PENGGUNAAN ALGORITME *COLUMN GENERATION* UNTUK MENENTUKAN RUTE OPTIMAL PENGANGKUTAN SAMPAH DI KOTA BOGOR

FARIDA HANUM

RINGKASAN. Pengelolaan sampah di kota besar pada saat ini telah menimbulkan persoalan lingkungan yang memerlukan penanganan segera. Jika tidak dikelola dengan baik, maka sampah akan menjadi sumber penyakit, menimbulkan bau busuk, mencemari air tanah, menimbulkan banjir, dan dapat pula mengurangi keindahan lingkungan. Ada beberapa tahapan pengelolaan sampah yang selama ini dilakukan di beberapa kota di Indonesia yaitu (i) mengumpulkan sampah dari lokasi penghasil sampah (sumber sampah) kemudian diangkut ke TPS (Tempat Penampungan Sementara) sampah, (ii) mengangkut sampah dari TPS-TPS ke TPA (Tempat Pembuangan Akhir) sampah, dan (iii) memusnahkan sampah dengan lahan urug saniter, membakarnya di dalam *incenerator*, atau mengelola sampah menjadi barang yang dapat dipakai lagi, misalkan kompos, barang-barang keperluan rumah tangga dll. Di kota Bogor, biaya pengangkutan sampah ternyata lebih tinggi dibandingkan dengan biaya komponen lain dalam pengelolaan sampah. Pada penelitian ini, dilakukan pemodelan pengangkutan sampah di kota Bogor dengan *capacitated node routing problem* dan penentuan solusi optimalnya dengan menggunakan algoritme *column generation*.

1. PENDAHULUAN

Sampah adalah barang yang dibuang karena tidak terpakai lagi. Jumlah timbunannya meningkat dari tahun ke tahun sejalan dengan meningkatnya kegiatan dan jumlah penduduk. Pengelolaan sampah di kota-kota besar pada saat ini telah menimbulkan persoalan lingkungan yang memerlukan penanganan segera. Jika tidak dikelola dengan baik, maka sampah akan menjadi sumber penyakit, menimbulkan bau busuk, mencemari air tanah, menimbulkan banjir, dan mengurangi keindahan lingkungan. Ada beberapa tahapan pengelolaan sampah yang selama ini dilakukan di beberapa kota di Indonesia yaitu (i) mengumpulkan sampah dari lokasi penghasil sampah (sumber sampah) kemudian diangkut ke TPS (Tempat Penampungan Sementara) sampah, (ii) mengangkut sampah dari TPS-TPS ke TPA (Tempat Pembuangan Akhir) sampah, dan (iii) memusnahkan sampah dengan lahan urug saniter atau mengelola sampah menjadi barang yang dapat dipakai lagi, misalkan kompos, barang-barang keperluan rumah tangga dll.

Berdasarkan Perda No 19 Tahun 2002 tentang Organisasi Perangkat Daerah Kota Bogor, pelaksana teknis bidang kebersihan di Bogor ditangani oleh Dinas Kebersihan dan Pertamanan (DKP) Kota Bogor. DKP harus menangani sampah dari ±380 TPS dan *transfer depo* di Kota Bogor dan mengangkutnya ke TPA Galuga Leuwiliang di Kabupaten Bogor yang berjarak ±25 km dari pusat kota Bogor. Daerah layanan kebersihan di Kota Bogor yang mempunyai luas ±11.850 ha ini dibagi menjadi tiga kategori

(1) daerah primer/protokol/komersial dengan timbulan sampah tinggi mendapat layanan 3 kali penyapuan dan 2 kali pengangkutan dalam satu hari,
(2) daerah sekunder/protokol/komersial dengan timbulan sampah sedang mendapat layanan 2 kali penyapuan dan 1 kali pengangkutan dalam satu hari,
(3) daerah tersier dengan timbulan sampah sedang/kurang mendapat layanan 1 kali penyapuan dalam satu hari dan 1 kali pengangkutan dalam dua hari atau lebih.

Setelah dikumpulkan, sampah diangkut menuju TPA Galuga untuk dimusnahkan dengan sistem *control landfill* atau *composting.*

Alat yang dimiliki Dinas Kebersihan dan Pertamanan Kota Bogor untuk menangani masalah kebersihan adalah sebagai berikut [Wismanto, 2004]:

(1) armada pengangkutan, yaitu:
    (a) *dump truck* (kapasitas 8 m3) sebanyak 50 unit,
    (b) *arm roll truck* (kapasitas 6 m3) sebanyak 17 unit,
    (c) truk bak kayu sebanyak 1 unit,
    (d) *Kijang* operasional sebanyak 4 unit,
    (e) *container* sebanyak 98 unit.
(2) alat berat, yaitu
    (a) buldozer sebanyak 2 unit,
    (b) *track loader* sebanyak 1 unit,
    (c) *whell loader* sebanyak 1 unit,
    (d) eksavator sebanyak 1 unit,
(3) petugas operasional kebersihan (di luar kepala seksi atau staf administrasi)
    (a) petugas penyapuan sebanyak 280 orang,
    (b) petugas angkutan sampah sebanyak 264 orang,
    (c) petugas TPA sebanyak 32 orang.

Pada saat ini, volume timbulan sampah yang dapat dilayani pengangkutan sampahnya baru sekitar 67,65 % dari 2.124 m3 sampah per hari [Wismanto, 2004]. Jika dilihat dari asal atau sumber sampah maka volume timbulan dan keterangkutan sampah di setiap sumber sampah dapat dilihat pada Tabel 1 berikut:

Tabel 1. Volume timbulan dan keterangkutan sampah berdasarkan sumber sampah

No.	Sumber Sampah	Timbulan sampah (m^3/hari)	Terangkut (m^3/hari)
1.	Pemukiman	1340	770
2.	Pasar	262	243
3.	Pertokoan, restoran, dan hotel	149	125
4.	Fasilitas umum dan sosial	95	76
5.	Sapuan jalan	159	139
6.	Kawasan industri	99	84
T o t a l		2.124	1.437

Terlihat bahwa dengan sumber daya yang dimiliki saat ini ternyata pengelolaan sampah di Kota Bogor belum dapat dilakukan secara optimal.

Selain itu, jika ditinjau dari aspek biaya, ternyata biaya pengangkutan sampah merupakan bagian terbesar dari biaya pengelolaan sampah secara keseluruhan seperti yang terlihat pada Tabel 2 berikut ini.

Tabel 2. Persentase dan anggaran biaya sistem pengelolaan sampah di kota Bogor
[Wismanto, 2004]

Sistem Pengelolaan	Persentase	Anggaran 2003 (Rupiah)
Pengumpulan	6	413.037.140
Pengangkutan	70	4.818.766.635
Incinerator	5	344.197.617
TPA	19	1.307.950.944
T o t a l		**6.883.952.336**

Banyak faktor yang mungkin dapat menjadi penyebab tingginya biaya pengangkutan ini, antara lain banyaknya kendaraan pengangkut, banyaknya petugas pengambil sampah, banyaknya sampah yang harus diangkut, kondisi jalan, kondisi lalu lintas, luas daerah pelayanan, jarak TPA dengan daerah layanan, rute pengangkutan, dan lain-lain. Minimisasi biaya pada setiap faktor pendukung berarti juga minimisasi biaya sistem pengangkutan itu sendiri. Dari uraian di atas terlihat bahwa peningkatan efisiensi, efektivitas, serta produktivitas yang terkait dengan sistem pengangkutan sampah merupakan salah satu kunci utama dalam mengurangi total biaya sistem pengelolaan sampah. Oleh karena itu, penelitian mengenai optimasi salah satu faktor pendukung sistem pengangkutan sampah di kota Bogor ini perlu dilakukan.

Masalah pengangkutan sampah dapat dipandang sebagai masalah penentuan rute dengan biaya/jarak minimum yang memenuhi kendala-kendala seperti banyaknya kendaraan dan kapasitas kendaraan, serta waktu kerja petugas Dinas Kebersihan. Dalam tulisan ini, masalah pengangkutan sampah tersebut akan dimodelkan dalam salah satu varian dari *Vehicle Routing Problem* (VRP) .

2. *Vehicle Routing Problem* (VRP)

Vehicle Routing Problem (VRP) adalah nama generik untuk masalah yang berkaitan dengan penentuan rute untuk dapat mengunjungi semua pelanggan (*customer*) oleh sejumlah kendaraan (*vehicle*). VRP juga dikenal dengan nama *vehicle scheduling* (Clarke & Wright, 1964), *vehicle dispatching,* atau masalah "pengiriman" (*delivery*) saja (Christofides *et al.* , 1979). Banyak dijumpai terapan VRP dalam masalah sehari-hari, misalkan pengambilan surat dari kotak-kotak pos yang tersebar di seluruh kota, pengantaran dan penjemputan anak sekolah dengan bis sekolah, pengiriman cucian baju (*laundry*), dan lain-lain.

Secara matematis, VRP dapat dinyatakan melalui suatu graf $G = (N, A)$, dengan $N = \{0, 1, \ldots, n\}$ menyatakan himpunan *node* yang menyatakan lokasi, dan $A = \{(i, j) \,|\, i, j \in N, i \neq j\}$ merupakan himpunan sisi yang menyatakan jalan yang menghubungkan lokasi. *Node* 0 menyatakan *depot,* yaitu tempat menyimpan kendaraan-kendaraan yang digunakan sehingga rute kendaraan dimulai dan berakhir di *depot.* Berbagai jenis kendala dapat menghasilkan banyak varian dari VRP, antara lain:

(1) kendaraan dapat mempunyai kapasitas yang sama atau berbeda. Jika kapasitas semua kendaraan adalah sama, misalkan C, maka VRP dinamakan dengan *Capacitated VRP* (CVRP).
(2) jika diperlukan selang waktu di setiap *node,* maka masalahnya menjadi VRPTW (*Vehicle Routing Problem with Time Window*),
(3) jika pelanggan terbagi menjadi dua bagian, yaitu pelanggan yang harus diambil barangnya (*pickup*) dan pelanggan yang harus diantarkan barangnya (*delivery*), maka dikenal masalah *Pickup and Delivery Problem* (PDP). Masalah PDP juga mempunyai banyak varian antara lain PDP dengan *time windows* (PDPTW).

VRP dapat dimodelkan dalam bentuk pemrograman dinamik ataupun *Integer Linear programming* (ILP). Pada tulisan ini pemodelan masalah VRP yang dibahas hanyalah dalam bentuk ILP karena model ini yang banyak digunakan dan lebih sederhana [Kulkarni & Bhave, 1985].

Misalkan

n = banyaknya *node*/lokasi,

V = banyaknya kendaraan,

C_k = kapasitas kendaraan ke- k, dengan $k = 1, 2, \ldots, V$

T_k = maksimum ogkos/waktu yang diperbolehkan untuk suatu rute yang menggunakan kendaraan ke-k,

Q_i = kuantitas pada *node*/lokasi ke-i,

y_i = bilangan real sembarang,

c_{ij} = biaya perjalanan dari *node* i ke *node* j,

$$x_{ijk} = \begin{cases} 1, & \text{jika sisi } (i,j) \text{ berada pada rute kendaraan ke-}k, \\ 0, & \text{jika selainnya.} \end{cases}$$

Maka formulasi VRPnya (seperti dalam [Kulkarni & Bhave, 1985], [Christofides *et al.*, 1979]) adalah

$$\text{Minimumkan} \quad z = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{V} c_{ij}x_{ijk}, \tag{2.1}$$

terhadap

$$\sum_{i=1}^{n}\sum_{k=1}^{V} x_{ijk} = 1, \text{ untuk } j = 1, 2, \ldots, n-1 \tag{2.2}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{V} x_{ijk} = 1, \text{ untuk } i = 1, 2, \ldots, n-1 \tag{2.3}$$

$$\sum_{i=1}^{n} x_{ihk} - \sum_{j=1}^{n} x_{hjk} = 0, \text{ untuk } k = 1, \ldots, V \; ; \; h = 1, \ldots, n, \tag{2.4}$$

$$\sum_{i=1}^{n} Q_i \sum_{j=1}^{n} x_{ijk} \leq C_k, \text{ untuk } k = 1, 2, \ldots, V \, , \tag{2.5}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ijk} \leq T_k, \text{untuk } k = 1, 2, \ldots, V \tag{2.6}$$

$$\sum_{j=1}^{n-1} x_{Njk} \leq 1, \text{ untuk } k = 1, 2, \ldots, V \tag{2.7}$$

$$\sum_{i=1}^{n-1} x_{iNk} \leq 1, \text{ untuk } k = 1, 2, \ldots, V \tag{2.8}$$

$$x_{ijk} = 0 \text{ atau } 1, \forall i, j, k \tag{2.9}$$

$$y_i - y_j + n\, x_{ijk} \leq n - 1, \tag{2.10}$$

$$\text{untuk } 1 \leq i \neq j \leq n-1, 1 \leq k \leq V. \tag{2.11}$$

Pada formulasi di atas, kendala (2.2) dan (2.3) menjamin bahwa setiap pelanggan hanya dikunjungi oleh satu kendaraan. Kekontinuan rute dinyatakan dengan kendala (2.4), Kendala (2.5) menyatakan kendala kapasitas kendaraan, dan kendala (2.6) menyatakan batasan ongkos/waktu untuksetiap rute kendaraan. Kendala (2.7) dan (2.8)

menyatakan bahwa penentuan rute sesuai dengan ketersediaan kendaraan, dan akhirnya kendala (2.10) merupakan kendala *subtour elimination*, yaitu kendala yang menjamin tidak terdapat subrute (hanya sebagian pelanggan yang dilayani pada suatu rute). Karena kendala (2.2) dan (2.4) mengakibatkan berlakunya kendala (2.3) sedangkan kendala (2.4) dan (2.7) mengakibatkan berlakunya kendala (2.8), maka kendala (2.3) dan (2.8) merupakan kendala yang berlebih (*redundant*) sehingga dapat dibuang dari formulasi masalah tanpa mempengaruhi hasil akhir. Kendala eliminasi *subtour* dapat juga dituliskan dalam bentuk lain [Christofides *et al.*, 1979]

$$y_i - y_j + n\sum_{k=1}^{V} x_{ijk} \leq n-1, \text{ untuk } 1 \leq i \neq j \leq n-1.$$

Dalam pembahasan ini diasumsikan bahwa untuk $1 \leq i \leq n$ dan $1 \leq k \leq V$ berlaku

$$Q_i \leq \max(C_k),$$

yaitu bahwa permintaan di setiap *node*/lokasi harus kurang dari atau paling banyak sama dengan kapasitas minimum dari kendaraan.

2.0.1. *Node Routing Problem dengan Kendala Kapasitas dan Panjang.* *Node Routing Problem* dengan kendala Kapasitas dan Panjang (NRP-KP) adalah menentukan sejumlah m rute kendaraan dengan ongkos terkecil yang berawal dan diakhiri di *depot* sedemikian rupa sehingga

- setiap pelanggan dikunjungi tepat satu kali
- setiap pelanggan i dipadankan dengan *demand* p_i (*demand* boleh diambil atau diantarkan, tapi tidak keduanya); dan selanjutnya total *demand* yang dilayani setiap kendaraan tidak boleh melebihi kapasitas kendaraan q (kendaraan diasumsikan identik); kendala ini disebut **kendala kapasitas,**
- setiap pelanggan dipadankan dengan waktu pelayanan s_i, maka total durasi setiap rute, termasuk waktu pelayanan pelanggan dan waktu perjalanan, tidak boleh melebihi waktu kerja T.

Masalah ini dapat diformulasikan melalui graf berarah $G = (V, A)$ atau graf takberarah $G = (V, E)$ begantung pada matriks biayanya simetrik atau taksimetrik. Dalam kedua kasus tersebut, himpunan *node* (simpul) $V = \{0\} \cup U$, dengan 0 menyatakan *depot*, dan U adalah himpunan pelanggan.

Misalkan

c_{ij} adalah biaya/jarak perjalanan dari *node* i ke *node* j, dan

$$x_{ij} = \begin{cases} 1, & \text{jika sisi } (i,j) \in A \text{ merupakan bagian dari solusi} \\ 0, & \text{jika selainnya} \end{cases}$$

Jika NRPKP diformulasikan dalam model TSP (*Traveling Salesman Problem*) yang simetrik, akan diperoleh

$$\text{minimumkan} \sum_{(i,j)\in E} c_{ij}x_{ij}$$

terhadap

$$\sum_{\substack{i\in V,\\(i,j)\in E}} x_{ij} + \sum_{\substack{i\in V,\\(i,j)\in E}} x_{ji} = 2,\ j \in U, \tag{2.12}$$

$$\sum_{\substack{i\in V,\\(0,i)\in E}} x_{0i} = 2m, \tag{2.13}$$

$$\sum_{\substack{(i,j)\in E,\\i\in S, j\notin S}} x_{ij} + \sum_{\substack{(j,i)\in E,\\i\in S, j\notin S}} x_{ji} \geq 2\alpha(S),\ S \subset V\setminus\{0\}, |S| \geq 2, \tag{2.14}$$

$$\sum_{\substack{(i,j)\in E,\\i\in S, j\notin S}} x_{ij} + \sum_{\substack{(j,i)\in E,\\i\in S, j\notin S}} x_{ji} \geq 4,\ S \subset V\setminus\{0\}, |S| \geq 2, t^*_{STSP} > T \tag{2.15}$$

$$x_{ij} \in \{0,1\},\ (i,j) \in E$$

Kendala (2.12) menyatakan bahwa dua sisi *incident* dengan pelanggan $j \in U$ (**kendala derajat pelanggan**). Dengan cara serupa, kendala (2.13) menjamin bahwa $2m$ sisi *incident* dengan *depot* (**kendala derajat depot**). Kendala (2.14) merupakan **kendala kapasitas** yang menyatakan bahwa banyaknya kendaraan yang melayani pelanggan di S paling sedikit adalah $\alpha(S)$. Pada prakteknya

$$\alpha(S) = \left\lceil \frac{\sum_{i\in S} p_i}{q} \right\rceil.$$

Kendala (2.15) merupakan **kendala panjang,** yang menyatakan bahwa suatu rute tunggal tidak dapat melayani semua pelanggan di S jika waktu terkecil dari *cycle* Hamilton (yang melewati semua *node* tepat satu kali) pada masalah STSP (*symmetric travelling salesman problem*) t^*_{STSP} melebihi batasan waktu T yang diberikan.

Formulasi *Set Partitioning*. Formulasi alternatif dari masalah NRPKP adalah diberikan berikut ini. Misalkan K menyatakan himpunan rute di G yang memenuhi kendala kapasitas dan kendala panjang, dan misalkan $c_k, k \in K$ menyatakan biaya/jarak untuk rute ke-k. Didefinisikan untuk setiap $i \in V$, dan $k \in K$

$$a_{ik} = \begin{cases} 1, & \text{jika } \textit{node } i \text{ berada pada rute } k \\ 0, & \text{jika selainnya} \end{cases}$$

Misalkan untuk setiap $k \in K$ didefinisikan

$$y_k = \begin{cases} 1, & \text{jika rute } k \text{ digunakan pada solusi optimal} \\ 0, & \text{jika selainnya} \end{cases}$$

Maka NRPKP dapat diformulasikan menjadi

$$\text{minimumkan} \sum_{k \in K} c_k y_k$$

terhadap

$$\sum_{k \in K} a_{ik} y_k = 1, \quad i \in V \tag{2.16}$$

$$\sum_{k \in K} y_k = m, \tag{2.17}$$

$$y_k \in \{0,1\}, \; k \in K$$

Kendala (2.16) menyatakan bahwa setiap pelanggan $i \in V$ harus terlayani, sedangkan kendala (2.17) menyatakan bahwa tepat m kendaraan yang digunakan.
Beberapa algoritme Heuristik untuk NRPKP.

(1) "*Cluster first, route second*" yang menyelesaikan masalah NRPKP dalam 2 langkah,. yaitu
   (a) memartisi himpunan pelanggan V menjadi subset-subset $U_k \subset V \setminus \{0\}$, dengan $k = 1, 2, \ldots, m$ dan setiap U_k berpadanan dengan kendaraan $k = 1, 2, \ldots, m$.
   (b) untuk setiap kendaraan $k = 1, 2, \ldots, m$, masalah STSP diselesaikan (boleh secara eksak atau heuristik) pada sungraf yang diinduksi oleh $U_k \cup \{0\}$.

(2) "*Route first, cluster second*" menyelesaikan masalah dalam 2 tahap juga, yaitu
   (a) menentukan satu *cycle* Hamilton (sering tidak fisibel untuk NRPKP), dengan algoritme eksak atau algoritme heuristik,
   (b) *cycle* yang diperoleh didekomposisi menjadi m rute fisibel yang berawal dan berakhir di *depot*.

2.0.2. *Capacitated Arc Routing Problem.* Masalah pencarian rute pada sisi yang berkapasitas banyak diaplikasikan dalam kehidupan sehari-hari, antara lain dalam masalah pengumpulan sampah, pengumpulan surat di kotak-kotak surat dll. [Amponsah & Salhi, 2004], [Ghiani *et al., 2003*].. Masalah ini juga dapat diformulasikan dalam bentuk ILP dan graf seperti berikut ini [Amponsah & Salhi, 2004].

Misalkan diberikan graf terhubungkan $G = (V, E \cup A)$, dengan V adalah himpunan simpul (*node*), E adalah himpunan sisi takberarah dengan $E \subseteq V \times V$, dan A adalah himpunan sisi berarah dan $A \subseteq V \times V$. Masalah Penentuan Rute pada Sisi (*Arc Routing Problem/ARP),* atau sering dikenal dengan Masalah *Rural Postman* (*Rural Postman problem/RPP*) adalah masalah menentukan suatu perjalanan dengan ongkos minimum pada suatu himpunan bagian $R \subseteq E \cup A$. Dalam *Capacitated Arc Routing Problem* (CARP) atau *Capacitated*

Rural Postman Problem. ditambahkan bahwa rute yang dicari harus memenuhi *demand* $q_{ij} \geq 0$ untuk setiap sisi (sisi berarah) (i,j) yang dilayani oleh satu dari sejumlah kendaraan dengan kapasitas W. Tujuan masalah ini adalah menentukan banyaknya *circuit* dengan ongkos terkecil yang melalui *depot* yang memenuhi setiap permintaan dan sesuai dengan kapasitas kendaraan.

Misalkan

c_{ij} adalah ongkos perjalanan untuk satu sisi (sisi berarah) $(i,j) \in E(A)$,

x_{ijk} adalah berapa kali sisi (sisi berarah) $(i,j) \in E \cup A$ dilewati oleh *trip* k,

$$y_{ij} = \begin{cases} 1, & \text{jika sisi } (i,j) \in R \text{ di}cover \text{ oleh } trip\ k \\ 0, & \text{jika selainnya} \end{cases}$$

M adalah konstanta positif yang lebih besar atau sama dengan jumlah dari kapasitas sisi (sisi berarah) pada $S \subseteq R$,

$V[S]$ adalh himpunan *node* yang *incident* dengan himpunan sisi berarah di S,

k menyatakan *trip*,

dan K adalah maksimum banyaknya *trip* yang diperbolehkan.

Masalah CARP dapat dimodelkan dengan formulasi sebagai berikut

$$\text{minimumkan} \sum_{(i,j)\in E} \sum_{k=1}^{K} c_{ij} x_{ijk} \tag{2.18}$$

terhadap

$$\sum_{p\in V} x_{pik} - \sum_{p\in V} x_{ipk} = 0, \forall i \in V, \quad k = 1,2,\ldots,K, \tag{2.19}$$

$$\sum_{k=1}^{K} y_{ijk} = 1, \forall (i,j) \in R, \tag{2.20}$$

$$x_{ijk} \geq y_{ijk}, \forall (i,j) \in R, \quad k = 1,2,\ldots,K, \tag{2.21}$$

$$\sum_{(i,j)\in R} q_{ij} y_{ijk} \leq W, \quad k = 1,2,\ldots,K, \tag{2.22}$$

$$M \sum_{\substack{i\in V[S],\\ j\in V[S]}} x_{ijk} \geq \sum_{(j,p)\in S} x_{jpk} \begin{cases} \forall S \subseteq R, \\ 0 \in V[S], \\ k = 1,2,\ldots,K, \end{cases} \tag{2.23}$$

$$y_{ijk} \in \{0,1\}, .\forall (i,j) \in R, \quad k = 1,2,\ldots,K, \tag{2.24}$$

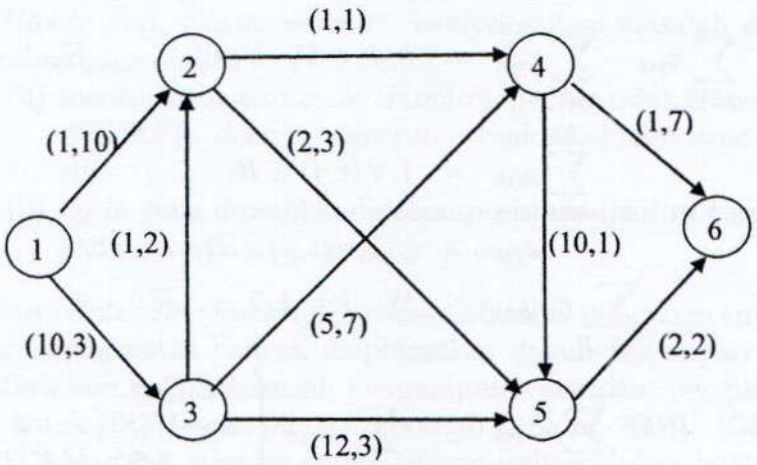
$$x_{ijk} \in Z^{+}, \forall (i,j) \in E, \quad k = 1,2,\ldots,K,$$

Fungsi objektif (2.18) menentukan ongkos minimum, persamaan (2.19) memastikan kekontinuan rute, persamaan (2.20) menyatakan bahwa setiap sisi dengan *demand* positif dilayani tepat satu kali. Pertaksamaan (2.21) menjamin bahwa *circuit* k meng*cover* sisi $(i,j) \in R$,

pertaksamaan (2.22) menyatakan batasan kapasitas kendaraan, dan pertaksamaan (2.23) merupakan kendala *eliminasi subtour*, dan kendala (2.24) merupakan kendala bilangan bulat.

2.1. **Algoritme *Column Generation*.** Masalah pencarian rute optimal (masalah *routing*) pada umumnya merupakan masalah kombinatorial yang melibatkan pencarian solusi terbaik di antara sejumlah besar solusi yang mungkin. Jika semua solusi fisibel diperiksa dan dibandingkan nilainya, akan memerlukan waktu yang seringkali tidak mungkin lagi dapat dikerjakan. Pada dekade terakhir ini telah dibuktikan bahwa algoritme *column generation* merupakan salah satu metode yang efektif untuk menentukan hampiran solusi optimal dari masalah *large-scale integer programming* (IP), yaitu *integer programming* yang melibatkan banyak variabel keputusan (Wilhelm, 2001).

Misalkan diberikan masalah untuk menentukan *path* terpendek terkendala (*constrained shortest path*). Misalkan diberikan suatu *network* $G = (V, A)$ seperti pada Gambar 2. Untuk setiap sisi $(i, j) \in A$, selain didefinisikan besaran biaya c_{ij} juga ditambahkan besaran t_{ij} yang menyatakan, misalkan waktu tempuh. Tujuannya adalah menentukan *path* terpendek dari *node* 1 ke *node* 6 sedemikian sehingga total waktu tempuh tidak melebihi angka tertentu, misalkan 14 satuan waktu.



GAMBAR 1

Misalkan

$$x_{ij} = \begin{cases} 1, & \text{jika ada } \textit{flow} \text{ dari } \textit{node } i \text{ langsung ke } \textit{node } j, \\ 0, & \text{jika selainnya.} \end{cases}$$

Salah satu cara menyatakan masalah *network flow* ini ke dalam model matematik adalah dengan menggunakan *Integer Programming* (IP)

menjadi masalah (SPT) sebagai berikut:

$$\text{minimumkan} \quad z = \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{2.25}$$

$$\text{terhadap} \quad \sum_{j:(1,j)\in A} x_{ij} = 1 \tag{2.26}$$

$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0, \quad i = 2,3,4,5 \tag{2.27}$$

$$\sum_{i:(i,6)\in A} x_{i6} = 1 \tag{2.28}$$

$$\sum_{(i,j)\in A} t_{ij}x_{ij} \leq 14 \tag{2.29}$$

$$x_{ij} = 0 \text{ atau } 1, \text{untuk } (i,j) \in A \tag{2.30}$$

Dengan cara biasa, terlihat bahwa terdapat 9 kemungkinan *path* dari *node* 1 ke *node* 6 yang fisibel, dan solusi optimalnya adalah *path* 1-3-2-4-6 dengan biaya 13 dan total waktu tempuh 13. Kendala waktu (2.29) tidak memungkinkan untuk menyelesaikan masalah SPT dengan menggunakan algoritme untuk masalah *path* terpendek yang klasik, seperti algoritme Dijkstra. Tentu saja jika banyaknya *node* cukup besar, maka cara manual seperti ini sangat tidak praktis untuk dilakukan. Karena masalah ini seperti masalah *path* terpendek biasa (kecuali ada kendala waktunya), maka masalah ini dicoba didekati dengan cara seperti masalah menentukan *path* terpendek dengan mereformulasi kembali masalah (SPT) di atas.

2.1.1. *Reformulasi yang Setara.* Jika kendala waktu (2.29) pada masalah SPT dihilangkan, maka daerah fisibel yang ada adalah

$$X = \{x_{ij} = 0 \text{ atau } 1 | x_{ij} \text{ memenuhi kendala (2.26) - (2.28)}\}.$$

Dari teori *network flow* diketahui bahwa titik ekstrim dari *polytope* yang didefinisikan dengan *convex hull* dari X, yaitu $\overline{x}_p = (x_{pij})$, berpadanan dengan suatu *path* $p \in P$ dalam *network* tersebut. Hal ini mengakibatkan bahwa setiap *flow* pada sisi *network* dapat dinyatakan sebagai kombinasi konveks dari *flow* pada *path*, atau:

$$x_{ij} = \sum_{p\in P} x_{pij}\lambda_p, \text{ untuk } (i,j) \in A, \tag{2.31}$$

$$\sum_{p\in P} \lambda_p = 1, \tag{2.32}$$

$$\lambda_p \geq 0, \text{ untuk } p \in P. \tag{2.33}$$

Dengan mensubstitusi $\overline{\mathbf{x}}$ pada persamaan (2.25) dan (2.29) pada masalah SPT akan diperoleh apa yang disebut dengan *master problem:*

$$\text{minimumkan } z = \sum_{p \in P} \left(\sum_{(i,j) \in A} c_{ij} x_{pij} \right) \lambda_p \tag{2.34}$$

$$\text{terhadap } \sum_{p \in P} \left(\sum_{(i,j) \in A} t_{ij} x_{pij} \right) \lambda_p \leq 14, \tag{2.35}$$

$$\sum_{p \in P} \lambda_p = 1,$$

$$\lambda_p \geq 0, \text{ untuk } p \in P$$

$$\sum_{p \in P} x_{pij} \lambda_p = x_{ij}, \text{ untuk } (i,j) \in A, \tag{2.36}$$

$$x_{ij} = 0 \text{ atau } 1, \text{untuk } (i,j) \in A \tag{2.37}$$

Pada formulasi di atas, penentuan x_{ij} yang membentuk *path* pada X terlihat pada kondisi "$p \in P$". Koefisien biaya λ_p pada fungsi objektif (2.34) adalah biaya untuk *path* p, dan koefisien dari λ_p di kendala (2.35) merupakan lama perjalanan *path* p. Kendala (2.36) dan (2.37) mempertahankan hubungan antara masalah yang asli dengan *master problem.*

2.1.2. *Relaksasi Linear dari Master Problem.* Untuk menyelesaikan masalah *Integer Programming,* terlebih dahulu diselesaikan relaksasi linear dari *master problem.* Jika solusi optimal yang diperoleh sudah berupa bilangan bulat semuanya, maka solusi optimal tersebut merupakan solusi optimal masalah *IP* yang asli. Jika kendala (2.37) direlakskan, sehingga hanya menjadi $x_{ij} \geq 0$ saja, maka tidak ada lagi hubungan antara variabel $\overline{\mathbf{x}}$ dan $\overline{\lambda}$ sehingga kendala (2.36) tidak diperlukan lagi. Jika solusi optimal dari relaksasi linear *master problem* bukan bilangan bulat, maka didefinisikan *restricted master problem* (RMP) yang bekerja hanya dengan sebagian variabel. Penambahan variabel (kolom) lainnya dilakukan bertahap dengan kriteria pemilihan tertentu sampai diperoleh solusi optimal yang *integral.*

2.1.3. *Metode Pembangkitan Kolom untuk Linear Programming.* Misalkan diberikan suatu *master problem* yang merupakan suatu *Linear Programming* berikut ini:

$$\begin{aligned} \text{minimumkan} \quad & z = \sum_{j \in J} c_j \lambda_j \\ \text{terhadap} \quad & \sum_{j \in J} \overline{\mathbf{a}}_j \lambda_j \geq \overline{\mathbf{b}} \\ & \lambda_j \geq 0, \ j \in J \end{aligned} \tag{2.38}$$

Dalam setiap iterasi metode simpleks, selalu dicari variabel non-basis yang akan masuk ke dalam basis, yang disebut dengan proses *pricing*. Pada proses ini, misalkan diberikan vektor variabel dual $\mathbf{u} \geq 0$, maka akan dicari

$$\arg\min \left\{ \bar{c}_j := c_j - \mathbf{u}^t \mathbf{a}_j | j \in J \right\}.$$

Variabel non-basis yang dipilih tentu saja menjadi sangat banyak jumlahnya bila $|J|$ sangat besar. Untuk itu, didefinisikan subproblem dari masalah *master problem*, yaitu hanya dengan melibatkan kolom-kolom λ_j dengan $j \in J' \subseteq J$ dengan jumlah yang cukup layak dikerjakan, yang disebut *restricted master problem* (RMP). Misalkan $\boldsymbol{\lambda}$ dan $\boldsymbol{\pi}$ berturut-turut adalah solusi optimal masalah primal dan solusi optimal masalah dual dari RMP yang bersangkutan. Jika kolom-kolom $\mathbf{a}_j$, dengan $j \in J$ dinyatakan sebagai anggota dari himpunan $\mathcal{A} \neq \emptyset$, dan koefisien biaya c_j dapat dihitung dari $\mathbf{a}_j$ melalui fungsi c, maka subproblem atau *oracle*

$$\bar{c}^* = \min \left\{ c(\mathbf{a}) - \boldsymbol{\pi}^t \mathbf{a} | \mathbf{a} \in \mathcal{A} \right\} \tag{2.39}$$

membentuk suatu proses *pricing*. Jika $\bar{c}^* \geq 0$, maka begitu pula $\bar{c}_j, j \in J$ (tidak ada *reduced cost* variabel non-basis yang negatif), dan solusi $\boldsymbol{\lambda}$ untuk masalah RMP adalah solusi optimal dari RMP dan juga solusi optimal dari masalah aslinya. Jika tidak, ditambahkan kolom pada masalah RMP yang diperoleh dari tabel optimal masalah RMP yang terakhir, kemudian masalah yang baru diselesaikan kembali. Subproblem (2.39) juga disebut **subproblem pembangkit kolom** (*column generator*).

2.1.4. *Metode Pembangkitan Kolom untuk Integer Programming.* Misalkan diberikan suatu *integer programming* (IP)

$$\begin{aligned} \text{minimumkan} \quad & z = \sum_{j \in J} c_j \lambda_j \\ \text{terhadap} \quad & \sum_{j \in J} \bar{\mathbf{a}}_j \lambda_j \geq \bar{\mathbf{b}} \\ & \lambda_j \in \mathbb{Z}_+, \ j \in J \end{aligned} \tag{2.40}$$

yang masalah linear relaksasinya diselesaikan dengan menggunakan metode pembangit kolom. Jika solusi optimal masalah linear relaksasinya sudah berupa bilangan bulat, maka solusi tersebut juga merupakan solusi optimal masalah aslinya. Jika tidak, metode/algoritme *branch & bound* digunakan untuk menyelesaikan masalah tersebut. Pada setiap *node* hasil pencabangan, relaksasi linear di setiap *node* diselesaikan dengan menggunakan metode pembangkitan kolom, sampai semua pencabangan dapat dihentikan sehingga diperoleh suatu kesimpulan.

3. Model Matematik Pengangkutan Sampah di Kota Bogor

Masalah penentuan rute optimal pengangkutan sampah di kota Bogor dimodelkan sebagai NRPKP yang telah dibahas pada bagian sebelumnya. Dalam masalah pengangkutan sampah ini:

(1) *node* pada graf adalah lokasi TPS/*transfer depo*,
(2) sisi adalah jalan yang menghubungkan TPS-TPS,
(3) c_{ij} adalah jarak antara TPS-i dengan TPS-j,
(4) *demand* p_i adalah kapasitas TPS/*transfer depo*, yaitu
   (a) kapasitas TPS : $0,5 - 1m^3$ [PU, 2004]
   (b) kapasitas *transfer depo* bervariasi.
(5) kapasitas kendaraan q adalah kapasitas kendaraan pengangkut, yaitu 8 m^3 untuk *dump truck,* dan 6 m^3 untuk *amroll truck,*
(6) banyaknya kendaraan yang digunakan (m) adalah sesuai dengan banyaknya kendaraan layak-pakai yang dimiliki Dinas Kebersihan Kota Bogor.
(7) Karena keterbatasan jumlah kendaraan, maka kendala panjang/waktu tidak diikutkan.
(8) Lokasi *container* tidak diikutkan dalam penentuan rute.

Pada gambar Rencana Tata Ruang Wilayah (RTRW) Kota Bogor tahun Anggaran 1999/2000 (edisi terakhir yang dimiliki Kantor Dinas Kebersihan dan Pertamanan per 2004), hanya dipetakan 152 tempat pengumpulan sampah, yang terdiri dari 48 *container,* 97 TPS (tempat pengumpulan sementara), dan 7 transfer depo. Pada prakteknya, 1 TPS sering mewakili beberapa tempat sampah yang ada di depan rumah warga. Dari data rute pengangkutan sampah armada *dump truck* Dinas Kebersihan Kota Bogor (per Desember 2005) perlu ditambahkan beberapa TPS dan *transfer depo* dibandingkan dari data TPS di RTRW Kota Bogor.

Pada daftar rute pengangkutan sampah armada *dump truck* Dinas Kebersihan Kota Bogor (per Desember 2005) terlihat terdapat beragam volume "TPS". Sebagai contoh, untuk kendaraan *dump truck* dengan nomor polisi F 8286 A melayani 24 TPS dengan volume 18 m^3, yang berarti bahwa rata-rata volume 1 TPS adalah 0.75 m^3. Untuk *dump truck* dengan nomor polisi F 8287 A rata-rata kapasitas TPSnya adalah 0,3 m^3. Selain itu kendaraan masih melayani pengambilan *door to door* yaitu mengambil sampah dari bak-bak sampah (kapasitas 0,4 - 0,6 m^3) [PU, 2004] sehingga "bak-bak sampah" ini perlu diwakili dengan "TPS". Jadi, *node* TPS yang dipetakan adalah didasarkan pada lokasi/jalan yang diketahui. Jika tidak diketahui secara eksplisit, maka kapasitas rata-rata TPS yang digunakan adalah $1m^3$.

Dengan banyaknya TPS yang harus dilayani, tentu memerlukan variabel yang sangat banyak jika akan diselesaikan dengan menggunakan ILP, dalam hal ini NRPKP. Untuk tahap penyelesaiannya, digunakan algoritme heuristik "*cluster first, route second*" terlebih dahulu

yang mengelompokkan TPS-TPS ini menjadi beberapa *cluster*. Untuk penyederhanaan, pengelompokan ini didasarkan pada pembagian kecamatan yang ada di Bogor, yaitu

(1) Kecamatan Tanah Sareal,
(2) Kecamatan Bogor Barat,
(3) Kecamatan Bogor Timur,
(4) Kecamatan Bogor Tengah,
(5) Kecamatan Bogor Selatan,
(6) Kecamatan Bogor Utara.

Untuk setiap wilayah dialokasikan sejumlah tertentu kendaraan, kemudian diformulasikan masalah NRPK(P) dan diselesaikan dengan algoritme *column generation*.

4. Kesimpulan dan Saran

Karena keterbatasan armada kendaraan, maka sulit ditemukan solusi optimal yang bisa mengangkut semua sampah setiap harinya. Di beberapa wilayah diperlukan lebih dari satu kali ritasi pengangkutan. Yang dapat dilakukan hanya mencari rute terpendek dari setiap rute kendaraan.

Selain itu karena banyaknya variabel, maka penghitungan rute dapat lebih rinci dilakukan per kecamatan, tapi tentu saja perlu data TPS yang lebih detil lagi.

Pustaka

[1] **Adiwibowo, S. 2004.** Pengelolaan Sampah Terpadu. Disampaikan dalam *Pelatihan Pengelolaan Sampah & Teknologi Pengomposan*. Pusat Penelitian Hidup IPB, Bogor.

[2] **Amponsah, S.K. & S. Salhi. 2004.** The investigation of a class of capacitated arc routing problems: the collection of garbage in developing countries. *Waste Management* **24:** 711 – 721.

[3] **Balas, E. & M.W. Padberg. 1979.** Set partitioning: A survey. Dalam *Combinatorial Optimization* (Christofides, N. et al. eds), John Wiley, NewYork.

[4] **Barnhart,C. E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, & P.H. Vance. 1998.** Branch-and-price: column generation for solving huge integer programs. *Operations Research* **46(3):** 316 – 329.

[5] **Bramono, S.E. 2004**. *Pengkajian keterkaitan sistem jalan dan transportasi terhadap sistem persampahan*. Departemen Teknik Lingkungan ITB. Tersedia di http://www.tlitb.org/ . [4 Juni 2004]

[6] **Christofides. 1985.** Vehicle routing. Dalam *The Traveling Salesman Problem*. Lawler *et al.*, eds, Wiley, New York.

[7] **Christofides,N., A. Mingozzi, & P. Toth. 1979.** The vehicle routing problem. Dalam *Combinatorial Optimization* (Christofides, N. et al. eds), John Wiley, NewYork.

[8] **Cullen, F.H., J.J. Jarvis, & H.D. Ratliff, 1981.** Set partitioning based heuristics for interactive routing. *Networks* **11:** 125 – 143.

[9] **Departemen Pekerjaan Umum. 2004.** Pengelolaan sampah di perumahan padat. Tersedia di http://www.pu.go.id/publik/infote~1/html/ind/sampah01.htm. [04 Juni 2004].

[10] **Desrosiers, J.F. 2004.** A primer in column gengeration. *Les Cahiers du GERAD G-2004-02. HEC Montreal.*

[11] **Desrosiers,J. F. Soumis, M. Desrochers. 1984.** Routing with time windows by column generation. *Networks* **14:** 545 – 565.

[12] **Desrosiers,J. F. Soumis, M. Desrochers, M. Sauvé. 1986.** Methods for routing with time windows. *European Journal of Operational Research* **23:** 236 – 245.

[13] **Ghiani,G, G. Laporte & R. Musmanno, 2003.** *Introduction to Logistics Systems Planning and Control.* Wiley, New York.

[14] **Kompas 10 Januari 2004.** Warga Harus Peduli Sampah.

[15] **Kulkarni, R.V. & P.R. Bhave, 1985.** Integer programming formulations of vehicle routing problems. *European Journal of Operational Research* **20:** 58 – 67.

[16] **Lübbecke, M.E., & J. Desrosiers. 2002.** Selected topics in column generation. *Les Cahiers du GERAD G-2002-64.* HEC Montreal.

[17] **Malandraki C. & M.S. Daskin. 1992.** Time dependent vehicle routing problems: formulations, properties and heuristic algorithms. *Transportation Science* **26(3):** 185 – 200.

[18] **Ralphs,T.K., L. Kopman, W.R. Pulleyblank, & L.E. Trotter,Jr. 2003.** On the capacitated vehicle routing problem. *Math. Programming. Ser. B* **(94):** 343 – 359.

[19] **Ribeiro, C.C. & F. Soumis. 1994.** A column generation approach to the multiple-depot vehicle scheduling problem. *Operations Research* **42(1):** 41 – 52.

[20] **Solomon, M.M. 1987.** Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research* **35(2):** 254 – 265.

[21] **Solomon, M.M. & J. Desrosiers. 1988.** Time window constrained routing and scheduling problems: survey paper. *Transportations Science* **22(1):** 1 – 13.

[22] **Villeneuve, D., J. Desrosiers, M.E. Lübbecke, F. Soumis. 2003.** On compact formulations for integer programs solved by column generation. *Les Cahiers du GERAD G-2003-06.* HEC Montreal.

[23] **Wilhelm, W. 2001.** A technical review of column generation in integer programming. *Optimization and Engineering* **2:** 159 – 200.

[24] **Wismanto, D. 2004.** *Pengelolaan Sampah Kota Bogor (Studi Kasus).* Pusat Penelitian Lingkungan Hidup IPB, Bogor.

[25] **Wismanto, D. 2005.** Komunikasi pribadi.

[26] **Wolsey, L.A. 1998.** *Integer Programming.* John Wiley & Sons, Chichester.

[27] **Yanadi, R. 2004.** Ke mana sampah Kota Bogor dibuang? (3-Habis): Kinerja dinas kebersihan buruk, Walikota berang. *Radar Bogor,* 1 Desember 2004.