PEMBENTUKAN SET JALUR ALIRAN PROGRAM MENGGUNAKAN TEKNIK JUMLAH JALUR MINIMUM DAN TEKNIK JUMLAH PREDIKAT MINIMUM

Rosa de Lima Endang Padmowati
Jurusan Teknik Informatika, Fakultas Teknologi Informasi dan Sains, Universitas Katolik Parahyangan Bandung
Email: rosad5@home.unpar.ac.id

ABSTRACT
Structured testing is based on the control flow of programs. The analysis of the control flow of program is conducted on a reduced flowgraph, called a decision-to-decision graph (DDGraph). The relations of dominance and implication between arcs enable to immediate identification of a subset of DDGraph arcs. These arcs are called unconstrained arcs, which have a property that, when the unconstrained arcs are exercised, the traversal of all the other arcs are guaranteed.
In order to find a path cover for given program flowgraph, there are two methods: minimum number of paths technique and less pred technique.

Keywords: *decision-to-decision graph, the minimum number of paths*

ABSTRAK
Pengujian terstruktur dilakukan berdasarkan pada aliran kontrol program. Analisis aliran kontrol program dilakukan pada flowgraph yang dikurangi, disebut dengan decision-to-decision graph (DDGraph). Hubungan dominasi dan implikasi antara busur memungkinkan identifikasi-langsung subset dari busur DDGraph. Busur ini disebut busur tanpa konstrain (unconstrained arcs), dengan propertinya yaitu ketika busur tanpa konstrain (unconstrained arcs) ini dijalankan, maka dijamin adanya lintasan untuk semua busur yang lain.
Untuk menemukan jalur aliran graf dari suatu program, terdapat dua metode: teknik jumlah jalur minimum dan teknik jumlah predikat minimum.

Kata Kunci: *decision-to-decision graph, jumlah jalur minimum*

Pengujian suatu program dapat dilakukan secara fungsional dan secara struktural [1]. Pengujian fungsional artinya program diuji kelayakan kualitasnya dengan cara memasukan sejumlah kasus uji dan menguji kesesuaian hasil eksekusi dengan hasil yang diharapkan. Metode ini menempatkan sebuah perangkat lunak sebagai sebuah kotak hitam sehingga penguji program tidak perlu mengetahui struktur atau isi program dan cukup menyediakan satu set data input sebagai kasus uji. Kelemahan metode ini adalah tidak ada jaminan bahwa satu set data input tersebut sudah mewakili semua kasus uji, sedemikian sehingga semua pernyataan dalam program terjamin tereksekusi.

Pengujian struktural artinya program diuji kelayakan kualitasnya dengan cara memasukan sejumlah kasus uji untuk setiap satu jalur aliran kendali program. Penguji program harus mengetahui struktur atau isi program dan menyiapkan satu set jalur aliran program yang memuat sekumpulan jalur yang terjamin telah mencakup seluruh pencabangan program atau simpul keputusan. Penguji program menyediakan satu set data input sebagai kasus uji untuk sebuah jalur. Pengujian struktural dengan cara eksekusi setiap jalur dalam set jalur dapat menjamin semua pernyataan program akan tereksekusi.

Proses pembentukan satu set jalur yang mencakup seluruh pencabangan program dapat dilakukan dengan dua teknik [2]. Teknik Jumlah Jalur Minimum adalah teknik yang menghasilkan satu set jalur dengan jumlah jalur seminimal mungkin, sehingga setiap jalur dapat memuat sebanyak-banyaknya pencabangan program atau simpul keputusan. Teknik Jumlah Predikat Minimum adalah teknik yang menghasilkan satu set jalur, tanpa dibatasi jumlah jalurnya, dengan setiap jalur memuat jumlah pencabangan program atau simpul keputusan seminimal mungkin.

GRAF BERARAH DAN DDGRAF

Untuk menjalankan kedua teknik tersebut, dibutuhkan aplikasi teori graf berarah yang umum disebut Directed Graph (DiGraf) [3]. Sebuah unit program dapat dipresentasikan menjadi sebuah Digraf G = $\{S, B\}$, yang terdiri dari satu set simpul S = $\{1, 2, \ldots, n\}$ dan satu set busur berarah B = $\{e_1, e_2, \ldots, e_k\}$. Setiap busur dalam DiGraf mewakili setiap pernyataan dalam program. DiGraf dibangun dengan busur masukan dan busur keluaran yang unik dimana $e_1 \neq e_k$. Setiap pencabangan berupa kondisi dan pengulangan program direpresentasikan dengan pencabangan busur pada DiGraf G.

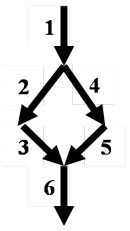
Gambar 1 memperlihatkan sebagian pernyataan dalam program Hitung_Akar. Segmen program ini memuat enam pernyataan program dengan satu pencabangan program atau simpul keputusan.

Segmen program Hitung_Akar pada Gambar 1 dapat dipresentasikan menjadi sebuah graf berarah. Gambar 2 memperlihatkan Digraf G merupakan hasil pemodelan Gambar 1. Digraf G memuat enam busur dengan busur $e_1 \neq$ busur e_6 .

Setiap busur pada DiGraf G mewakili sebuah pernyataan program. Pembentukan sebuah jalur berawal dari busur masukan (busur-1) dan berakhir pada busur keluaran (busur-6). Dari digraf G, dapat dibentuk sebuah set jalur yang memuat dua jalur yaitu jalur-1: 1-2-3-6 dan jalur-2 : 1-

```
PROGRAM Hitung_Akar:
...
CALL HITUNG_DETERMINAN  (1)
IF DET >= 0 THEN (2)
CALL HITUNG_AKAR_REAL (3)
ELSE (4)
CALL HITUNG_AKAR_IMAJ (5)
ENDIF
PRINT AKAR_PERS_KUADRAT (6)
...
```

Gambar 1: Segmen Program Akar_Kuadrat.



Gambar 2: DiGraf G; $e_1 \neq e_6$

4-5-6. Setiap jalur akan digunakan oleh penguji program dengan memasukkan berbagai data input sebagai kasus uji.

Dalam sebuah DiGraf dapat terjadi sekumpulan busur yang sejenis, sehingga harus dilakukan proses reduksi. Proses reduksi bukan berarti menghilangkan pernyatan program, tetapi menggabungkan sekumpulan pernyataan sejenis menjadi satu segmen program. Untuk hal ini sebuah busur merepresentasikan sebuah segmen program. Proses reduksi untuk DiGraf G diawali dengan cara menggabungkan busur-2 dan busur-3 menjadi busur-2 dan menggabungkan busur-4 dan busur-5 menjadi busur-3, dan busur-6 berubah nama menjadi busur-4, sehingga DiGraf G' memiliki 4 busur. Digraf jenis ini disebut Decision-to-Decision Graph (DDGraf). Sebuah DDGraf terutama memodelkan aliran kendali pencabangan yaitu analisis kasus dan pengulangan program sehingga busur menjadi penghubung antar simpul keputusan.

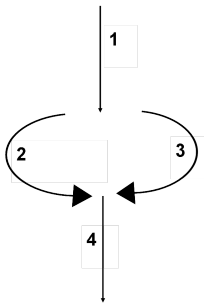
Definisi-1. DDGraf
Suatu DDGraf adalah suatu digraf G = (S, B) dengan busur masukan e_1 dan busur keluaran e_k yang unik $(e_1 \neq e_k)$. Untuk setiap busur $e \in B$, minimal ada satu jalur dari busur masukan e_1 ke busur e dan dari busur e ke busur keluaran e_k.

Gambar 3 memperlihatkan sebuah DDGraf G' hasil reduksi dari DiGraf G pada Gambar 2. DDGraf G' memuat 4 busur dan pada G' dapat dibentuk satu set jalur yang memuat 2 jalur yaitu jalur-1: 1-2-4 dan jalur-2: 1-3-4.

POHON DOMINASI DAN POHON IM-PLIKASI

Untuk dapat membentuk set jalur, harus dibentuk sebuah pohon Dominasi dan Pohon Implikasi dari sebuah DDGraf. Dari pohon tersebut akan diperoleh jalur-jalur digraf sehingga terbentuk set jalur. Gambar 4 memperlihatkan sebuah fungsi GetOp [4] sebuah modul bahasa C untuk mengambil Operator atau Ope-rand berikutnya dalam program Kalkulator.

Modul program GetOp memuat 10 simpul keputusan



Gambar 3: Decision-to-Decision Graf G'

dan dipresentasikan menjadi DDGraf G1 (Gambar 5). Pada DDGraf G1 dimuat 14 simpul dan 25 busur dengan $e_1 \neq e_{25}$. Dalam DDGraph sebuah busur menjadi representasi sebuah segmen program, sehingga relasi antar busur lebih diperlukan daripada relasi antar simpul. Dua relasi antar busur diperlukan yaitu relasi dominasi dan relasi implikasi.

Definisi-2. Relasi Dominasi
Diketahui sebuah DDGraf G = $(S, B)\, e_1 \neq e_k$. Busur e_i mendominasi (dominates) busur e_j bila setiap jalur P dari busur masukan e_1 ke busur e_j harus melalui busur e_i. Busur yang mendominasi busur e_{10} adalah: e_1, e_3, e_5. Busur-busur yang mendominasi busur e_{15} adalah: e_1, e_3, e_5, e_{10}, e_{11}.

Definisi-3. Relasi Implikasi
Diketahui sebuah DDGraf G = $(S, B)\, e_1 \neq e_k$. Busur e_i melibatkan (implies) busur e_j bila setiap jalur P dari busur e_i ke busur keluaran e_k harus melalui busur e_j. Busur yang dilibatkan oleh busur e_8 adalah: e_9, e_{10}, e_{25}. Busur-busur yang dilibatkan oleh busur e_{14} adalah: e_{15}, e_{20}, e_{25}.

Setiap busur dapat memiliki lebih dari satu busur yang mendominasinya dan lebih dari satu busur yang dilibatkannya. Busur e_i disebut sebagai busur yang mendominasi-langsung *(immediate dominator)* busur e_j apabila setiap busur lain yang mendominasi busur e_j juga mendominasi busur e_i. Sebuah Pohon Domi-nasi *(dominator tree)* dapat dibentuk melalui relasi tersebut. Busur e_i disebut sebagai busur yang dilibatkan-langsung-oleh *(immediately implied)* busur e_j apabila setiap busur lain yang dilibatkan-oleh busur e_i juga dilibatkan-oleh busur e_j. Sebuah Pohon Implikasi *(implied tree)* dapat dibentuk melalui relasi tersebut.

Gambar 6 dan Gambar 7 memperlihatkan Pohon Dominasi (PD) dan Pohon Implikasi (PI) DDGraf G1. Simpul pohon dengan *outdegree* = 0 disebut daun.

PEMBENTUKAN SET JALUR

Dengan memperhatikan relasi dominasi dan relasi implikasi antar busur sesuai Definisi-2 dan Definisi-3, maka dapat dibentuk satu set busur bebas.

Definisi-4. Busur bebas
DDGraf G = $(S, B)\, e_1 \neq e_k$ Suatu busur $e \in B$ disebut bebas jika untuk setiap busur e' lainnya pada G, ada minimal satu jalur dari e_1 ke e_k yang memuat e' dan tidak memuat e. Set busur bebas merupakan irisan set daun PD(G) de-

```
getop (s,lim) /*get operator or operand*/
char s[]
int lim;
{
int i,c
while((c=getch())==''; || c=='t' || c=='\n';)
/*simpul keputusan-1 (simpul-1)*/

if(c!='.' && (c<'0'||c>'9'));
/*simpul keputusan-2 (simpul-2)*/

return (c);
s[0] = c;
for(i=1; (c=getchar())>= '0' && c <= '9'; i++)
/*simpul keputusan-3 (simpul-3)*/

if(i<lim) /*simpul keputusan-4(simpul-4)*/

s[i] = c;
if(c=='.') /*collect fraction*/
/*simpul keputusan-5 (simpul-6)*/

if(i<lim)
/*simpul keputusan-6 (simpul-7)*/
s[1] = c;
for(i++; (c=getchar())>='0' && c<='9'; i++)
/*simpul keputusan-7 (simpul-9)*/
if(i<lim)
/*simpul keputusan-8 (simpul-10)*/
s[i] = c;
}

if(i<lim) /*number is ok*/
/*simpul keputusan-9 (simpul-12)*/
{
ungetch(c);
s[i] = '\0';
return(NUMBER);
}
else
{/*it's too big; skip rest of line*/
while(c!='\n'&&c!=EOF)
{
/*simpul keputusan-10(simpul-13)*/
c=getchar();
s[lim-1]='\0';
return(TOOBIG);
}
}
```

Gambar 4: Fungsi GetOp

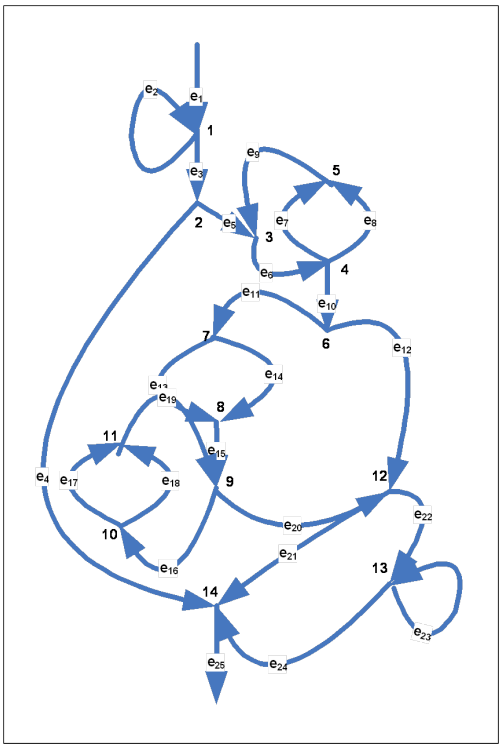
ngan set daun PI(G). BB(G) = DPD(G) ∩ DPI(G) dimana DPD(G) adalah set daun pada PD(G) dan DPI(G) adalah set daun pada PI(G). Dari DDGraf G1 diperoleh:
BB(G1) = DPD(G1) ∩ DPI(G1)
BB(G1) = $\{e_2, e_4, e_7, e_8, e_{12}, e_{13}, e_{14}, e_{17}, e_{18}, e_{21}, e_{23}\}$

Prosedur $Bentuk\_Satu\_Jalur$ dilakukan dengan cara mengambil sebuah busur bebas e_i yang belum terpakai, dan membentuk jalur dari busur masukan e_i ke busur bebas e_i (diambil dari PD(G)) dilanjutkan dari busur bebas e_i ke busur keluaran e_k(diambil dari PI(G)). Jika semua busur bebas dalam DDGraf sudah tercakup, maka proses pembentukan sebuah set jalur selesai [3].

Busur bebas dipilih mulai dari busur bebas dengan indeks penghubung tertinggi. Dalam DDGraf, G1 busur bebas dengan indeks penghubung tertinggi yaitu e_{23}. Prosedur ini membangkitkan satu jalur P(e_{23}) dari busur e_1 ke busur e_k melalui busur e_{23}.

Pada proses pembentukan set jalur mungkin terjadi kasus diskontinuitas. Pada PD(G1) dan PI(G1) (Gambar 6 dan Gambar 7) kasus ini ditandai dengan simbol =.

Prosedur $Bentuk\_Satu\_Jalur$ pada bagian DDGraf G = $\{S, B\}$; $e1 \neq ek$ adalah:
1) Ambil sebuah busur bebas yang belum digunakan dengan indeks penghubung tertinggi yaitu e_u

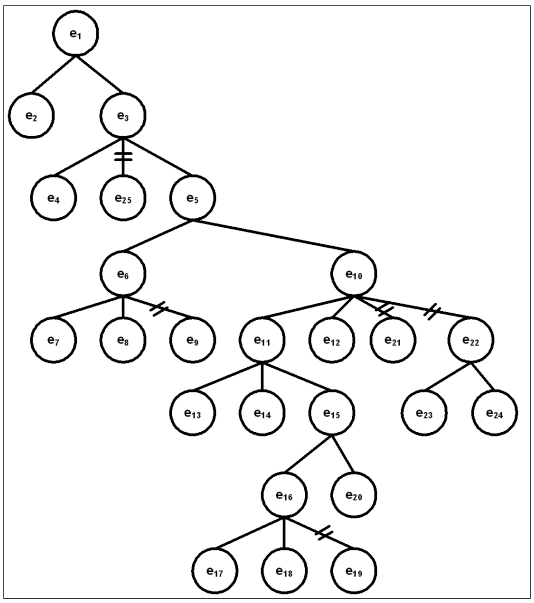


Gambar 5: Decision-to-Decision Graf G1

2) Bentuk jalur D(e_u) yaitu sebuah jalur dominasi dari busur masukan e_1 ke busur e_u. Jalur dominasi diambil dari PD(G), akan melalui busur-busur yang mendominasi busur e_u
3) Bentuk jalur I(e_u) yaitu sebuah jalur implikasi dari busur e_u ke busur keluaran e_k. Jalur implikasi diambil dari PI(G) akan melalui busur-busur yang melibatkan busur e_u.
4) Bentuk jalur P(e_u) yaitu gabungan jalur dominasi D(e_u) dan jalur implikasi I(e_u)
5) Pada jalur P(e_u) jika terjadi kasus diskontinuitas antara busur e_i dengan busur e_j, dimana $e_i \in$ P(e_u) dan $e_j \in$ P(eu) maka: a) Bentuk sub-DDGraf $G' = \{S', B'\}$; $e_i \neq e_j$; b) Bentuk subjalur $P'(e_u)$ dengan Prosedur $Bentuk\_Satu\_Jalur$ (proses rekursif) menggunakan sebuah busur bebas baru (yang belum digunakan) dengan indeks penghubung tertinggi.
6) Gabungkan subjalur $P'(e_u)$ ke dalam jalur P(e_u) sehingga masalah diskontinuitas terselesaikan [4].

TEKNIK JUMLAH JALUR MINIMUM

Pembentukan satu set jalur dengan teknik jumlah jalur minimum *(the minimum number of paths)* dilakukan dengan cara membentuk setiap jalur yang mencakup sebanyak mungkin busur bebas yang belum tercakup. Teknik ini mensyaratkan agar bilangan kardinalitas set jalur (n) tidak lebih besar daripada bilangan kardinalitas set busur bebas. Penggunaan teknik ini dalam DDGraf yaitu G1 = $\{S, B\}$; $e_1 \neq e_{25}$.
Set busur bebas: BB(G1) = $\{e_2, e_4, e_7, e_8, e_{12}, e_{13}, e_{14}, e_{17}, e_{18}, e_{21}, e_{23}\}$.
Jalankan prosedur $Bentuk\_Satu\_Jalur$ melalui sebuah busur bebas yang belum digunakan dengan indeks penghu-

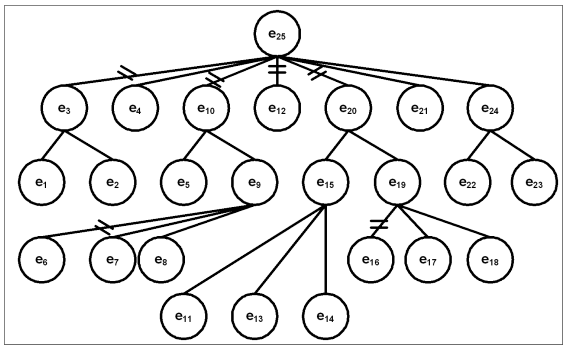


Gambar 6: Pohon Dominasi (PD) G1

bung tertinggi yaitu e_{23} dan urutan hasilnya adalah:
1) Jalur Dominasi: $D(e_{23}) = e_1e_3e_5e_{10}e_{22}e_{23}$
2) Jalur Implikasi: $I(e_{23}) = e_{23}e_{24}e_{25}$
3) Jalur $P(e_{23}) = e_1e_3e_5e_{10}e_{22}e_{23}e_{24}e_{25}$
4) Terjadi kasus diskontinuitas antara busur e_{10} dan busur e_{22}. Jalankan langkah solusi diskontinuitas dengan cara mengambil sebuah busur bebas yang belum digunakan dengan indeks penghubung tertinggi yaitu e_{18}.
5) Terjadi kasus diskontinuitas antara busur e_{10} dan busur e_{18}. Jalankan langkah solusi diskontinuitas dengan cara mengambil sebuah busur bebas yang belum digunakan dengan indeks penghubung tertinggi yaitu e_{14}.
6) Subjalur $P(e_{18}, e_{14}) = e_{10}e_{11}e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}$
7) Gabungkan subjalur $P(e_{18}, e_{14})$ ke dalam jalur $P(e_{23})$ sehingga terbentuk jalur $P(e_{23}, e_{18}, e_{14}) = e_1e_3e_5e_{10}$ $e_{11}e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}e_{23}e_{24}e_{25}$

Demikian seterusnya sehingga Teknik Jumlah Jalur Minimum (JJM) membentuk satu set jalur yang memuat tujuh jalur dan mencakup sebelas busur bebas dalam DDGraf G1 yaitu:
1) Jalur P1 (memuat busur bebas e_{23}, e_{18}, e_{14}): $e_1e_3e_5e_{10}$ $e_{11}e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}e_{23}e_{24}e_{25}$
2) Jalur P2 (memuat busur bebas e_{21}, e_{17}, e_{13}): $e_1e_3e_5e_{10}$ $e_{11}e_{13}e_{15}e_{16}e_{17}e_{19}e_{20}e_{21}e_{25}$
3) Jalur P3 (memuat busur bebas e_{12}): $e_1e_3e_5e_{10}e_{12}e_{22}$ $e_{23}e_{24}e_{25}$
4) Jalur P4 (memuat busur bebas e_8): $e_1e_3e_5e_6e_8e_9e_{10}$ $e_{11}e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}e_{23}e_{24}e_{25}$
5) Jalur P5 (memuat busur bebas e_7): $e_1e_3e_5e_6e_7e_9e_{10}$ $e_{11}e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}e_{23}e_{24}e_{25}$
6) Jalur P6 (memuat busur bebas e_4): $e_1e_3e4e_{25}$
7) Jalur P7 (memuat busur bebas e_2): $e_1e_2e_3e_5e_{10}e_{11}$ $e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}e_{23}e_{24}e_{25}$

Karakteristik teknik ini yaitu membentuk jalur yang memuat busur bebas sebanyak-banyaknya, sehingga ter-



Gambar 7: Pohon Implikasi (PI) G1

bentuk set jalur dengan jumlah jalur paling minimal.

TEKNIK JUMLAH PREDIKAT MINIMUM

Pembentukan satu set jalur dengan teknik jumlah predikat minimum *(less-pred)* dilakukan dengan cara membentuk setiap jalur yang mencakup sesedikit mungkin busur bebas yang belum tercakup. Dengan teknik ini, bilangan kardinalitas set jalur (n) akan sama dengan bilangan kardinalitas set busur bebas. Sebelum menjalankan prosedur $Bentuk\_Satu\_Jalur$, setiap busur bebas akan dihitung nilai Simpul Keputusan (SK). Simpul keputusan adalah simpul dalam DDGraf yang merupakan simpul pencabangan atau busur yang kepalanya merupakan simpul pencabangan. Nilai D untuk sebuah busur bebas e_u adalah bilangan kardinalitas himpunan hasil irisan himpunan busur SK(G1) dengan himpunan busur pada jalur P(e_u).

DDGraf G1 = $S, B; e_1 \neq e_{25}$ Jika dikaitkan dengan titik simpul $\in$ S maka G1 memiliki 10 simpul keputusan dan jika dikaitkan dengan kepala busur $\in$ B maka G1 memiliki 15 simpul keputusan.
Simpul-simpul keputusan adalah:
K(e_1) = K(e_2), K(e_3); K(e_5) = K(e_9), K(e_6), K(e_{10}), K(e_{11}); K(e_{15}) = K(e_{19}), K(e_{16}); K(e_{12}) = K(e_{20}); K(e_{22}) = K(e_{23});

SK (Set Simpul Keputusan) =
$\{e_1, e_2, e_3, e_5, e_9, e_6, e_{10}, e_{11}, e_{15}, e_{19}, e_{16}, e_{12}, e_{20}, e_{22}, e_{23}\}$

Set busur bebas:
BB(G1) = $\{e_2, e_4, e_7, e_8, e_{12}, e_{13}, e_{14}, e_{17}, e_{18}, e_{21}, e_{23}\}$

D(e_u) = $\mid SK(G1) \cap P(e_u) \mid$

Nilai D untuk semua busur bebas G1 adalah:

e	e_2	e_4	e_7	e_8	e_{12}	e_{13}
D	3	2	6	6	5	7

e	e_{14}	e_{17}	e_{18}	e_{21}	e_{23}
D	7	9	9	4	6

Prosedur $Bentuk\_Satu\_Jalur$ dilakukan dengan cara mengambil sebuah busur bebas yang belum tercakup dengan nilai D terendah. Dalam DDGraf G1 busur yang

Tabel 1: Kasus Uji DDGraf

No DDGraf	Σ Busur	Σ Simpul Keputusan	Σ Busur Bebas
1	9	3	4
2	8	3	3
3	11	4	4
4	10	3	4
5	12	4	5
6	11	4	4
7	9	3	3
8	15	5	6
9	25	10	11
10	15	7	6

memenuhi adalah busur e_4. Jika ada lebih dari satu busur bebas dengan nilai D yang sama, maka akan dipilih busur bebas dengan indeks penghubung paling tinggi.

Teknik Jumlah Predikat Minimum (JPM) pada DDGraf G1 akan menghasilkan satu set jalur yang memuat sepuluh jalur dan mencakup 11 busur bebas dalam DDGraf G1 yaitu:
1) Jalur P1 (memuat busur bebas e_4): $e_1e_3e_4e_{25}$
2) Jalur P2 (memuat busur bebas e_2): $e_1e_2e_3e_4e_{25}$
3) Jalur P3 (memuat busur bebas e_{21}): $e_1e_3e_5e_{10}e_{12}e_{21}e_{25}$
4) Jalur P4 (memuat busur bebas e_{12}, e_{23}): $e_1e_3e_5e_{10}e_{12}$ $e_{22}e_{23}e_{24}e_{25}$
5) Jalur P5 (memuat busur bebas e_8): $e_1e_3e_5e_6e_8e_9e_{10}$ $e_{12}e_{21}e_{25}$
6) Jalur P6 (memuat busur bebas e_7): $e_1e_3e_5e_6e_7e_9e_{10}$ $e_{12}e_{21}e_{25}$
7) Jalur P7 (memuat busur bebas e_{14}): $e_1e_3e_5e_{10}e_{11}e_{14}$ $e_{15}e_{20}e_{21}e_{25}$
8) Jalur P8 (memuat busur bebas e_{13}): $e_1e_3e_5e_{10}e_{11}e_{13}$ $e_{15}e_{20}e_{21}e_{25}$
9) Jalur P9 (memuat busur bebas e_{18}): $e_1e_3e_5e_{10}e_{11}e_{14}$ $e_{15}e_{16}e_{18}e_{19}e_{20}e_{21}e_{25}$
10) Jalur P10 (memuat busur bebas e_{17}): $e_1e_3e_5e_{10}e_{11}e_{14}$ $e_{15}e_{16}e_{17}e_{19}e_{20}e_{21}e_{25}$

Karakteristik teknik ini adalah membentuk jalur yang memuat busur bebas seminimal mungkin, sehingga terbentuk set jalur dengan jumlah jalur maksimal sama dengan jumlah busur bebas.

PENGUJIAN TEKNIK PEMBENTUKAN JALUR

Kedua teknik pembentukan jalur telah diujicobakan untuk sepuluh DDGraf [4] dengan jumlah busur berkisar antara 8-25. Tabel 1 memperlihatkan sepuluh DDGraf tersebut. DDGraf-9 yang terdiri dari 25 busur merupakan representasi struktur program fungsi GetOp yang dapat dilihat pada Gambar 4. Fungsi GetOp adalah sebuah fungsi dalam bahasa pemrograman C untuk mengambil *operand* dalam program Kalkulator. DDGraf-9 dapat dilihat pada Gambar 5.

DDGraf-4 yang terdiri dari 10 busur merupakan representasi dari struktur Program SortData. Program ini merupakan sebuah modul dalam bahasa pemrograman Fortran untuk mengambil data pada file masukan.dat dan diurutkan secara *ascending*, serta hasilnya disimpan dalam file yang bernama $keluar.dat$. Modul program SortData dapat dilihat pada Gambar 8. DDGraf-4 sebagai representasi Program SortData dapat dilihat pada Gambar 9.

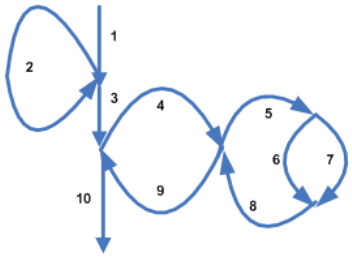
```
real A(100), temp
open(11, file = 'masukan.dat')
open(22, file = 'keluar.dat')
read(11,*) N
do i=1,N
read(11,*) A(i)
end do

close(11)
write(22,*) 'Data yang telah diurutkan:'

do i=1, N
do j=1, N
if(A(j) .lt. A(i)) then
temp = A(i)
A(i) = A(j)
A(j) = temp
end if
end do
write(22,*) i, A(i)
end do

stop 'Sudah.'
end
```

Gambar 8: Program SortData



Gambar 9: Decision-to-Decision Graf-4

Kedua teknik pembentukan set jalur (teknik JJM dan teknik JPM) diujicobakan pada sepuluh DDGraf yang termuat dalam Tabel 1. Hasil pembentukan set jalur untuk sepuluh DDGraf tersebut dapat dilihat pada tabel 2. Dari sepuluh DDGraf yang menjadi kasus uji tersebut, dapat disimpulkan bahwa 1) Jumlah busur yang dimiliki oleh DDGraf tidak mempengaruhi jumlah jalur yang terbentuk; 2) Teknik jumlah predikat minimum umumnya menghasilkan jumlah jalur yang sama banyaknya dengan jumlah busur bebas.

SIMPULAN

Pembentukan set jalur aliran program dapat dilakukan dengan bantuan DDGraf, sebuah graf berarah, dimana sebuah busur dalam DDGraf merepresentasikan satu segmen pernyataan program. Ada dua teknik pembentukan set jalur yaitu teknik jumlah jalur minimum dan teknik jumlah predikat minimum.

Pada setiap teknik yang digunakan untuk membentuk suatu jalur, dapat menghasilkan jalur yang tidak fisibel *(infeasible path)* [5]. Jalur tidak fisibel artinya jalur yang tidak mungkin dilalui, dalam program berarti jalur yang tidak mungkin dapat dieksekusi.

Seluruh jalur hasil teknik JJM pada DDGraf G1 bersifat fisibel. Sedangkan hasil teknik JPM pada DDGraf G1

Tabel 2: Kasus Uji Teknik Pembentukan Jalur

No	Σ Busur	Jumlah Jalur Teknik JJM	Jumlah Jalur Teknik JPM
1	9	3	4
2	8	3	3
3	11	4	4
4	10	4	4
5	12	3	5
6	11	4	4
7	9	3	3
8	15	4	6
9	25	7	10
10	15	4	6

memperlihatkan jalur P5, P7, P9, dan P10 tidak fisibel. Hal ini akibat dari: busur e_8 tidak kompatibel dengan e_{13}, e_{17}, e_{21}; busur e_{14} tidak kompatibel dengan e_{17}, e_{21}; dan busur e_{18} tidak kompatibel dengan e_{21};

Sifat tidak saling kompatibel tersebut dapat dengan mudah diletakan pada catatan *(account)* dengan cara mengatur konstrain yang tepat pada suatu set busur bebas yang belum digunakan. Dengan teknik JPM pada DDGraf G1, ketika membangun satu jalur yang melalui busur e_7, prosedur pemilihan busur bebas lainnya tidak boleh memilih busur bebas e_{12}, e_{16}, e_{20}. Teknik JPM untuk DDGraf G1 disertai catatan tentang sifat tidak saling kompatibel pada beberapa busur akan menghasilkan sepuluh jalur fisibel sebagai berikut:

1) Jalur P1 (memuat busur bebas e_4): $e_1e_3e_4e_{25}$
2) Jalur P2 (memuat busur bebas e_2): $e_1e_2e_3e_4e_{25}$
3) Jalur P3 (memuat busur bebas e_{21}): $e_1e_3e_5e_{10}e_{12}e_{21}e_{25}$
4) Jalur P4 (memuat busur bebas e_{12}, e_{23}): $e_1e_3e_5e_{10}$ $e_{12}e_{22}e_{23}e_{24}e_{25}$
5) Jalur P5 (memuat busur bebas e_8): $e_1e_3e_5e_6e_8e_9$ $e_{10}e_{12}e_{22}e_{23}e_{24}e_{25}$
6) Jalur P6 (memuat busur bebas e_7): $e_1e_3e_5e_6e_7e_9$ $e_{10}e_{12}e_{21}e_{25}$
7) Jalur P7 (memuat busur bebas e_{14}): $e_1e_3e_5e_{10}e_{11}$ $e_{14}e_{15}e_{20}e_{22}e_{23}e_{24}e_{25}$
8) Jalur P8 (memuat busur bebas e_{13}): $e_1e_3e_5e_{10}e_{11}$ $e_{13}e_{15}e_{20}e_{21}e_{25}$
9) Jalur P9 (memuat busur bebas e_{18}): $e_1e_3e_5e_{10}e_{11}$ $e_{14}e_{15}e_{16}e_{18}e_{19}e_{20}e_{22}e_{23}e_{24}e_{25}$
10) Jalur P10 (memuat busur bebas e_{17}): $e_1e_3e_5e_{10}e_{11}$ $e_{14}e_{15}e_{16}e_{17}e_{19}e_{20}e_{21}e_{25}$

Dari hasil analisis ini, penelitian dilanjutkan dengan membangun sebuah perangkat lunak yang mampu membangkitkan secara otomatis satu set jalur. Data masukan berupa sebuah DDGraf (yang merepresentasikan struktur sebuah modul program) dan data keluaran berupa sebuah set jalur yang terbentuk melalui kedua teknik, untuk digunakan sebagai set jalur eksekusi program [4].

DAFTAR PUSTAKA

[1] Schach, S.R.: *Object Oriented and Classical Software Engineering*. 5th edition edn. McGraw Hill Co. (2002)

[2] R.Woodward, M., Hedley, D., M.A.Hennell: *Experience with Path Analysis and Testing of Programs*. In: IEEE Transactions on Software Engineering. Volume SE-6. (1980) 278–286

[3] S.Hecht, M.: *Flow Analysis of Computer Programs*. (1977)

[4] de Lima, R., S.Oerip, S.: *Pembangkit Otomatis Cakupan Jalur Berbasis Graf Aliran Kendali Bagi Pengujian Cabang*. Master's thesis, Institut Teknologi Bandung (1997)

[5] Yates, D.F., Malevris, N.: *Reducing the effects of Infeasible Path in Branch Testing*. In: ACM SigSoft Software Engineerings Notes. Volume 14. (1989) 48–54