IMPLEMENTASI ALGORITMA *BRANCH AND BOUND* PADA PENUGASAN *VIRTUAL PATH* CADANGAN DALAM JARINGAN BERMODUS TRANSFER SATU ARAH (*ASYNCHRONOUS TRANSFER MODE NETWORKS)*

DODY DHARMA*

*Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung
Perumahan Sukamenak Indah Blok.I No.62 Jalan Kopo Sayati Bandung
e-mail: dharma_xp2@yahoo.com , dody-dharma@students.itb.ac.id

ABSTRAK

Daya tahan (*survivability*) suatu jaringan merupakan persoalan yang sangat penting. karena para pengguna jaringan komputer harus terjamin kesuksesan transaksi data-datanya. Dalam Jaringan komputer berkeceptan tinggi dengan modus transfer satu arah (*high-speed Asynchronous Transfer Mode (ATM))* bisa saja terjadi kehilangan data dalam jumlah yang cukup besar yang diakibatkan oleh kegagalan jaringan (*network failure*) yang lebih jauh berdampak pada kerugian secara ekonomi. Makalah ini mengupas persoalan daya tahan jaringan (*network survivability*). Karakterisitik dari *virtual paths* dan pengaruhnya pada restorasi jaringan akan diuji. Sebuah persoalan baru, proses *Backup Virtual Path Routing,* ditampilkan sebagai Strategi proses perutean tujuan-lokal. Sebuah fungsi objektif, *flow lost* yang diakibatkan oleh terputusnya sebuah hubungan dalam jaringan ditetapkan sebagai indeks performa. Kami mengembangkan sebuah algoritma dengan menggunakan pendekatan *branch and bound*. Lebih dari itu, dua buah algoritma heuristik juga diajukan. Hasil kuantitatif juga ditampilkan.

Kata kunci: Jaringan berdayatahan, ATM, algoritma branch and bound

1. PENDAHULUAN

Dalam beberapa tahun terakhir kita telah mengamati sebuah peningkatan peran dari jaringan komputer dalam dunia modern. Sekarang ini Jaringan komputer menawarkan sebuah kesempatan besar untuk menyalurkan dan mengembangkan bisnis secara elektronis. Banyak perusahaan, organisasi dan institusi yang bergantung pada jaringan komputer, mempergunakan jaringan komputer sebagai medium basis untuk mengirimkan berbagai macam informasi. Sebuah kegagalan jaringan, bahkan yang kecil sekalipun, dapat mengakibatkan kerusakan dan konsekuensi yang amat banyak, termasuk kerugian ekonomi, hilangnya pendapatan secara signifikan, hingga konflik politik. Oleh karena itu kami menaruh perhatian pada perancangan metode restorasi yang dapat diterapkan untuk menunjang daya tahan jaringan.

Mengurangi biaya kostruksi dan memberikan daya tahan yang cukup merupakan persoalan utama yang dihadapi para perencana dan insinyur jaringan. Adanya peningkatan *bandwidth* pada transmisi optis bermakna bahwa kegagalan pada sebuah hubungan dalam jaringan (*link)* saja akan mempengaruhi banyak layanan (services) (Kawamura dan Tozikawa, 1995). Metode-metode restorasi yang dibutuhkan adalah metode yang menawarkan proses penyembuhan-mandiri (*self-healing*). Penyembuhan mandiri bermakana bahwa jaringan mampu me-rekonfigurasi dirinya sendiri ketika terjadi kerusakan atau kegagalan sehingga dapat diciptakan suatu keadaan dengan tingkat kehilangan lalu lintas data sekecil mungkin (Van Landegem *et al.,*1994).

Kegagalan jaringan terjadi karena berbagai alasan, yang mengakibatkan gangguan layanan dalam rentang detik hingga berminggu-minggu. Kejadian representatif yang menyebabkan kegagalan-kegagalan adalah insiden putusnya kabel, malfungsi perangkat keras, error pada perangkat lunak, bencana alam (kebakaran, banjir, dsb) dan kelalaian *(human error).* Kebanyakan penyebab kegagalan tersebut berada diluar kontrol penyedia jaringan. Sebuah jaringan yang terpercaya haruslah mampu menyediakan ketersediaan koneksi jaringan dengan kualitas tinggi *(high level availability).* Berhubung tidak ada sistem atau komponen yang tidak pernah rusak atau gagal, maka jaringan haruslah memiliki mekanisme proteksi dalam mengantisipasi kerusakan yang tak tercegah. Di satu sisi, interupsi lalu lintas data dapat dicegah sebanyak mungkin melalui *secure fibre routings*, proteksi kabel, serta standar desai dan keamanan yang tinggi pada peralatan. Di sisi lain, teknik perlindungan jaringan dipergunakan hanyalah untuk membatasi/ mengurangi pengaruh terhadap layanan ketika gangguan yang tak terelakkan dari luar muncul.

Ketika sebuah kegagalan pada jaringan muncul, sebuah mekanisme restorasi dibutuhkan. Proses restorasi jaringan mencakup: pendeteksian kegagalan, pengiriman informasi mengenai kegagalan, penambahan kapasitas (*spare capacity alocation*), strategi perutean (*rerouting strategies,* bagaimana lalu lintas data didistribusikan) dan pengendalian jaringan (*network control)*. Dalam makalah ini kami hanya membicarakan masalah strategi perutean dan distribusi lalu lintas data dalam Jaringan dengan modus *asynchronous transfer mode* (ATM *Networks*).

ATM adalah salah satu teknologi jaringan yang paling menjanjikan. ATM menawarkan: performa tinggi,

kemampuan untuk menyalurkan banyak tipe layanan (data, suara, dan video), kemampuan untuk mebawa lalu lintas data melalui berbagai macam fisik jaringan, serta jaminan kualitas layanan (*Quality of Service*, QoS) yang memfasilitasi aplikasi kelas baru semacam multimedia.

Pembahasan dalam makalah ini lebih terkonsentrasi pada isu mekanisme daya tahan yang dipergunakan pada jaringan ATM. Kami menampilkan basis metode restorasi, yang memfokuskan pada metode proteksi *virtual path*, yang di dalamnya terdapat tiga strategi restorasi perutean. Mari kita lihat salah satunya, katakanlah, perutean destinasi-lokal dan memformulasi persoalan optimasi kombinatorial yang disebut *Backup Virtual Path Routing (BVPR) problem for local-destination rerouting.* Ia adalah sebuah persoalan optimasi NP-*complete.* Fungsi objektif yang dipergunkan adalah fungsi *lost-flow*. Dalam literatur-literatur, persoalan ini tidaklah menerima banyak perhatian. Hanya ada beberapa algoritma heuristik sederhana yang memecahkan persoalan BVPR. Oleh karena itu kami mengkostruksi sebuah algoritma baru berdasarkan metode *branch and bound.* Hasil kuantitatif dari eksperimen juga disetrtakan dalam makalah ini.

2. KONSEP *VIRTUAL PATH* DALAM JARINGAN ATM BERDAYATAHAN

Dalam jaringan ATM informasi ditransportasikan pada paket kecil dengan panjang tetap yang dikenal sebagai sel. ATM adalah sebuah teknologi berorientasi-koneksi. Hal ini bermakna bahwa informasi antara sistem akhir dibawa sepanjang sebuah sirkuit virtual yang telah diciptakan. Perutean dilakukan pada saat proses *setup* koneksi dengan membuat masukan-masukan yang sesuai dalam tabel *look-up routing* pada setiap *switch.* Sirkuit ATM terdiri dari dua tipe: *virtual paths* (VP's), *identified by virtual path identifiers* (VPI's), dan *virtual channels (VC's), identified by VCI's*. VP adalah koleksi dari VC, dan dikirimkan bersamaan melalui sebuah *link*. Konsep *Virtual Path* memiliki banyak keuntungan(Kawamura *et al*., 1994; Kawamura dan Tozikawa, 1995) yaitu: menyederhanakan struktur jaringan karena *virtual path* bersifat non-hierarkis, adanya independensi dari penciptaan *path route* dan *bandwidth assignment (*sebuah rute yang ditentukan oleh sebuah VPI*)*, sebuah VP dengan *bandwith* nol, menyederhanakan kontrol dan pengelolaan jaringan, mengelompokkan lalu lintas data yang mirip (lalu lintas data yang memiliki parameter QoS yang sama) dalam satu *path.* Untuk informasi lebih lengkap mengenai *virtual path* lihat (Burgina d Dorman, 1991; Chlamatac *et al.*, 1994; Friesen *et al.*, 1996; Gerstel *et al.*, 1996; Sato *et al.*, 1990).

Van landegem *et al.* (1994) memaparkan empat metode restorasi penyembuhan-mandiri(*self-healing*), yaitu:

- *Automatic Protection Switching* (Ayanoglu and Gitlin, 1996; Veitch and Johnson, 1997);
- *Virtual Path Protection Switching* (Anderson *et al.*,1994; Ayanoglu and Gitlin, 1996; Kawamura *et al.*,1994; Kawamura and Tokizawa, 1995; Murakami and Kim, 1996; Veitch and Johnson, 1997);
- *Self-Healing Rings* (Kajiyama *et al.*, 1994; May *et al.*, 1995);
- *Flooding Algorithms* (Ayanoglu and Gitlin,1996; Kawamura and Tokizawa, 1995).

Semua metode diatas menggunakan beberapa sumber daya yang sama untuk menciptakan daya tahan jaringan. Perhatikan bahwa keempat metode diatas dapat mempergunakan sebuah *virtual path* sebagai basis elemen terproteksi. Dalam makalah ini kami hanya mengkonsentrasikan pada metode *virtual path protection swtiching*. Dalam metode ini elemen terproteksi yang menjadi basis adalah sebuah *virtual path*. Setiap virtual path dalam jaringan memiliki sebuah *virtual path* cadangan, Setelah sebuah kegagalan terjadi, *path* yang gagal akan dialihkan(*switched)* ke rute cadangan(*backup route).*Proses pengalihan adalah mudah. Prose ini juga menyertakan peroses pengubahan jumlah VPI di ATM *switches.* Konfigurasi dari *virtual path* cadangan dapat ditemukan melalui algoritma khusus dan diimplementasikan pada noktah-noktah dalam jaringan. Di awal, semua *virtual path* cadangan memiliki *bandwitdh* sama dengan nol, dan setelah proses aktivasi mereka akan diberikan *bandwidth* sesuai dengan yang diperlukan.

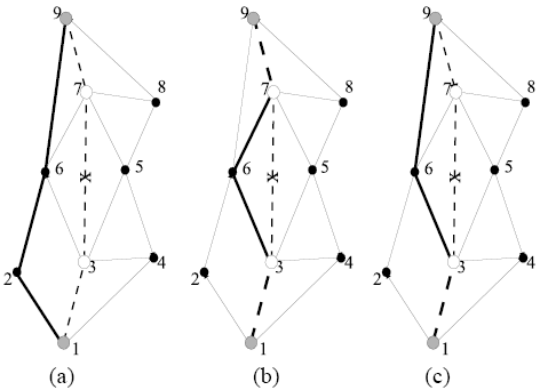
Metode *virtual path protection switching* terdiri dari dua fase. Fase pertama mengandung proses seleksi sebuah strategi perutean dan perancangan sebuah konfigurasi *path* cadangan dengan tujuan untuk mengotimasi kriteria daya tahan jaringan. Dalam (Nederlof *et al.,1995)* kriteria yang terpenting sebagai berikut: waktu restorasi, lalu lintas data yang hilang (*lost traffic*), jumlah kapasitas tersedia, jumlah-pesan pesan yang tercipta dan harga restorasi(*restoration cost*). Kedua, kalkulasi dari kapasitas tesedia dibutuhkan untuk menjamin tercapainya restorasi 100% ketika terjadi kasus kegagalan pada jaringan. Menurut (Murakami dan Kim, 1996), dalam jaringan fiber sebuah kegagalan *link*-tunggal adalah kejadian kegagalan yang paling umum dan paling sering dilaporkan. Oleh karena itu, kebanyakan model-model optimasi sebuah kegagalan pada *link*-tungal diambil sebagai ruang keadaan keseluruhan dari kegagalandan dan kapasitas *link* tersedia dihitung untuk menyediakan restorasi penuh dalam kasus sebuah kegagalan dari *link*-tunggal. Bagaimanapun juga, metode proteksi *virtual path* dapat juga diterapkan dalam jaringan dengan sumber daya terbatas (kapasitas *link*). Dalam jaringan semacam ini, restorasi 100% tidaklah selalu memungkinkan dan rute-rute didesain sedemikian rupa untuk meminimalkan pengaruh-pengaruh dari kegagalan yang terjadi. Dalam makalah ini kami mengasumsikan bahwa jaringan dipastikan memiliki kapasitas *link* yang tetap (*fixed link capacities*).

Dalam pendekatan kami, kami mengasumsikan sebuah *multiplexing* statistis dari lalu lintas data dengan parameter QoS yang mirip yang melalui sirkuit dalam *virtual path*, namun unutk lalu lintas data dengan *virtual path* yang bervariasi dalam satu *link* kami mempergunakan *multiplexing* deterministik. Oleh karena itu *bandwidth* dari *virtual path* dapat dengan mudah dihitung untuk memeriksa batasan kapsitas jaringan. konsep dari kapasitas ekivalen diajukan(Guerin *et al*., 1991; Hui *et al*., 1991) untuk menyediakan sebuah kesatuan fungsi yang

merepresentasikan beban dari *virtual path* dan dapat pula dipakai untuk menentukan syarat *bandwith* dari *virtual path*. Pendekatan ini menyederhanakan analisis.

Tiga stategi perutean diajukan dalam literatur *(*Anderson *et al*., 1994; Ayanoglu dan Gitlin, 1996):

- ***Source-Based Rerouting***(Gambar.1 (a)). Setiap *virtual path* memiliki sebuah cadangan, *link* disjoin dengan rute normal dari *virtual path*. *Virtual path* yang dipengaruhi adalah yang ditelusuri balik ke noktah sumber, yang merutekan kembali *virtual path* pada sebuah *path* cadangan yang telah dikalkulasi terlebih dahulu, atau noktah sumber/asal mencari rute cadangan berdasarkan seluruh informasi yang dimiliki oleh noktah. Kelebuihan utama *source-based rerouting* adalah bahwa ia mencakup seluruh jaringan, sehingga kapasitas yang tersedia digunakan secara efisien dan aktivasi *virtual path* didistribusikan ditengah-tengah banyak noktah. Tidak terlepas, pesan-pesan restorasi dalam jumlah besar diciptakan dalam jaringan, waktu restorasi cukup besar dan proses penentuan rute cadangan sangat kompleks.
- ***Local Rerouting***(Gambar.1 (b)). Teknik ini merupakan kebalikan dari teknik sebelumnya. Rute cadangan ditemukan hanya disekitar *link* yang gagal. Setiap *virtual path* memiliki banyak rute cadangan sebagai *links*. Setiap rute cadangan mencakup rute normal kecuali *link* yang gagal. Noktah-noktah yang bertetangga dengan *link* yang gagal bertanggungjawab atas proses *rerouting*. Semua *virtual path* diproses secara lokal, oleh karena itu waktu restorasi menjadi kecil. Kerugian dari *local-rerouting* adalah sebagai berikut: proses restorasi dilakukan memalui himpunan noktah yang sama, dan hanya sumber daya dari kapasitas *link* tersedia yang dekat dengan *link*- gagal yang dipakai.
- ***Local-Destination Rerouting*** (Gambar.1 (c)). Strategi ini merupakan penengah antara kedua metode sebelumnya. Dalam skema metode ini, rute cadangan disjoin dengan rute normal yang dimulai dari sebuah noktah awal dari *link* yang gagal. Oleh karena itu, setiap *virtual path* memiliki banyak rute cadangan sebagai *link*. Noktah permulaan dari *link* ynag gagal bertanggungjawab terhadap proses *rerouting*. Sekema ini menghasilkan *intermediate requirements* dari kapasitas *link* tersedia dan waktu restorasi.



Gambar 1. ***Rerouting Strategies:*** **(a)** ***source-based rerouting*** **(b)** ***local rerouting*** **(c)** ***local-destination rerouting***

Gambar 1 mengilustrasikan strategi perutean-kembali. *Virtual path* dengan rute kerja 1-3-7-9 rusak ketika *link* 3-7 mengalami kegagalan. Untuk *source-based rerouting* Noktah 1 diinformasikan tentang kegagalan yang terjadi dan ia bertanggung jawab untuk mengalihkan rute ke *path* cadangan 1-2-6-9 (Gambar. 1(a)). Dalam sekema *local rerouting*, Noktah 3 dihubungkan ke *path* cadangan 3-6-7 untuk menghindari *link* gagal (Gambar. 1(b)).Ini Menghasilkan rute cadangan 1-3-6-7-9. Terakhir, untuk *local-destination rerouting*, Noktah 3 dihubungkan ke *path* dengan rute 3-6-9, karena *path* tidak berubah dari noktah sumber ke titik awal dari *link* yang gagal,Ini menghasilkan rute cadangan 1-3-6-9 (Gambar. 1(c)). Jika kita memilih *local rerouting strategy*, path 3-6-9-7 unutk menghindari *link* 3-7, *virtual path* yang diambil akan mengalami *backhauling*, yang mana dua kesempatan ekstra dilalui(dari noktah9 ke noktah 7, dan kembali dari noktah 7 ke noktah 9) (Anderson *et al.*, 1994; Iraschko *et al.*, 1998).

3. PERSOALAN MERUTEKAN *VIRTUAL PATH* CADANGAN

Pada bagian ini kami mem-*formulasi* persoalan BVPR. Kami mengambil strategi *local-destination* untuk mengalirkan *rerouting* setelah terjadi kegagalan pada sebuah *link*. Anggap jaringan ATM dimodelkan sebagai graf berarah *G = (N,L,C)* diman *N* adalah himpunan yang mengandung *n* noktah yang merepresentasikan berbagai *Switch* pada jaringan ATM, *L* adalah himpunan *l* buha *link* dan *C* adalah sebuha vektor dari kapasitas *link*. Ambil *o(m)* mewakili noktah asal dari *link m*, dan *d(m)* sebagi noktah tujuan(akhir) dari *link m.* unutk mererpresentasikan maslah secara matematis, kami memperkenalkan notasi-notasi berikut ini:

c_i capacity of link i,
P set of p virtual paths in the network,
P_m set of p_m virtual paths which use link m,
Q_i estimated bandwidth requirement for VP i,
Π_{im} set of backup routes for VP i after a failure of link m, $\Pi_{im} = \{\pi_{im}^k : k = 1, \ldots, l_i^m\}$,

$$a_{ij} = \begin{cases} 1 & \text{if the working route for VP } i \text{ uses link } j \in L, \\ 0 & \text{otherwise,} \end{cases}$$

$$y_{im}^k = \begin{cases} 1 & \text{if } \pi_{im}^k \text{ is the backup route for VP } i, \\ & \text{after a failure of link } m, \\ 0 & \text{otherwise,} \end{cases}$$

$$y_{im}^0 = \begin{cases} 1 & \text{if the VP } i \text{ is not restored after a failure} \\ & \text{of link } m, \\ 0 & \text{otherwise,} \end{cases}$$

$$b_{im}^{kj} = \begin{cases} 1 & \text{if the route } \pi_{im}^k \text{ uses link } j, \\ 0 & \text{otherwise.} \end{cases}$$

Since the virtual path can use only one route, we have

$$\sum_{k=0}^{l_i^m} y_{im}^k = 1 \text{ for } m \in L, \quad i \in P_m. \tag{1}$$

Let $\hat{Y}_r^m$ be the permutation of the values of all variables y_{im}^k, $k = 0, 1, \ldots, l_i^m$, for which the condition (1) is satisfied, and let Y_r^m be the set of variables which are equal to one in the permutation $\hat{Y}_r^m$. The set Y_r^m is called a *selection*. Each selection determines a unique set of backup routes used for restoration of a survivable ATM network.

Let f_{jr}^m be the total flow on the j-th link after a restoration of the failed link m. We have

$$f_{jr}^m = \sum_{i=1}^{p} a_{ij} Q_i - \sum_{i=1}^{p} a_{ij} a_{im} Q_i + \sum_{i=1}^{p} \sum_{k=1}^{l_i^m} b_{im}^{kj} y_{im}^k Q_i. \tag{2}$$

Let R_Y^m denote the family of all selections Y_r^m for which the following condition is satisfied:

$$f_{ir}^m \le c_i \text{ for } i = 1, \ldots, l. \tag{3}$$

The objective function is formulated as follows:

$$LFB_m(Y_r^m) = \sum_{i=1}^{p} a_{im} y_{im}^0 Q_i. \tag{4}$$

$$\min_{Y_r^m} LFB_m(Y_r^m) \tag{5}$$

$$Y_r^m \in R_Y^m. \tag{6}$$

4. ALGORITMA *BRANCH AND BOUND*

Persoalan (5),(6) adalah *NP-complete* karena ia ekivalen dengan persoalan *non-bifurcated flow*, yang merupakan *NP-complete(*Fratta *et al*., 1973). Oleh karena itu kami akan mempergunakan algoritma *branch and bound* unutk membangun **algoritma eksak.** Metode *branch and bound* merupakan metode pencaraian terstruktur yang cerdas dalam ruang solusi yang mungkin(*feasible).* Ruang solusi dipartisi secara berulang-ulang menjadi subset-subset yang lebih kecil, dan sebuha *lower bound* dari fungsi objektif sikalkulasi dalam tiap subset. Subset-subset dengan*bound* yang diluar solusi terbaik akan dikeluarkan dari proses partisi berikutnya.

4.1. Skema Kalkulasi

Kami memulai dengan memilih Y_1^m dan menciptakan sebuah urutan penseleksian Yr^m. Untuk memperoleh Y_1^m awal kita harus memcahkan persoalan BVPR menggunkan nalgoritma heuristik. Ada dua elemen pentingn dalam algoritma ini yang diperhitungkan pada setiap seleksi Y_1^m : sebuah *lower bound* dari fungsi kriteria dan aturan pencabangan. *Loer bound* diperhitungkan unutk menguji apakah sebuah solusi yang "lebih baik" dapt ditemukan. Tugas dasar dari aturan pencabangan adalah unutk mencari variabel untuk komplementasi demi menciptakan seleksi baru dengan nilai kemungkinan terkecil dari fungsi kriteria

.

4.2. Algoritma Heuristik

Untuk memperoleh sebuah solusi awal, dari algortima ini, kami mengajukan 2 algoritma heuristik. Salah satunya akan melkukan modifikasi dari algoritma *flow deviation* unutk *non-bifurcated flow.* Kami memperkenalkan sebuah *link metric* yang berguna untuk menghitung panjang rute. Kami mengimplementasikan algoritma ini dan membandingkan hasilnya dengan hasil optimal dari lagoritma *branch and bound.*

Yang kedua adalah algoritma genetik. Kami menguji algortima ini pada persoalan optimasi yang berkaitan dengan BVPR.

4.3. Aturan Pencabangan

Tugas utama operasi ini adalah unutk memilih sebuah variabel normal dan unutk memundurkan keadaan variabel unutk mengkomplemenkan dan menciptakan sebuah suksesor Y_s^m dari seleksi Yr^m dengan nilai dengan kemungkinan terkecil dari fungsi kriteria.

4.3.1. Operasi Pemilihan

Berfungsi untuk mencari variabel $y_{im}^k = 0, y_{im}^h = 1$, $y_{im}^k \in Y_r^m$, dan untuk menciptakan suksesor $Y_s^m = (Y_r^m - \{y_{im}^k\}) \cup \{y_{im}^h\}$.

We introduce a metric representing the length of the route π_{im}^k:

$$l_r^{FB}(\pi_{im}^k) = -\left(a_{im} \min_{j: b_{im}^{kj}=1} (e_{jr}^m) \right). \tag{7}$$

Notice that if we look for the shortest route according to metric $l_r^{FB}(\pi_{im}^k)$, we select a route for which the minimal value of the residual capacity e_{jr}^m of all the links belonging to π_{im}^k is the largest:

$$\left(-l_r^{FB}(\pi_{im}^k)\right) \ge Q_i. \tag{8}$$

Let Q_{ri}^m denote the maximal value of the bandwidth of all the virtual paths i for which $y_{im}^k \in O_r^m$, i.e.

$$Q_{ri}^m = \max_{j: y_{jm}^k \in O_r^m} Q_j. \tag{9}$$

$$\Lambda_{imr}^{kh} = a_{im} \sum_{j \in D_r^m} (b_{im}^{kj} - b_{im}^{hj}). \tag{10}$$

4.3.2. Operasi Regulasi

Operasi regulasi dieksekusi jika pda proses eleksi Y_r^m batras kapasitas (3) tidak memuaskan. Ambil K_r^m menjadi himpunan yang mencakup semua *link* j dengan *violated capacity constarint*

$$K_r^m = \bigcup_{j \in L} \{j : f_{jr}^m > c_j\}.$$

$$\Psi_{im}^k = a_{im} \sum_{j \in K_r^m} (b_{im}^{kj} - b_{im}^{hj}). \quad (11)$$

Tujuan dari proses ini adalah untuk mengurangi aliran pda *link* yang dimiliki himpunan K_r^m.

4.4. Lower Bounds

4.4.1 Lower Bounds LB_{rm}^{LFB1}

$$LB_{rm}^{LFB1} = \sum_{i: y_{im}^0 \in U_r^m} (y_{im}^0 Q_i) + \sum_{i \in P_r^m} (Q_i) - KP(m,r). \quad (12)$$

4.4.2 Lower Bounds LB_{rm}^{LFB2}

$$LB_{rm}^{LFB2} = \sum_{i: y_{im}^0 \in U_r^m} (y_{im}^0 Q_i) + \sum_{i \in P_r^m} (Q_i) - \sum_{i \in L(m)} (c_i - f_{ir}^{mu}). \quad (13)$$

4.4.2 Lower Bounds LB_{rm}^{LFB3}

$$LB_{rm}^{LFB3} = \sum_{i: y_{im}^0 \in U_r^m} (y_{im}^0 Q_i) + \sum_{i \in P_r^m} (Q_i) - MF(m,r). \quad (14)$$

4.5. Algoritma

Let $Y_1^m \in R_Y^m$ be the initial solution. We assume that $U_1^m = \emptyset$, $U_1^{mt} = \emptyset$ and $LFB^* = \infty$. Let LB_{rm}^{LFB} denote the lower bound of the current selection Y_r^m. Depending on to which lower bound is applied, we select $LB_{rm}^{LFB} = LB_{rm}^{LFB1}$, $LB_{rm}^{LFB} = LB_{rm}^{LFB2}$ or $LB_{rm}^{LFB} = LB_{rm}^{LFB3}$.

Step 1. (Test step) If there exists a link i such that $f_{ir}^{mu} > c_i$, then go to Step 5. Otherwise, compute LB_{rm}^{LFB}. If $LB_{rm}^{LFB} \geq LFB^*$, then go to Step 5. Otherwise, go to Step 2.

Step 2. (Evaluation step) Compute $LFB(Y_r^m)$ and set M_r^m. If for all links i the condition $f_{ir}^m \leq c_i$ is satisfied, then $M_r^m := M_r^m - U_r^{mt}$. If $LFB(Y_r^m) < LFB^*$, then $LFB^* := LFB(Y_r^m)$. If $LFB^* = 0$, then terminate the algorithm. The set Y^* assigned to the current value of LFB^* is the optimal solution. If $LFB^* > 0$, go to Step 3. If the condition $f_{ir}^m > c_i$ is not satisfied for all links i, then identify a new set K_r^m including all such links. Go to Step 4.

Step 3. (Choice operation) If $M_r^m = \emptyset$, then go to Step 5. Otherwise, find the set O_r^m. Write $O_r^m := O_r^m - U_r^{mt}$. If $O_r^m \neq \emptyset$, then select variables $y_{im}^0 \in E_r^{m0}$, $y_{im}^k \in O_r^m$ with the satisfied condition (9). Generate a new selection Y_s^m (the successor of the current selection Y_r^m) in the following way:

$$Y_s^m := (Y_r^m - \{y_{im}^0\}) \cup \{y_{im}^k\},$$

$$U_s^m := U_r^m \cup \{y_{im}^k\}, \quad U_s^{mt} := U_r^{mt}.$$

Go to Step 1. If $O_r^m = \emptyset$, then select variables $y_{im}^k \in E_r^m$, $y_{im}^h \in M_r^m$ for which Λ_{imr}^{kh} in (10) has the maximal value. Generate a new selection Y_s^m (the successor of the current selection Y_r^m) in the following way:

$$Y_s^m := (Y_r^m - \{y_{im}^k\}) \cup \{y_{im}^h\},$$

$$U_s^m := U_r^m \cup \{y_{im}^h\}, \quad U_s^{mt} := U_r^{mt}.$$

Go to Step 1.

Step 4. (Regulation operation) If $M_r^m = \emptyset$, then go to Step 5. Otherwise, select variables $y_{im}^k \in E_r^m$, $y_{im}^h \in M_r^m$ for which Ψ_{imr}^{kh} in (11) has the maximal value. Generate a new selection Y_s^m (the successor of the current selection Y_r^m) in the following way:

$$Y_s^m := (Y_r^m - \{y_{im}^k\}) \cup \{y_{im}^h\},$$

$$U_s^m := U_r^m \cup \{y_{im}^h\}, \quad U_s^{mt} := U_r^{mt}.$$

Go to Step 1.

Step 5. (Backtracking step) Backtrack to the predecessor Y_p^m of the selection Y_r^m. If the selection Y_r^m has no predecessor, then the algorithm terminates. The selection Y* associated with the current LFB^* is optimal. Otherwise, drop the data for Y_r^m and update the data for Y_p^m as follows: If Y_r^m has been generated from Y_p^m by complementing $y_{im}^h := 1$, $y_{im}^k := 0$, then $U_p^{mt} := U_p^{mt} \cup \{y_{im}^h\}$. If the backtracking is performed for the $(l_i^m$-1)-th time by reverse variables of the normal variable y_{im}^k, then

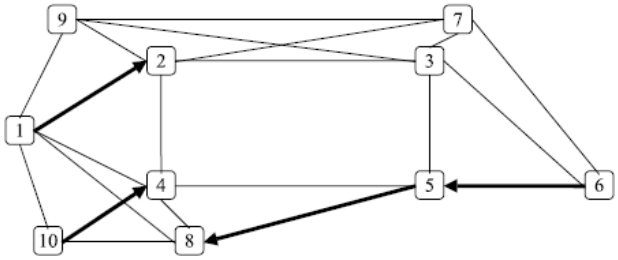
$$U_p^m := U_p^m \cup \{y_{im}^k\}, \quad U_p^{mt} := U_p^{mt} - \left(\bigcup_{a=1}^{l_i^m} \{x_i^a\}\right).$$

Go to Step 1.

5. CONTOH-CONTOH NUMERIK

Kami mengimplementasikan algoritma *branch and bound* dan algoritma heuristik berdasarkan metode FD dalam bahasa C++ dan melakukan sejumlah eksperimen numerik dalam jaringan yang memiliki rentang yang luas. Kami melakukan 302 tes terhadap 20 *links* yang dimiliki oleh tiga jaringan yang berbeda.

Semua jaringan yang diuji memiliki 10 noktah. Mereka meiliki jumlah *link* yang berbeda-beda: 46(NET46), 42(NET42), dan 36(NET36). Salah satu jaringan (NET42) dipresentasikan dalam gambar 2. *Link-link* yang telah diuji ditandai oleh garis tebal.

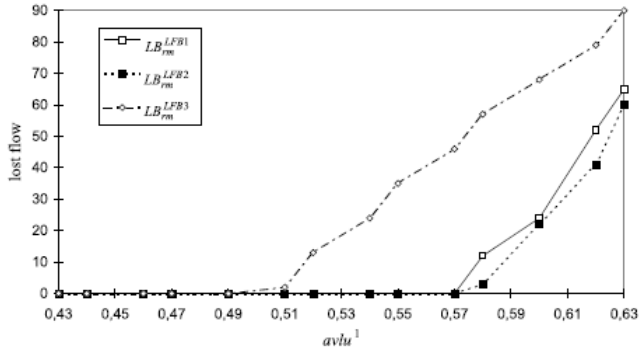


Gambar 2. Sebuah graf merepresentasikan jaringan Net42 yang diuji

Untuk mengevaluasi hasil-hasil yang diperoleh, kami menetapkan beberapa parameter. Misalkan $avlu^h$ dicatat sebagai rerata utilisasi *link* unutk semua *links* yang meninggalkan noktah *h*. untuk menghitung $avlu^h$, kita harus menjumlahkan aliran-aliran data dari semua *link* yang meninggalkan noktah *h* dan membagi jumlah kapasitas dari semua *link* yang meninggalkan noktah *h.*oleh karena itu parameter $avlu^h$ dapat diinterpretasikan sebagai sebuah saturasi berbagai *link* yang meninggalkan noktah *h*. Parameter nd_m adalah jumlah *link* yang meninggalkan *o(m)*. *links* yang diuji memiliki nilai nd_m berkisar antara 3 sampai 6.

5.1 Perbandingan *Lower Bounds*

Kami melakukan perbandingan terhadap *lower* bound yang ditampilkan. Gambar 3 memperlihatkan sebuah perbandingan dari 3 buah *lower bound* untuk panah (1,2) dari jaringan NET42 dengan paramneter nd_m sama dengan 5. Kami memperhatikan bahwa nilai kecil dari nd_m dari semua *lower bounds* memberikan hasil yang dapat dibandingkan. Kalkulasi dari LB_{rm}^{LFB2} adalah yang paling sederhana. Untuk memperoleh LB_{rm}^{LFB1} kita harus memecahkan sebuah persoalan *knapsack* NP-*complete*. Kalkulasi dari LB_{rm}^{LFB3} mensyaratkan perhitungan sebuah aliran data yang maksimal.



Gambar 3. Perbandigan *lower bounds* untuk *link*(1,2) dari NET42(nd_m=5)

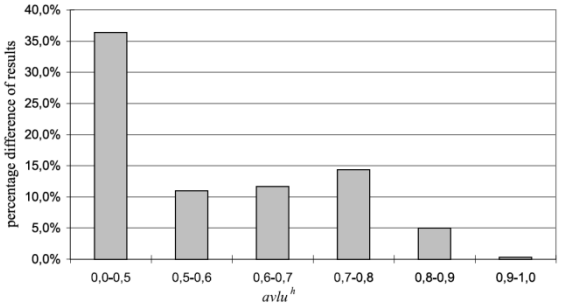
5.2 Evaluasi Hasil untuk Algoritma Heuristik

Untuk mengavaluasi hasil-hasil dari algoritma heuristik berdasarkan metode deviasi aliran data(*flow deviation)*, kami membandingkan hasil-hasil ini dengan hasil yang optimal yang diberikan oleh algoritma *branch and bound*.

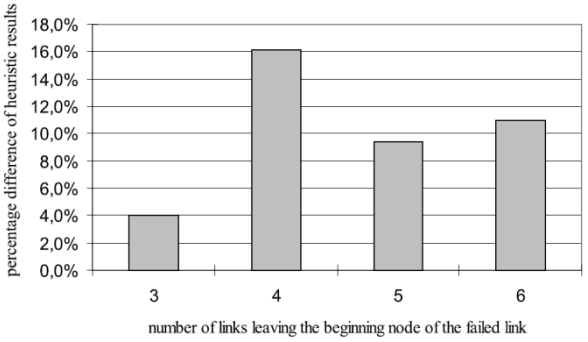
Kami melaukan 302 buah tes unutk 20 *link* dari 3 buha jaringan. Merangkum haildari semua tes, algoritma heurstik memberikan hasil 9,4% lebih buruk dari hasil optimal yang diperoleh dari algoritma *branch and bound.*

Gambar 4 memperlihatkan persentase selisih antara hasil yang diberikan oleh algoritma heuristik dan hasil optimal terhadap rerata utilisasi dari *link* yang meninggalkan noktah awal dari *link* yang gagal(rusak). 6 kisaran nilai dari parameter $avlu^h$ ditampilkan.

Catat bahwa untuk kenaikan nilai dari parameter $avlu^h$ hasil dari algoritma heuristik semakin medekati solusi optimal. Untuk kisaran $avlu^h$ dari 0,9 hingga 0,1 selisih nilainya hanyalah 0,3%. Ini disebabkan oleh fakta bahwa semakin besar nilai $avlu^h$ maka semakin tinggi pula nilai absolut dari fungsi *LFB.* Akibatnya, nilai relatif dari persentase selisih semakin kecil.



Gambar 4. Persentase perbedaan antara hasil dari algoritma Heuristik dengan algoritma *branch and bound*



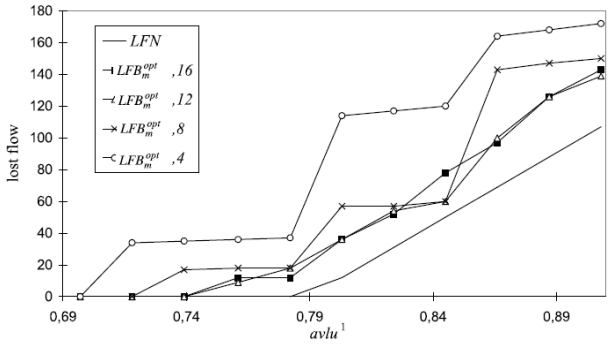
Gambar 5. Persentase selisih antara hasil dari algoritma Heuristik dengan algoritma *branch and bound* terhadap jumlah *link* yang ditinggalkan noktah awal dari *link* yang gagal(rusak).

5.3 Pengaruh dari Jumlah V*irtual Path* yang Menggunakan *link* yang Gagal(Rusak).

Kami juga menguji pengaruh dari jumlah *virtual path* yang menggunakan *link* gagal terhadap nilai optimal dari fungsi *LFB.* Kami melkukan percobaan numerikal untuk *link(1,2)* dari NET42. Dengan menggunakan algoritma *branch and bound* LFB_m^{opt}, kami menghitung nilai optimal dari fungsi *LFB* terhadap berbagai jumlah *virtual path* yang menggunkan *link* rusak. Kami mengasumsikan

bahwa jumlah semua *bandwidth virtual path* adalah konstan, tidak tergantung jumlah *virtual path*.

Gambar 6 menampilkan hasil yang bersesuaian. Cata bahwa lebih banyak virutal path yang dapat meng-utilisasi sumber daya jaringan adalah lebih baik, karena *bandwidth* path menjadi semakin kecil.



Gambar 6. Hubungan Nilai optimal dari fungsi *LFB* untuk beragam jumlah *virtual path* yang melalui *link*(1,2) dari NET 42 terhadap nilai dari parameter $avlu^{h=1}$.

6. SIMPULAN

Algoritma *branch and bound (exact algorithm)* digunakan secara luas unutk memperoleh solusi optimal pada persoalan optimasi jaringan yang berkaitan dengan persoalan BVPR. Dengan memiliki sebuah solusi optimal yang diberikan oleh sebuah algoritma eksak, kita dapat mengevaluasi tingkat efisiensi dari algoritma heuristik dengan membandingkan hasil yang diberikannya dnegan hasil yang optimal. Lebih jauh lagi, algortima eksa dapat dipakai pada fase perancangan aliran pada jaringan. ATM *networks* dipergunakan secara luas sebagai tulang punggung yang membawa lalu lintas data antara banyak lokasi yang tersebar luas. Dengan memiliki persyaratan *bandwith* yang telah diperkirakan, kita dapat meng-*assign* rute optimal ke *virtual path* dan menyebar dan menyelamatkan uang dalam jumlah yang signifikan. Terakhir,dalam opini kami, pengembangan algoritma eksak sangat membantu untuk memahami persoalan optimasi dan bahkan dapat menjadi sangat lebih berguna dalam membangun algoritma heuristik.

Metode yang diajukan pada makalah ini dapat dipergunakan untuk menciptakan daya tahan pada jaringan ATm yang ada. *switch* pada ATM *Networks* membutuhkan perangkat lunak khusus untuk memungkinkannya melakukan *rerouting* terhadap *virtual path* yang dipengaruhi oleh kegagaalan jaringan. Perangkat lunak ini haruslah mengandung algoritma yang mengkalkulasi rute dari *pah* cadangan.Algortima yang dibahas pada makalh ini dapat dipergunakan unutkl tujuan tersebut.

REFERENSI

[1] Walkowiak K, “The Branch and Bound Algorithm for a Backup Virtual Path Assignment in ATM Networks”, Int.J.Appl.Math.Comput.Sci, Vol.12, No.2, 2002,257-267.

[2] Anderson J., Doshi B., Dravida S. and Harshavardhana P. (1994): “Fast restoration of ATM networks”.—IEEE JSAC, Vol. 12, No. 1, pp. 128–138.