BAB III

ALGORITMA UNTUK PENYELESAIAN MASALAH MAKING CHANGE

3.1. Making Change

Berikut akan digambarkan masalah making change. Seorang pelanggan membeli suatu barang dagangan yang mempunyai harga 100 - C ($0 \leq C \leq 100$), memberikan satu lembar uang kertas 100 kepada kasir. Permasalahannya bagaimana selanjutnya kasir memberikan sisa kembalian kepada pelanggan dengan mata uang sejumlah C.

Masalah making change dapat disajikan sebagai berikut. Diberikan $W = \{w_1, w_2, \ldots, w_n\}$ sebagai himpunan bilangan bulat positif tertentu atas $n \geq 1$, sedemikian sehingga $w_1 < w_2 < \ldots < w_n$. W dinyatakan sebagai denominasi mata uang yang disediakan dengan jumlah tidak terbatas. Jika besarnya jumlah nominal yang harus dibayarkan kepada pelanggan adalah sebesar C , maka dapat dibentuk suatu model matematika dari permasalahan making change, yaitu

$$\sum_{i=1}^{n} a_i w_i = C$$

dengan a_i bilangan bulat positif.

Harganya adalah $\sum_{i=1}^{n} a_i$

Definisi 2

Jika $W = \{w_1, w_2, \ldots, w_n\}$ adalah himpunan mata uang yang didefinisikan. Untuk setiap bilangan bulat tidak negatif C dan bilangan positif $m \leq n$ terdapat $(a_1, a_2, \ldots, a_m)$, sehingga memenuhi

$$\sum_{i=1}^{m} a_i w_i = C$$

maka $(a_1, a_2, \ldots, a_m)$ dikatakan distribusi-m dari C.

Definisi 3

Distribusi-m dari C $(a_1, a_2, \ldots, a_m)$, dikatakan minimal jika diberikan distribusi-m yang lain dari C yaitu $(b_1, b_2, \ldots, b_m)$, sedemikian sehingga

$$\sum_{i=1}^{m} a_i \leq \sum_{i=1}^{m} b_i$$

dengan $\sum_{i=1}^{m} a_i$ adalah jumlah total mata uang yang dibutuhkan.

Diantara sekian banyak permasalahan dalam Ilmu Komputer, terdapat beberapa masalah yang bisa diselesaikan menggunakan lebih dari satu algoritma. Masing-masing algoritma memiliki bentuk yang berbeda.

Demikian pula pada masalah making change, terdapat lebih dari satu algoritma penyelesaian. Dalam Tugas Akhir

ini akan dibahas dua buah algoritma untuk menyelesaikan masalah making change, yaitu Algoritma Greedy dan MINDIST.

3.2. ALGORITMA GREEDY

Algoritma Greedy mempunyai sifat-sifat atau karakteristik yang bersifat umum karena ternyata algoritma ini tidak hanya digunakan pada masalah making change saja, contohnya masalah penentuan spanning tree minimum pada graph, masalah scheduling (penjadwalan), dan lain sebagainya. Secara umum Algoritma Greedy dan permasalahan-permasalahan yang diselesaikan mempunyai sifat sebagai berikut :

1. Adanya permasalahan yang harus diselesaikan dengan suatu cara pengoptimalan. Untuk membuat solusi dari permasalahan tersebut harus dipunyai himpunan calon atau kandidat, dalam hal ini tersedianya mata uang yang didefinisikan sebagai $W = \{w_1, w_2, \ldots, w_n\}$.
2. Selama algoritma diproses, dijumlahkan dua himpunan yang berbeda. Himpunan pertama mengandung kandidat-kandidat yang sudah didefinisikan dan sudah dipilih, himpunan yang lain berisi kandidat-kandidat yang sudah didefinisikan dan tidak dipilih.
3. Terdapat suatu fungsi yang mengecek apakah suatu anggota himpunan kandidat tersebut memberikan solusi permasalahan, dengan mengabaikan pertanyaan-pertanyaan optimalitas. Contohnya, apakah mata uang

yang sudah dipilih menambah atau menaikkan jumlah yang akan dibayarakan ?

4. Adanya fungsi yang mengecek apakah himpunan kandidat tersebut fisibel. Apakah mungkin atau tidak melengkapi himpunan tersebut dengan menambahkan kandidat lagi untuk menentukan sedikitnya satu solusi permasalahan. Diharapkan permasalahan ini sedikitnya mempunyai satu penyelesaian yang dapat ditentukan dengan menggunakan kandidat dari kumpulan yang tersedia.
5. Terdapat fungsi penyeleksi, yang menunjukkkan kapan saja suatu kandidat dapat dipilih atau tidak, yaitu kandidat yang paling memungkinkan.
6. Adanya suatu fungsi sasaran (fungsi obyektif) yang memberikan harga dari suatu solusi yang didapatkan. Jumlah mata uang yang diperlukan untuk making change. Tidak seperti ketiga fungsi yang disebutkan di atas, fungsi sasaran tidak muncul secara eksplisit pada Algoritma Greedy.

Untuk menyelesaikan permasalahan di atas, ditentukan suatu himpunan kandidat yang memberikan suatu solusi.Algoritma Greedy prosesnya berlangsung langkah demi langkah (*step by step*). Harga awal dari himpunan yang terpilih adalah kosong. Kemudian pada tiap-tiap step dilakukan percobaan dengan menjumlahkan ke himpunan ini dan menyisakan kandidat yang tidak dicoba. Jika himpunan

kandidat yang membesar sudah tidak fisibel, kandidat yang terakhir ditambahkan dibuang. Pada kasus ini kandidat yang sudah dicoba dan dibuang tidak dipandang lagi. Tetapi jika himpunan yang membesar itu masih fisibel ditambahkan kandidat terakhir ke himpunan kandidat terpilih. Tiap saat himpunan diperbesar dan terus menerus dilakukan cek apakah menghasilkan suatu penyelesaian permasalahan tersebut. Secara umum Algoritma Greedy dapat diformulasikan sebagai berikut :

```
function Greedy (C : set)
        {W adalah himpunan kandidat }
        S  ←— Ø  {Akan dicari solusi dalam himpunan S}
        while C ≠ Ø and not solution(S) do
           x ←— select(C)
           C ←— C \ {x}
           if feasibel (S U {x}) then
              S ←— S U {x}
        if solution(S) then
              return S
        else
              return "Tidak didapatkan solusi"
```

Algoritma Greedy secara umum

Setelah dapat memahami Algoritma Greedy secara umum, dengan mudah dapat disusun Algoritma Greedy pada masalah making change. Dalam bahasa yang sederhana ALgoritma Greedy pada masalah making change adalah sebagai berikut:

step 1

Dimulai dengan memberikan harga awal terhadap kumpulan mata uang yang dihasilkan sama dengan himpunan kosong.

$$S = \{\ \} = \emptyset$$

step 2

Dipilih sebuah mata uang paling besar nilainya yang tersedia ($W = \{w_1, w_2, \ldots, w_n\}$) dan tidak melebihi jumlah yang harus dibayarkan. Mata uang tersebut adalah salah satu penyelesaian. Ambil lagi mata uang tersebut sebagai penyelesaian jika masih memenuhi, hingga mata uang tersebut tidak Jika jumlah yang harus dibayarkan sudah terkumpul maka selesai. Jika belum ke step 3. Mata uang yang sudah pernah dipilih tidak diperhatikan lagi (dibuang).

step 3

Ulangi step 2 sampai kondisi yang dinginkan terpenuhi, yaitu jumlah yang harus dibayarkan terkumpul. Dengan membentuk suatu distribusi-m bilangan bulat positif dari jumlah yang didefinisikan

(C), misalnya (a_1, a_2, ... , a_m), yang sesuai

$$\sum_{i=1}^{n} a_i w_i = C$$

Algoritma Greedy pada masalah making change di atas dapat diformulasikan sebagai berikut :

function *Greedy (Making Change)* (C) : himpunan mata uang

{Pembayaran C unit menggunakan kemungkinan jumlah mata uang , konstanta W menentukan harga satuan mata uang yang digunakan}

const $W = \{w_1, w_2, w_3, \ldots, w_n\}$

S ←— Ø *{S adalah himpunan mata uang solusi}*

s ←— Ø *{s adalah jumlah dari mata uang dalam S}*

while s ≠ C do

x ←— harga terbesar dalam W yang memenuhi

while s + x ≤ C do

S ←— S U *{mata uang dengan harga x}*

s ←— s + x

If *data yang dimaksud tidak ada* then

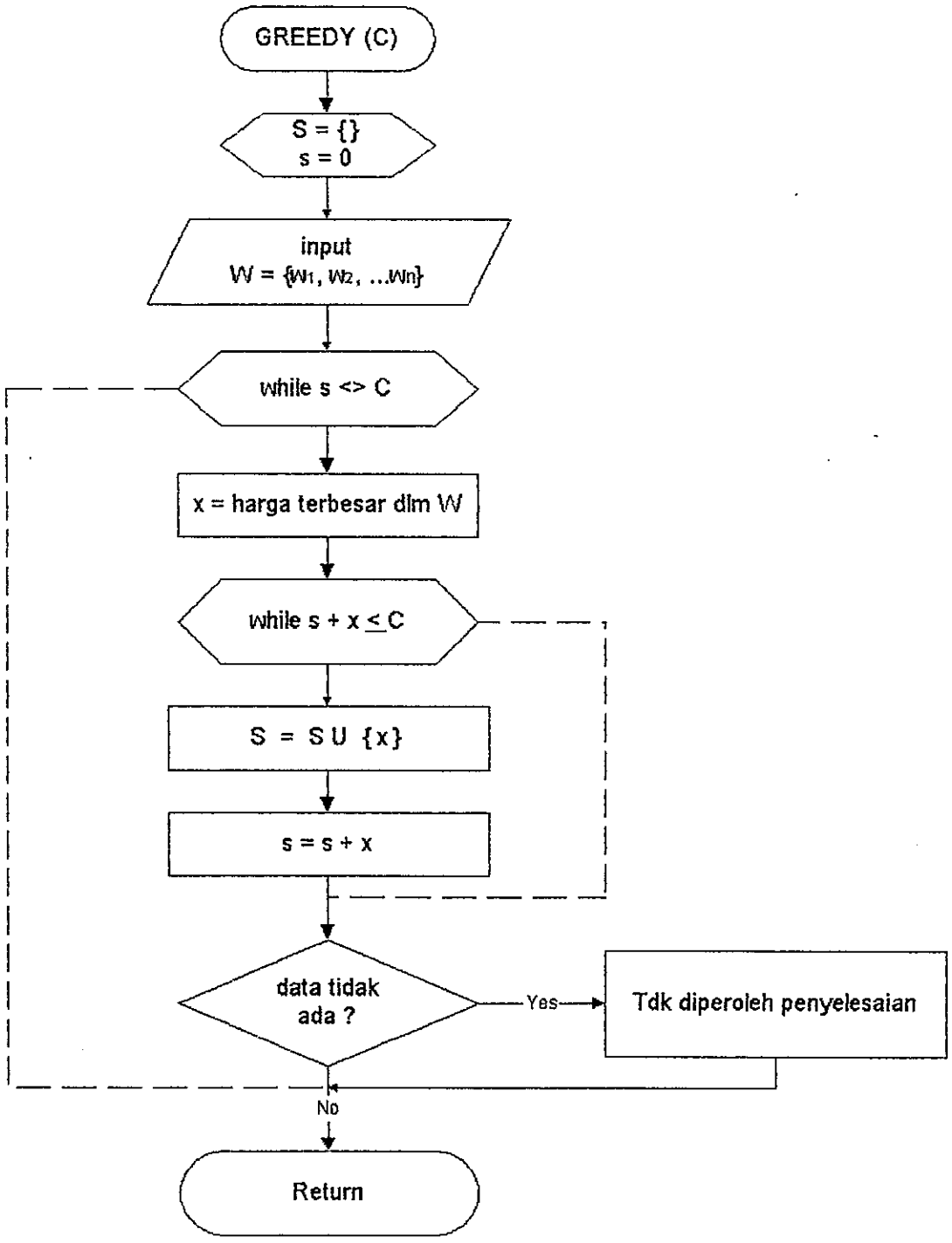
return *'Solusi tidak didapatkan'*

return S

Algoritma Greedy pada making Change

Algoritma Greedy juga dapat disajikan dalam bentuk flowchart seperti bagan berikut ini :

Flowchart GREEDY (C)



Pertama kali yang harus diketahui adalah jumlah uang yang harus dibayarkan, dalam algoritma di atas diberi notasi C. Sesudah itu harus didefinisikan satuan mata uang yang dipergunakan. Satuan yang digunakan dinotasikan dengan $W = \{w_1, w_2, w_3, \ldots, w_n\}$, artinya terdapat n satuan mata uang yang diberlakukan. Nilai-nilai mata uang yang diberikan harus mempunyai persediaaan dalam jumlah tak terbatas pada masing-masing harga satuan, agar algoritma ini selalu memberikan penyelesaian atas permasalahan tersebut. Jika tidak Algoritma Greedy mungkin tidak dapat dilaksanakan.

Untuk mempermudah penggambaran tentang cara kerja, dimisalkan terdapat himpunan S yang akan memuat kandidat-kandidat yang telah diseleksi oleh suatu fungsi penyeleksi. Himpunan S diberi harga awal himpunan kosong. Terdapat pula s yang berlaku sebagai variabel kontrol. Variabel s merupakan jumlah kumulatif kandidat-kandidat yang memenuhi. Harga awal s adalah nol.

Setelah pendefinisian harga awal dan satuan mata uang, maka sekarang algoritma Greedy mulai bekerja. Iterasi kesatu dengan statemen kontrol

$$\textbf{while } s \neq C$$

pasti dilakukan karena pasti $s < C$, untuk $C > \emptyset$.
Dimulai pengambilan kandidat terbesar yang memungkinkan misalnya x, kemudian dilakukan iterasi kedua yaitu yang memenuhi fungsi seleksi apakah

$$s + x \leq C$$

Jika kondisinya benar maka x adalah salah satu penyelesaian sehingga dimasukkan dalam himpunan S. Dan juga sekarang harga s tidak nol lagi tetapi sudah ditambah dengan x sesuai dengan statemen

$$s = s + x$$

Iterasi kedua terus dilakukan selama $s + x \leq C$ masih terpenuhi untuk suatu x. Bila kandidat tersebut masih memenuhi maka kandidat tersebut dimasukkan ke dalam himpunan S sebagai penyelesaian. Jika suatu kandidat x sudah tidak memenuhi lagi untuk menambah harga s, yaitu

$$s + x > C$$

maka kandidat tersebut akan dibuang (dianggap bukan kandidat lagi) dan dianggap tidak ada. Maka iterasi kedua tahap pertama dihentikan.

Lakukan kembali tahapan iterasi kesatu dengan lebih dahulu mengecek apakah keadaan optimal telah tercapai

$$s = C$$

Jika sudah, maka iterasi pertama dihentikan. Jika belum tahapan seleksi dilakukan lagi dengan cara yang sama. Diambil kandidat terbesar yang memungkinkan.

Demikian seterusnya hingga keadaan tercapai yaitu s = C. Perlu diperhatikan bahwa, sekali mata uang termasuk sebagai penyelesaian maka untuk selamanya tetap sebagai penyelesaian.

Jika dalam beberapa kali iterasi tidak ditemukan x yang memenuhi maka tidak diperoleh penyelesaian.

Himpunan S yang didapatkan merupakan distribusi-m minimal dari C, sedangkan jumlah banyaknya elemen di dalamnya menyatakan jumlah mata uang yang dipergunakan.

THEOREMA 1

Jika $W = \{w_1, w_2, \ldots, w_m\}$ adalah satuan mata uang yang didefinisikan dengan $w_1 < w_2 < \ldots < w_m$ dan $w_1 = 1$ maka dapat ditemukan suatu penyelesaian pada masalah making change.

BUKTI

Misalkan C adalah jumlah yang harus dibayarkan , dan mata uang yang diberlakukan didefinisikan sebagai $W = \{w_1, w_2, \ldots, w_m\}$. Andaikan diambil $w_1 \neq 1$ maka terdapat suatu kasus pada penyelesaian masalah making change menggunakan algoritma Greedy yaitu :

$$\begin{aligned} \text{mod } (C, w_m) &\neq \emptyset = z_1 \\ \text{mod } (z_1, w_{m-1}) &\neq \emptyset = z_2 \\ &\vdots \\ \text{mod } (z_{n-2}, w_2) &\neq \emptyset = z_{n-1} \\ \text{mod } (z_{n-1}, w_1) &\neq \emptyset = z_n \end{aligned}$$

Karena sampai mata uang terkecil (w_1) sisa jumlah yang harus dibayarkan tidak habis dibagi yaitu terdapat sisa sebesar z_n, sehingga tidak diperoleh penyelesaian. Maka kontradiksi, pengandaian harus diingkar yang benar adalah $w_1 = 1$. ■

Contoh 1 :

Andaikata seseorang hidup di suatu negara yang memberlakukan beberapa jenis mata uang yaitu dollar (1ØØ sen), quarters (25 sen), dimes (1Ø sen), nickels (5 sen), dan pennies (1 sen).

Permasalahannya adalah bahwa seorang kasir harus memberikan sisa pengembalian kepada pembeli sebesar $.2.89 (289 sen).

Penyelesaian

Diketahui :

W = { 1, 5, 1Ø, 25, 1ØØ} dan

C = $ 289

Pemberian harga awal

S = { }

s = Ø

Tahapan yang dilakukan oleh Algoritma Greedy akan diperlihatkan pada tabel berikut ini.

Tabel Proses Algoritma Greedy

i	W	x	s+x	S	Ket
	100,25,10,5,1		0	ø	
1	100,25,10,5,1	100	100	100	
2	100,25,10,5,1	100	200	100,100	
3	100,25,10,5,1	100	300	100,100	100 tm dibuang
4	25,10,5,1	25	225	100,100,25	
5	25,10,5,1	25	250	100,100,25 25	
6	25,10,5,1	25	275	100,100,25 25,25	
7	25,10,5,1	25	300	100,100,25 25,25	25 tm dibuang
8	10,5,1	10	285	100,100,25 25,25,10	
9	10,5,1	10	295	100,100,25 25,25,10	10 tm dibuang
10	5,1	5	290	100,100,25 25,25,10	5 tm dibuang
11	1	1	286	100,100,25 25,25,10,1	
12	1	1	287	100,100,25 25,25,10,1 1	

i	W	x	s+x	S	Ket
13	1	1	288	100,100,25 25,25,10,1 1,1	
14	1	1	289	100,100,25 25,25,10,1 1,1,1	s = s+x s = C optimal

Keterangan

tm = tidak memenuhi yaitu s + x > C

Dari tabel yang telah tersusun maka penyelesaian permasalahan di atas sudah terjawab, yaitu mata uang yang digunakan untuk membayar adalah sesuai dengan himpunan S yang terdiri dari 10 elemen

S = {100, 100, 25, 25, 25, 10, 1, 1, 1, 1}

Artinya jumlah mata uang yang digunakan adalah 10 , masing-masing 2 dollars, 3 quarters, 1 dimes, dan 4 pennies. ■

Mengacu pada contoh masalah making change di atas, terdapat suatu cara tentang gambaran umum Algoritma Greedy dapat disamakan ke gambaran khusus dari permasalahan tersebut.

1. Kandidat adalah himpunan mata uang yang mewakili. Contoh yang diambil adalah 1ØØ, 25, 1Ø, 5, dan 1 sen, dengan mata uang tiap-tiap harga satuan yang mencukupi sehingga tidak pernah habis (tetapi kumpulan kandidat itu terbatas).
2. Fungsi penyelesaian melakukan cek apakah harga dari mata uang terpilih pada saat yang sama sudah tepat memenuhi jumlah yang dibayarkan.
3. Himpunan mata uang adalah fisibel jika total harganya tidak melebihi jumlah yang dibayarkan.
4. Fungsi seleksi memilih mata uang harga terbesar yang termuat dalam himpunan kandidat.
5. Fungsi sasaran menghitung jumlah mata uang yang digunakan pada penyelesaian ini.

Namun meskipun algoritma Greedy memiliki sejumlah kelebihan dibanding dengan algoritma yang lain. Algoritma tersebut memiliki kekurangan atau kelemahan.

Contoh 2 :

Jika dipilih beberapa harga satuan mata uang untuk melakukan making change yaitu $1, $5, dan $11. Jumlah yang harus dibayarkan adalah $15. Maka tentukan jumlah cacah mata uang yang digunakan.

Penyelesaian

Diketahui :

W = { 1, 5, 11} dan

C = $ 15

Pemberian harga awal

S = { }

s = Ø

Tahapan yang dilakukan oleh Algoritma Greedy adalah sebagai berikut :

Iterasi pertama

Algoritma akan memilih harga satuan terbesar yaitu $11 sehingga sekarang

s = s + x = Ø + 11 = 11

s = 11 ≤ C maka $11 masuk sebagai penyelesaian

Berikutnya dicoba lagi dengan harga satuan yang sama

s = s + x = 11 + 11 = 22

s = 22 > C maka $11 dibuang dari himpunan kandidat

Iterasi kedua

Selanjutnya diambil satuan terbesar berikutnya yaitu $5 sehingga

s = s + x = 11 + 5 = 16

s = 16 > C maka $5 dibuang dari himpunan kandidat.

Iterasi ketiga

Tinggal satu kandidat lagi yaitu $1

s = s + x = 11 + 1 = 12

s = 12 ≤ C maka $1 masuk sebagai penyelesaian

Coba lagi dengan kandidat yang sama

s = s + x = 12 + 1 = 13

s = 13 ≤ C maka $1 masuk sebagai penyelesaian

Coba lagi dengan kandidat yang sama

s = s + x = 13 + 1 = 14

s = 14 ≤ C maka $1 masuk sebagai penyelesaian

Coba lagi dengan kandidat yang sama

s = s + x = 14 + 1 = 15

s = 15 = C maka $1 masuk sebagai penyelesaian

Iterasi dihentikan karena s = C, maka diperoleh himpunan hasil

S = { 11, 1, 1, 1, 1 }

dengan jumlah mata uang = 5, artinya jumlah satuan mata uang minimal yang dibutuhkan adalah 5, yaitu 1 mata uang $11, dan 4 mata uang $1.

Padahal dengan ketiga satuan mata uang yang didefinisikan yaitu $1, $5, $11, dapat dilakukan making change sebesar $15, cukup dengan 3 mata uang dengan harga satuan $5. ■

3.3. ALGORITMA MINDIST

Algoritma kedua yang akan disajikan dalam Tugas Akhir ini adalah Algoritma MINDIST. Namun sebelum dipaparkan cara kerjanya akan diuraikan beberapa kaidah yang harus diperhatikan.

THEOREMA 2

Diberikan $C = qw_2 + r$ dengan $\emptyset \leq r < w_2$, maka (r,q) adalah distribusi minimal-2 dari C.

BUKTI

Diberikan (r',q') sebagai distribusi-2 dari C, maka harus memenuhi $q' \leq q$ atau $q - q' \geq \emptyset$.

Karena $w_2 > w_1 = 1$, $w_2 - 1 \geq \emptyset$ sehingga

$$\begin{aligned} r' + q' &= (C - q'w_2) + q' = r + qw_2 - q'w_2 + q' \\ &= r + q + (q - q')(w_2 - 1) \\ &\geq r + q \end{aligned}$$

oleh karena itu (r,q) adalah distribusi minimal-2 dari C ■

THEOREMA 3

Diberikan $(a_1, a_2, \ldots, a_{m-1})$ sebagai suatu distribusi-$(m-1)$ minimal dari $C < w_m$ maka $(a_1, a_2, \ldots, a_{m-1}, \emptyset)$ adalah suatu distribusi-m minimal dari C.

BUKTI

Karena $(a_1, a_2, \ldots, a_{m-1})$ suatu distribusi-(m-1) minimal dari C maka

$$\sum_{i=1}^{m-1} a_i = a_1 + a_2 + \ldots + a_{m-1}$$

Jika $(a_1, a_2, \ldots, a_{m-1}, \emptyset)$ distribusi dari C yang lain maka

$$\sum_{i=1}^{m} a_i = a_1 + a_2 + \ldots + a_{m-1} + \emptyset = a_1 + a_2 + \ldots + a_{m-1}$$

jadi

$$\sum_{i=1}^{m-1} a_i = \sum_{i=1}^{m} a_i$$

maka $(a_1, a_2, \ldots, a_{m-1}, \emptyset)$ adalah distribusi minimal dari $C < w_m$ ■

LEMMA 1

Suatu distribusi-m minimal dari C ada untuk setiap m dan C

BUKTI

Paling sedikit terdapat satu distribusi-m minimal dari C untuk setiap m dan C. Contoh $(C, \emptyset, \ldots, \emptyset)$, jadi suatu distribusi-m minimal dari C pasti ada. ■

LEMMA 2

Diberikan $(a_1,a_2,\ldots,a_m)$ sebagai distribusi-m minimal dari C. Maka distribusi-m minimal dari C $+w_m$ adalah $(a_1,a_2,\ldots,a_m+1)$ atau berbentuk $(b_1,b_2,\ldots,b_{m-1},\emptyset)$.

BUKTI

Dengan Lemma 1, C + w_m paling sedikit memiliki satu distribusi, misalnya $(b_1,b_2, \ldots ,b_m)$ yaitu

$$\sum_{i=1}^{m} b_i w_i = C + w_m \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

Andaikan $(a_1,a_2, \ldots ,a_m+1)$ bukan distribusi-m minimal dari C + w_m sehingga

$$\sum_{i=1}^{m} b_i < 1 + \sum_{i=1}^{m} a_i \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

Dari (1) maka

$$b_1w_1+ \ ,\ldots, \ + b_{m-1}w_{m-1} + b_mw_m - w_m = C$$

$$b_1m_1+ \ ,\ldots, \ + b_{m-1}w_{m-1} + (b_m-1)w_m = C \ \ldots\ldots \quad (3)$$

Andaikan bahwa $b_m > \emptyset$, persamaan (3) menyatakan bahwa $(b_1, \ldots , b_{m-1},b_m-1)$ adalah suatu distribusi minimal dari C. Dari persamaan (2)

$$b_1+\ldots+ b_{m-1}+ (b_m-1) < \sum_{i=1}^{m} a_i$$

menerangkan bahwa $(a_1, a_2, \ldots, a_m)$ adalah bukan distribusi minimal-m dari C, maka kontradiksi. Oleh karena itu salah satu dari kedua pengandaian tersebut pasti salah. Maka distribusi minimal dari $C + w_m$ adalah salah satu dari $(a_1, a_2, \ldots, a_m + 1)$ atau $(b_1, b_2, \ldots, b_{m-1}, \emptyset)$. ■

THEOREMA 4

Diberikan $(a_1, a_2, \ldots, a_m)$ sebagai suatu distribusi-m minimal dari C dan $(b_1, b_2, \ldots, b_{m-1})$ distribusi-(m-1) minimal dari $C + w_m$. Maka distribusi-m minimal dari $C + w_m$ adalah

$(a_1, a_2, \ldots, a_{m-1}, a_m + 1)$ jika $\sum_{i=1}^{m} a_i < \sum_{i=1}^{m-1} b_i$

$(b_1, b_2, \ldots, b_{m-1}, \emptyset)$ jika tidak

BUKTI

Andaikan

$$\sum_{i=1}^{m} a_i < \sum_{i=1}^{m-1} b_i \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

dan $(a_1, a_2, \ldots, a_m + 1)$ adalah bukan distribusi-m minimal dari $C + w_m$. Maka dengan Lemma 2 setiap distribusi-m minimal dari $C + w_m$ harus mempunyai elemen $\emptyset$ di ujung paling kanan. Oleh karena itu jika $(b_1, b_2, \ldots, b_{m-1})$ adalah distribusi-(m-1) minimal dari $C + w_m$, $(b_1, b_2, \ldots, b_{m-1}, \emptyset)$ adalah distribusi-m minimal dari $C + w_m$

$$\sum_{i=1}^{m-1} b_i < 1 + \sum_{i=1}^{m} a_i \qquad \text{atau}$$

$$\sum_{i=1}^{m-1} b_i \leq \sum_{i=1}^{m} a_i$$

ternyata kontradiksi dengan (4), oleh karena itu jika (4) dipenuhi $(a_1, a_2, \ldots, a_{m-1}, a_m + 1)$ pasti merupakan distribusi-m minimal dari $C + w_m$.

Jika $(a_1, a_2, \ldots, a_{m-1}, a_m + 1)$ bukan distribusi-m minimal dari $C + w_m$ maka sesuai dengan Lemma 2 distribusinya pasti $(b_1, b_2, \ldots, b_{m-1}, \emptyset)$ seperti ditunjukkan di atas. ■

THEOREMA 5

Diberikan $(a_1, a_2, \ldots, a_m)$ sebagai distribusi-m minimal dari C dan diberikan

$$\left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_i \right) - C \right) \Big/ \left(w_m - w_{m-1} \right) \right] \leq \emptyset \quad \ldots\ldots(5)$$

maka untuk setiap bilangan bulat $j \geq \emptyset$, suatu distribusi-m minimal dari $C + jw_m$ adalah

$$(a_1, a_2, \ldots, a_{m-1}, a_m + j)$$

BUKTI

Dengan induksi pada j

Basis untuk $j = \emptyset$, Theorema tersebut jelas benar.

Langkah berikutnya anggap terdapat distribusi-m minimal dari $C + (j+1)w_m$ untuk $j = j + 1$ dengan bentuk $(b_1, b_2, \ldots, b_{m-1}, \emptyset)$ oleh karena itu

$$C + (j+1)w_m = \sum_{i=1}^{m-1} b_i w_i \leq w_{m-1} \sum_{i=1}^{m-1} b_i \quad \ldots\ldots\ldots(6)$$

Dengan pengujian induktif $(a_1, a_2, \ldots, a_{m-1}, a_m + j)$ adalah suatu distribusi-m minimal dari $C + jw_m$ dan oleh karena itu

$$C + jw_m = a_1 w_1 + \ldots + a_{m-1} w_{m-1} + (a_m + j)w_m$$

atau

$$C + (j + 1)w_m = a_1 w_1 + \ldots + a_{m-1} w_{m-1} + (a_m + j + 1)w_m$$

oleh karena itu $(a_1, \ldots, a_{m-1}, a_m + j + 1)$ adalah suatu distribusi-m minimal dari $C + (j + 1)w_m$, jadi

$$\sum_{i=1}^{m-1} b_i \leq j + 1 + \sum_{i=1}^{m} a_i \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(7)$$

kombinasi (6) dan (7) menghasilkan

$$C + (j + 1)w_m \leq w_{m-1}\left(j + 1 + \sum_{i=1}^{m} a_i \right)$$

$$(j + 1)w_m - w_{m-1}(j + 1) \leq w_{m-1} \sum_{i=1}^{m} a_i - C$$

$$(j + 1)(w_m - w_{m-1}) \leq w_{m-1} \sum_{i=1}^{m} a_i - C$$

$$\left[\left(w_{m-1} \sum_{i=1}^{m} a_i \right) - C \right] \Big/ (w_m - w_{m-1}) \geq j + 1 \geq 1$$

ternyata kontradiksi dengan (5). Oleh karena itu, distribusi-m minimal dari $C + (j + 1)w_m$ tidak dapat sesuai dengan bentuk $(b_1, b_2, \ldots, b_{m-1}, \emptyset)$. Dengan Lemma 2 dan induksi maka distribusinya pasti

$$(a_1, a_2, \ldots, a_{m-1}, a_m + j + 1)$$ ■

LEMMA 3

Diberikan $(a_{j1}, a_{j2}, \ldots, a_{jm})$ sebagai distribusi-m minimal dari $C_j = C + jw_m$ $(j = \emptyset, 1, 2, \ldots)$ dan diberikan

$$h_j = \left[\left[\left(w_{m-1} \sum_{i=1}^{m} a_{ji} \right) - C_j \right] \Big/ (w_m - w_{m-1}) \right]$$

maka h_j adalah fungsi menurun dengan cepat dari j.

BUKTI

$(a_{j1}, a_{j2}, \ldots, a_{j(m-1)}, a_{jm} + 1)$ adalah suatu distribusi-m dari $C + (j+1)w_m$ maka pasti dipenuhi

$$\sum_{i=1}^{m} a_{(j+1)i} \leq 1 + \sum_{i=1}^{m} a_{ji} \quad \text{atau} \quad \sum_{i=1}^{m} (a_{ji} - a_{(j+1)i}) \geq -1$$

jadi

$$h_j - h_{j+1} = \left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_{ji} \right) - C_j \right) \Big/ (w_m - w_{m-1}) \right]$$

$$- \left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_{(j+1)i}\right) - C_{j+1}\right) \Big/ \left(w_m - w_{m-1}\right)\right]$$

$$> \left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_i\right) - C_j\right) \Big/ \left(w_m - w_{m-1}\right)\right]$$

$$- \left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_{(j+1)i}\right) - C_{j+1}\right) \Big/ \left(w_m - w_{m-1}\right)\right] - 1$$

$$= \left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_{ji} - a_{(j+1)i}\right) - \left(C_{j+1} - C_j\right)\right) \Big/ \left(w_m - w_{m-1}\right)\right] - 1$$

$$\geq \left[\left(w_{m-1}(-1) + w_m\right) / \left(w_m - w_{m-1}\right)\right] - 1 = \emptyset$$

Oleh karena itu $h_{j+1} < h_j$ ■

Algoritma penyelesaian masalah making change dengan MINDIST sesungguhnya merupakan suatu sub program MINDIST (C′,m) yang bisa diakses dari program utama dan di dalam sub program itu sendiri. C′ dan m adalah parameter formal yang nilainya ditentukan oleh C dan n pada parameter aktual yang didefinisikan dalam program utama. Secara sistematis langkah-langkah algoritma MINDIST (C′,m) dapat disusun sebagai berikut.

step 1

Dimulai dengan pengecekan apakah m = 1 ? Jika salah maka ke step 2 dan jika benar maka algoritma selesai. Disribusi minimal dari C′ adalah

MINDIST = (C′)

dan prosedur keluar melalui Exit 1 kembali ke pemanggil.

step 2

Menentukan besarnya nilai q dan r dengan formula

$q = C' / w_m$

$r = C' - (C / w_m)$

step 3

Melakukan cek apakah m = 2 ?. Jika benar ke step 4, dan jika salah maka maka algoritma selesai. Disribusi minimal dari C' adalah

MINDIST = (r,q)

dan prosedur keluar melalui Exit 2 kembali ke pemanggil.

step 4

Nilai j diset sama dengan nol (j = Ø).

step 5

Menentukan distribusi (b_{j1},b_{j2}, ... ,$b_{j(m-1)}$) yang nilainya tergantung pada hasil perhitungan rekursif MINDIST ($r+jw_m$,m-1). Artinya prosedur MINDIST (C',m) dilaksanakan dalam prosedur ini dengan C'= $r+jw_m$ dan m = m-1.

step 6

Tes kondisi apakah j = Ø ?. Jika benar ke step 8 dan jika salah ke step 7.

step 7

Tes kondisi apakah

$$\sum_{i=1}^{m} a_{(j-1)i} < \sum_{i=1}^{m-1} b_i \quad ?$$

Jika tidak ke step 8, dan jika benar maka

$$(a_{j1},\ldots,a_{jm}) = (a_{(j-1)1},\ldots,a_{(j-1)(m-1)},a_{(j-1)m}+1)$$

kemudian ke step 9

step 8

Diperoleh

$$(a_{j1},\ldots,a_{jm}) = (b_{j1},a_{j2}, \ldots ,b_{j(m-1)}, Ø)$$

step 9

Cek apakah j = q ? Jika salah maka ke step 1Ø, dan jika benar maka maka algoritma selesai. Distribusi minimal dari C′ adalah

$$\text{MINDIST} = (a_{j1},a_{j2}, \ldots ,a_{j(m-1)},a_{jm})$$

dan prosedur keluar melalui Exit 3 kembali ke program utama.

step 1Ø

Melakukan tes kondisi apakah

$$h_j = \left[\left[\left(w_{m-1} \sum_{i=1}^{m} a_i \right) - r+jw_m \right] \Big/ (w_m - w_{m-1}) \right] \leq \emptyset \quad ?$$

Jika jawabannya tidak, maka nilai j ditambah 1 atau j = j+1, kemudian kembali ke step 5.

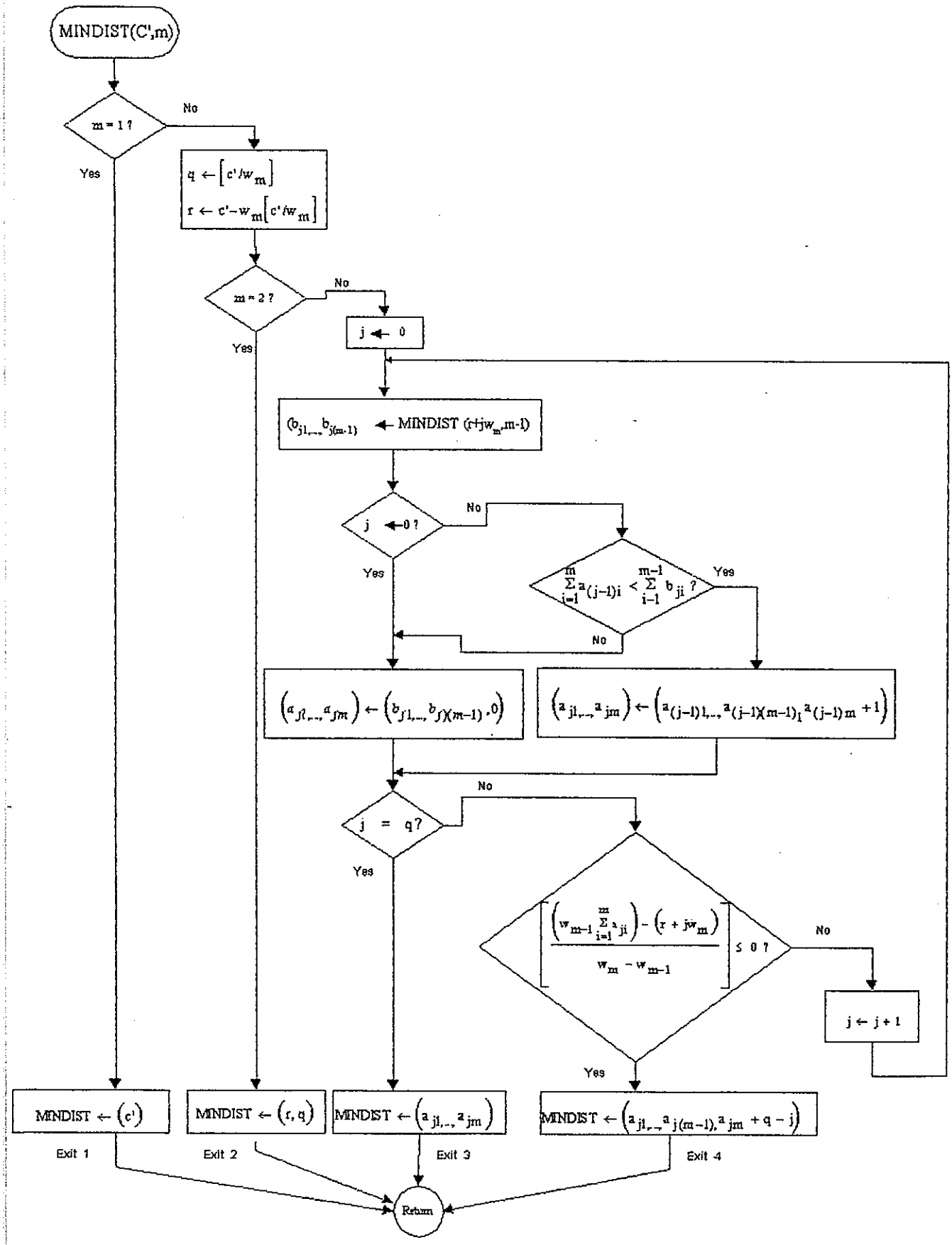
Jika kondisinya benar maka algoritma selesai. Disribusi minimal dari C´ adalah

MINDIST = $(a_{j1}, a_{j2}, \ldots, a_{j(m-1)}, a_{jm} + q - j)$

dan prosedur keluar melalui Exit 4 kembali ke pemanggil.

Algoritma dari sub program MINDIST (C´,m) juga disajikan dalam bentuk flowchart seperti ditunjukkan pada bagan berikut ini.

Flowchart MINDIST (C',m)



Jika m = 1 maka MINDIST = C′ dan prosedur akan keluar kembali melalui EXIT 1.

Jika m > 1, q dan r dihitung sedemikian sehingga memenuhi persamaan $C' = qw_m + r$ ($\emptyset \leq r < w_m$). Jika m = 2, MINDIST = (r,q) sesuai dengan Theorema 2 dan kembali melalui EXIT 2.

Jika m > 2 distribusi-(m-1) minimal $(b_{j1}, b_{j2}, \ldots, b_{j(m-1)})$ dari $r + jw_m$ ($j = \emptyset, 1, 2, \ldots$) dihitung dengan melakukan rekursi MINDIST (C′,m). Kemudian distribusi-m minimal $(a_{j1}, a_{j2}, \ldots, a_{jm})$ dari $r + jw_m$ dihitung dimulai j = Ø,

$(a_{o1}, a_{o2}, \ldots, a_{om}) = (b_{o1}, b_{o2}, \ldots, b_{o(m-1)}, \emptyset)$

(Lihat Theorema 3).

Sedangkan untuk $j > \emptyset$, $(a_{j1}, \ldots, a_{jm})$ adalah

$* \ (a_{(j-1)1}, \ldots, a_{(j-1)(m-1)}, a_{(j-1)m}+1)$ jika $\sum_{i=1}^{m} a_{(j-1)i} < \sum_{i=1}^{m-1} b_i$

$* \ (b_{j1}, \ldots, b_{j(m-1)}, \emptyset)$ jika tidak

(Lihat Theorema 4)

Proses ini terus dilakukan hingga j = q, pada kondisi tersebut $r + jw_m = r + qw_m = C'$ dan $(a_{j1}, a_{j2}, \ldots, a_{jm})$ adalah distribusi-m minimal dari C′. Pada kasus ini MINDIST = $(a_{j1}, a_{j2}, \ldots, a_{jm})$ dan prosedur kembali melalui EXIT 3.

Jika j = q belum terpenuhi akan dilakukan test apakah

$$h_j = \left[\left(\left(w_{m-1} \sum_{i=1}^{m} a_i \right) - r + jw_m \right) \Big/ \left(w_m - w_{m-1} \right) \right] \leq 0$$

Jika jawabannya tidak maka j ditambah satu dan proses dilanjutkan (sesuai flowchart). Tetapi jika jawabannya benar maka distribusi minimalnya adalah

$$(a_{j1}, a_{j2}, \ldots, a_{j(m-1)}, a_{jm}+q-j)$$

(lihat Theorema 5)

Pada kasus ini MINDIST = $(a_{j1}, a_{j2}, \ldots, a_{j(m-1)}, a_{jm}+q-j)$ dan prosedur kembali lewat EXIT 4.

Berikut ini akan diberikan tabel penyelesaian masalah making change dengan menggunakan algoritma prosedur MINDIST(C′,m). Dalam contoh ini diberikan W = {1,4,5,6,7}, m = 5, C′ = 352.

Selanjutnya untuk menghitung distribusi-m minimal $(b_{j1}, \ldots, b_{j(m-1)})$ dari $r + jw_m$, mengarah pada perhitungan $C' = r + jw_m$ pada sub tabel m - 1. Contohnya pada sub tabel 5, (0,1,0,2) adalah distribusi-4 minimal dari 16. Untuk menghitung distribusi tersebut dijelaskan pada sub tabel 4 dengan C′=16. Akhirnya (0,0,2) diberikan sebagai distribusi-3 minimal dari 10, hasil ini diperoleh dari sub tabel 3 dengan C′=10. Demikian seterusnya hingga proses rekursi selesai dan dapat ditemukan penyelesaian dari masalah tersebut.

M	C´	q	r	j	r+ jw_m	(b_{j1},... $b_{j(m-1)}$	(a_{j1},... a_{jm}	h_j	EXIT	MINDIST
	352	50	2	0	2	(2,0,0,0)	(2,0,0,0,0)	10		
				1	9	(0,1,1,0)	(0,1,1,0,0)	3		
5				2	16	(0,1,0,2)	(0,1,1,0,1)	2		
				3	23	(0,0,1,3)	(0,1,1,0,2)	1		
				4	30	(0,0,0,5)	(0,1,1,0,3)	0	4	(0,1,1,0,49)
	2	0	2	0	2	(2,0,0)	(2,0,0,0)		3	(2,0,0,0)
	9	1	3	0	3	(3,0,0)	(3,0,0,0)	12		
				1	9	(0,1,1)	(0,1,1,0)		3	(0,1,1,0)
4	16	2	4	0	4	(0,1,0)	(0,1,0,0)	1		
				1	10	(0,0,2)	(0,1,0,1)	0	4	(0,1,0,2)
	23	3	5	0	5	(0,0,1)	(0,0,1,0)	0	4	(0,0,1,3)
	30	5	0	0	0	(0,0,0)	(0,0,0,0)	0	4	(0 0,0,5)
	2	0	2	0	2	(2,0)	(2,0,0)		3	(2,0,0)
	3	0	3	0	3	(3,0)	(3,0,0)		3	(3,0,0)
	9	1	4	0	4	(0 1)	(0,1,0)	0	4	(0,1,1)
3	4	0	4	0	4	(0,1)	(0,1,0)		3	(0,1,0)
	10	2	0	0	0	(0,0)	(0,0,0)	0	4	(0,0,2)
	5	1	0	0	0	(0,0)	(0,0,0)	0	4	(0,0,1)
	0	0	0	0	0	(0,0)	(0,0,0)		3	(0,0,0)
	2	0	2						2	(2,0)
2	3	0	3						2	(3,0)
	4	1	0						2	(0,1)
	0	0	0						2	(0,0)

Tabel MINDIST (C´,m)