Transversaalin laskeminen

Laskemalla $G_{\mathcal{B}} \subseteq G$:n transversaali T ja sitten $T(\mathcal{B})$ saadaan $\mathcal{B}$:n rata $G(\mathcal{B})$.

Alla lasketaan transversaali varsin naiivisti tarkistamalla kullekin ryhmän alkiolle, onko jokin saman sivuluokan alkio jo transversaalissa. Ellei, lisätään alkio transversaaliin.

```
TRANSVERSAL(H, G):
r ← |G| / |H|
T ← ∅
for g ∈ G:
      for t ∈ T:
            if t⁻¹g ∈ H:
                  goto skip
      T ← T ∪ {g}
      if |T| ≥ m:
            return T
      skip:
```