PERBANDINGAN ALGORITMA FIBONACCI SEARCH DAN GOLDEN SECTION SEARCH DALAM OPTIMISASI FUNGSI

Haeni Budiati
Ilmu Komputer, FMIPA, UKRIM

Abstraksi

Fibonacci Search dan Golden Section Search merupakan dua algoritma untuk mencari titik optimum fungsi, yang keduanya mempunyai kemiripan, sebab sama-sama menggunakan deret bilangan Fibonasi. Hanya saja pada Golden Section search, bilangan Fibonasi tidak digunakan secara langsung. Tetapi dengan menggunakan rumusan yang sebenarnya berasal dari deret Fibonasi juga. Perbandingan meliputi banyaknya iterasi, flop dan waktu.

Kata Kunci:Optimasi, grafik,

1. PENDAHULUAN

Kadangkala seseorang diperhadapkan pada masalah dimana ia harus mengoptimalkan nilai fungsi tersebut yaitu dengan menentukan nilai variabelnya yang membuat nilai fungsi tersebut menjadi optimum (maksimum atau minimum), atau dengan kata lain berarti mencari titik optimum fungsi tersebut. Masalah seperti ini, dalam bahasa inggris disebut "optimization" dan disini penulis menerjemahkannya sebagai "optimisasi", untuk membedakan dengan istilah umum yaitu optimasi yang berarti mengoptimalkan. Jika fungsi yang dioptimumkan tersebut merupakan fungsi linier, yaitu fungsi yang jika digambarkan secara geometris adalah berupa garis lurus, maka untuk mencari titik optimumnya tidaklah sulit, yaitu dengan membuat sketsa grafik fungsi yang bersangkutan, jika grafik naik maka titik optimum adalah batas x maksimum dan sebaliknya. Namun jika fungsi tersebut tidak linier, maka persoalannya akan menjadi lebih sulit.

Salah satu fungsi yang tidak linier adalah fungsi Polinom biasa, yaitu fungsi suku banyak dengan rumus $f(x) = anxn + an\text{-}1xn\text{-}1 + \ldots + a1x + a0$, dimana an, an-1,

..., a0 adalah suatu konstanta dan n adalah bilangan cacah. Jika an □ 0, maka polinom tersebut disebut berderajat n. Dengan kata lain bahwa derajat suatu polinom adalah pangkat tertingginya.

TUJUAN

membandingkan dua algoritma, yaitu Fibonasi Search dan Golden Section Search, sehingga dapat diketahui algoritma mana yang lebih baik, baik dalam hal kecepatan banyaknya flop dan iterasi, sehingga dapat diketahui apakah ada hubungannya atau tidak.

2. LANDASAN TEORI

1. Titik Optimum fungsi

Titik optimum fungsi yang dimaksudkan adalah sama dengan yang sering disebut dengan titik ekstrim, yaitu suatu titik yang menunjukkan nilai-nilai variabel yang membuat nilai fungsi menjadi optimum (maksimum atau minimum). Misalkan suatu fungsi f(x) bernilai maksimum 21 yang dicapai bila x = 3, maka dapat dikatakan bahwa titik optimum fungsi adalah (3,21) atau titik optimum terjadi pada x = 3. Walaupun demikian, sering juga dikatakan bahwa nilai seperti x = 3 tersebut disebut titik optimum dan dilambangkan dengan x*. Jadi dalam hal ini x* =3.

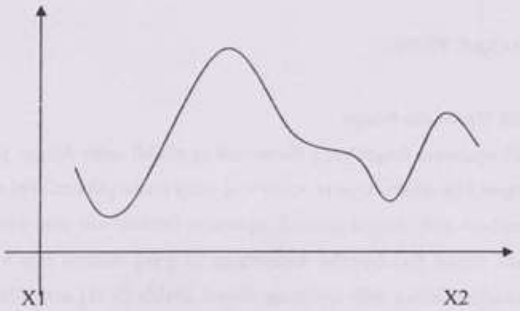
2. Metode Optimisasi fungsi

Pencarian titik optimum suatu fungsi dapat dilakukan dengan dua cara. Pertama, dengan menyelidiki turunan dari fungsi yang bersangkutan. Dan dengan cara seperti ini, penulis menyebutnya secara analitik, karena melibatkan perhitungan-perhitungan analitik turunan.. Kedua, sama sekali tidak menggunakan konsep turunan dan deferensial, tetapi dengan mencari dua titik yang mempunyai kemungkinan menjadi titik optimum, kemudian masing-masing titik diselidiki, sehingga hanya tertinggal satu titik saja yang mempunyai kemungkinan menjadi

x*. Kemudian dari satu titik tersebut, dengan menggunakan suatu prosedur tertentu, dua titik berikutnya.

A. Secara Analitik

Secara analitik, pencarian titik optimum suatu fungsi, dilakukan dengan menyelidiki turunan fungsi yang bersangkutan, yaitu dengan mencari akar turunan fungsi tersebut. Sebab secara geometris, turunan fungsi di suatu titik tertentu, merupakan gradien dari grafik fungsi tersebut. Itu berarti jika gradien di suatu titik tertentu pada fungsi sama dengan nol, maka titik tersebut mempunyai kemungkinan untuk menjadi titik optimum



Gambar . Gambaran geometris titik optimum

Titik-titik A,B,C dan D adalah titik-titik yang mempunyai kemungkinan menjadi titik optimum, tergantung pada jenis titik optimum (maksimum atau minimum) dan juga batas nilai (interval) x.

B. Secara Iterasi

Pencarian titik optimum fungsi dilakukan dengan cara menyelidiki titik-titik tertentu secara iterasi atau berulang-ulang hingga didapat titik optimumnya. Metode inilah yang akan dibandingkan dalam skripsi ini, yaitu metode Fibonacci Search dan Golden Section Search. Salah satu kelebihan metode iterasi dibandingkan dengan metode analitik adalah dalam hal menentukan jenis titik optimum. Kalau secara analitik, untuk suatu fungsi tertentu mungkin hanya bisa ditentukan salah satu jenis titik

optimum saja, maksimum atau minimum. Sedangkan pada metode iterasi, baik titik maksimum atau minimum bisa ditentukan untuk satu fungsi yang sama.

3. Fibonacci Search

Fibonacci Search atau dapat diterjemahkan sebagai "Pencarian Fibonasi" adalah suatu teknik pencarian yang dapat digunakan untuk mendapatkan titik optimum pada sebarang fungsi, dengan syarat bahwa fungsi tersebut unimodal dan univariate. Artinya fungsi hanya memiliki satu titik optimum dan satu variabel. Fibonasi dipakai disini, karena memang teknik ini berhubungan dengan (atau menggunakan) deretan bilangan yang disebut deret Fibonasi, yaitu deretan bilangan yang mempunyai rumus :

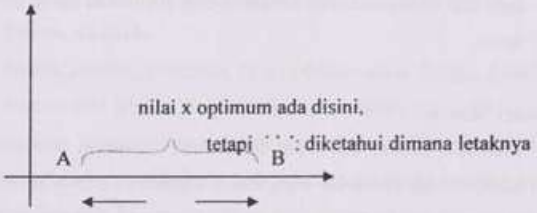
$Fn = Fn\text{-}1 + Fn\text{-}2 \quad n = 2,3,4,\ldots$

$F0 = 1$ dan $F1 = 1$

Identifier	Urutan	Bilangani Fibonasi
F0	0	1
F1	1	1
F2	2	2
F3	3	3
F4	4	5
F5	5	8
F6	6	13
F7	7	21
F8	8	34
F9	9	55
F10	10	89
dst.	dst.	dst.

Tabel . Bilangan Fibonasi

Untuk memahami cara kerja teknik ini, anggap bahwa suatu fungsi mempunyai interval (jangkauan) untuk nilai x antara titik A dan B



Gambar . Interval ketidakpastian

Nilai X optimum, dipastikan berada diantara batas kiri A dan batas kanan B dan bisa kita tuliskan sebagai X*. Untuk mencari X* tersebut dengan menggunakan teknik ini, secara lengkap dapat diberikan algoritma sebagai berikut:

Langkah 1. Tentukan interval ketidakpastian, yaitu selang antara batas kanan A dan batas kiri B dimana X* terletak diantaranya.

Langkah 2. Tentukan faktor resolusi atau toleransi ε, yaitu nilai yang diijinkan untuk memisahkan jarak antara titik A dan B pada iterasi terakhir, dimana X* terletak diantaranya

Langkah 3. Tentukan bilangan Fibonasi F_N sedemikian rupa sehingga memenuhi $F_N.(\varepsilon) \geq B - A$. Setelah F_N ditemukan, nilai ε berubah menjadi $e = (B - A) / F_N$, yang nilainya lebih kecil dan mendekati ε atau bahkan sama dengan ε. Nilai e inilah yang dipakai sebagai toleransi. Dari sini besarnya nilai N, yaitu banyak bilangan Fibonasi juga dapat diketahui.

Langkah 4. Lakukan inisialisai atau keadaan mula-mula, yaitu

- Tentukan $L = F_{N-1} * e$
- Tentukan $X1 = A + L$
- Tentukan $X2 = B - L$

Langkah 5. Lakukan iterasi sampai dengan $F_{N-1} = 2$ sebagai berikut

- Lakukan Eliminasi, yaitu menghilangkan daerah atau interval yang dipastikan X* tidak berada disitu.
- Dari langkah Eliminasi didapat titik A dan B yang baru serta satu titik X yang dianggap sebagai X* sementara.
- Hitung $L = F_{N-1} * e$
- Hitung $Xn = A + L$ atau $Xn = B - L$ tergantung daerah yang dieliminasi sebelumnya.

Dari sini didapat X* dan Xn yang dianggap sebagai X1 dan X2 yang baru.

Langkah 6. Tentukan X*, yaitu X yang tersisa diantara A dan B.

Untuk memperjelas algoritma diatas, diberikan satu contoh sebagai berikut.

Maksimumkan fungsi $f(x) = 6x^3 + 10x^2 - 4x + 15$

dengan interval ketidakpastian $-50 \geq x \geq 50$

dan toleransi = 0.5

Penyelesaian.

$A = -50$

$B = 50$

$$F_N(\varepsilon) \geq B - A \Leftrightarrow 0.5\, F_N \geq 100$$

Untuk menentukan F_N hilangkan pertidaksamaan diatas sehingga tinggal persamaan saja

$$0.5\, F_N = 100$$
$$F_N = 200$$

bilangani Fibonasi yang ≥ 200 adalah 233

Jadi ubahkan $F_N = 233$

dan bilangan itu adalah deret ke 12 dari deret Fibonasi, sehingga N = 12

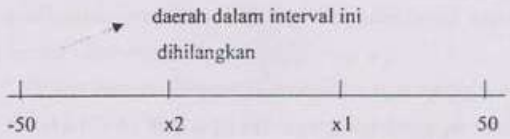
Itu berarti ada 9 kali iterasi (termasuk keadaan awal),
sebab N bergerak turun dari N-1 hingga 2.

Kemudian hitung $e = (B-A)/ F_N$
$= 100/ 233$
$= 0.429$

Keadaan mula-mula/ Inisialisi

$L = F_{N-1} * e = 144 * 0.429 = 61.78$
$X_1 = -50 + 61.78 = 11.8$
$X_2 = 50 - 61.78 = -11.8$
$f(x_1) = 11225.4$
$f(x_2) = -8403.5$

Oleh karena f(x1) > f(x2), maka interval antara A dan x2 dieliminasi.
Sketsa geometrisnya :



kemudian dimulai langkah iterasi
kurangi N sehingga L = F10 * 0.429
Dari A dan B yang baru, tentukan kembali X1 dan X2
kemudian eliminasi lagi

Dan langkah ini dilakukan terus menerus sampai N = 2.

iterasi n	A	B	X_1	$f(X_1)$	X_2	$f(X_2)$
0	-50	50	11.80	11225.4	-11.8	-8409.4
1	-11.8	50	11.80	11225.4	26.39	117210
2	11.80	50	26.39	117210	35.40	278755
3	26.39	50	35.40	278755	40.98	429787
4	35.40	50	40.98	429787	44.42	545470
5	40.98	50	44.42	545470	46.56	627373
6	44.42	50	46.56	627373	47.85	680242
7	46.56	50	47.85	680242	48.71	717088
8	47.85	50	48.71	707188	49.14	736000
9	48.71	50	49.57	755241	49.57	755241

Tabel Hasil contoh soal

Dari iterasi terakhir didapat

$X^* = 49.57$ dan $f(X^*) = 755421$

dengan e = 0.429

Jadi, f(x) maksimum pada x = 49.57

4. Golden Section Search

Golden Section Search merupakan suatu teknik/ metode optimisasi yang merupakan modifikasi dari Fibonacci search. Di dalam teknik ini, ada salah satu unsur utama yang terdapat dalam metode Fibonacci Search dihilangkan. Unsur yang dimaksudkan adalah penentuan F_N yang merupakan bilangan Fibonasi tertentu dalam deret ke N.

Nama Golden Section, sebenarnya merupakan suatu perbandingan. Jadi merupakan suatu bilangan real atau konstanta, yaitu 0.618.

Nilai konstanta ini didapat dari perbandingan atau pembagian antara FN-1 dan FN dalam deret Fibonasi, yaitu

$$\lim_{N\to\infty} \frac{F_{N-1}}{F_N} = \left(\frac{1+\sqrt{5}}{2}\right)^{-1} = 0.6180...$$

Prosedur pencarian titik optimum dengan metode Golden Section Search dapat diuraikan sebagai berikut.

Langkah 1. Tentukan A,B dan toleransi e seperti pada Algoritma Fibonacci Search

Langkah 2. Lakukan langkah-langkah inisialisasi atau keadaan mula-mula sebagai beriklut :

- L = 0.618 * (B-A)
- X1 = A + L
- X2 = B – L

Langkah 3. Selama L ≤ e lakukan iterasi sebagai berikut :

- Lakukan Eliminasi, seperti pada algoritma Fibonacci Search sehingga sebagai hasilnya didapat A, B dan X* Sementara.
- Jika L ≤ e kerjakan langkah 4 dan selesai.
- Hitung kmbali L = 0.618 * (B-A) sesuai A dan B yang baru.
- Hitung Xn = A + L atau Xn = B-L sesuai daerah yang dieliminasi sebelumnya.

Dari hasil ini didapat X* dan Xn sebagai X1 dan X2 yang baru, kemudian lakukan langkah eliminasi lagi sebelum iterasi selanjutnya dilakukan.

Langkah 4. Tentukan X optimum (X*), yaitu X yang tersisa dalam selang A,B pada iterasi yang terakhir.

3. ANTARMUKA PROGRAM

1) Menu Utama

Menu utama adalah tampilan awal progam, yang terdiri dari dua pilihan "Optimisasi" dan "Perbandingan"

Gambar . Form dengan menu utama

2) Program Optimisasi

Program optimisasi merupakan program untuk mencari titik optimum fungsi, sehingga juga bisa dianggap sebagai program bantu untuk mencari titik optimum fungsi. Program ini dapat dijalankan dengan memilih pilihan program optimisasi pada menu utama. Secara default, pilihan inilah yang

dipilih, sehingga jika user tidak menetapkan pilihan perbandingan, berarti program optimisasi yang akan dijalankan.

Gambar . Input Program Optimisasi

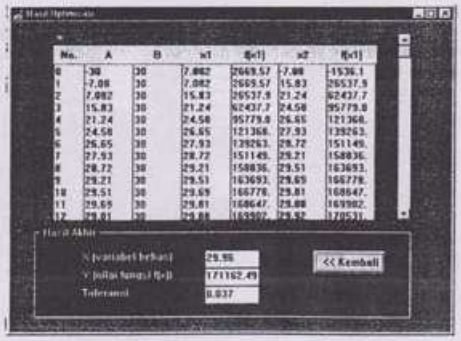
3) Program Perbandingan

Program perbandingan, akan membandingkan dua metode optimisasi fungsi, yaitu Fibonacci dan Golden Section Search. Program ini dijalankan dengan memilih pilihan perbandingan dari menu utama, yang selanjutnya akan muncul form input untuk perbandingan

Gambar . Input Program Perbandingan

4) Output Program Optimisasi

Sesuai dengan rancangan sebelumnya, output program optimisasi merupakanhasil proses dari program optimisasi. Disini diberikan contoh hasil optimisasi untuk fungsi $f(x) = 6x^3 + 10x^2 - 4x + 15$ dengan batas kiri –30,



batas kanan 30 dan batas toleransi maksimum 0.05 serta metode optimisasi ibonacci Search untuk memaksimumkan.

Gambar . Output Program Optimisasi

5) Output Program Perbandingan

Output program perbandingan merupakan tampilan dari program perbandingan atau pilihan perbandingan pada menu utama.

Pangkat	Fibonacci Search			Golden Section Search		
	Flop	Iterasi	Waktu (mdt)	Flop	Iterasi	Waktu (mdt)
2	152	15.25	0.6	147	16.57	0.5
3	219	15.3	0.8	220	16.55	1.1
4	285	15.19	0.5	292	16.58	1.1
5	350	15.14	1.1	363	16.51	1.1
6	419	15.21	1.1	436	16.55	1.7
7	488	15.29	1.6	510	16.62	1.7
8	556	15.31	1.6	581	16.57	2.2
9	622	15.29	2.2	655	16.62	2.2
10	692	15.36	2.2	731	16.7	2.8

Gambar . Output Program Perbandingan

Hasil yang diberikan oleh tampilan tersebut merupakan hasil rata-rata baik flop, iterasi maupun waktu untuk N kali program optimisasi pada masing-masing derajat, dimana N adalah jumlah percobaan yang diinputkan sebelumnya. Tampilan diatas adalah unttuk N = 100 dan toleransi 0.05.

Jumlah Percobaan

Jumlah percobaan yang dimaksudkan disini adalah jumlah pemanggilan terhadap tiap procedure optimisasi, baik Fibonacci Search ataupun Golden Section Search. Untuk dapat melihat hasil perbandingan dengan baik, disini penulis memberikan input 100 percobaan. Pemilihan nilai 100, dimaksudkan untuk dapat melihat waktu proses untuk pangkat yang berbeda-beda, sebab untuk jumlah percobaan kurang dari 100, hasilnya tidak terlihat karena kecepatan prosesnya yang kurang dari 0 milidetik.

Tabel . Hasil perbandingan untuk 100 percobaan

Pangkat	Fibonacci Search			Golden Section Search		
	flop	iterasi	Waktu	flop	iterasi	waktu
2	152	15.27	0.5	147	16.55	0.6
3	218	15.22	0.5	219	16.58	0.6
4	290	15.49	1	295	16.8	1.1
5	356	15.4	1.1	368	16.75	1.1
6	418	15.19	1.1	435	16.52	1.7
7	488	15.3	1.6	508	16.57	1.7
8	548	15.09	2.2	578	16.47	1.6
9	621	15.27	2.2	654	16.61	2.2
10	694	15.41	2.2	731	16.7	2.7

Tabel . Hasil perbandingan untuk 200 percobaan

Pangkat	Fibonacci Search			Golden Section Search		
	flop	iterasi	Waktu	flop	iterasi	waktu
2	152	15.28	0.55	147	16.57	0.55
3	219	15.29	0.55	219	16.56	0.8
4	288	15.37	0.85	295	15.76	1.1
5	354	15.33	1.1	366	15.66	1.1
6	422	15.34	1.35	439	15.70	1.65
7	485	15.19	1.35	507	15.50	1.95
8	553	15.24	1.65	581	15.58	2.2
9	619	15.2	2.2	653	15.57	2.2
10	687	15.23	2.45	724	15.53	2.75

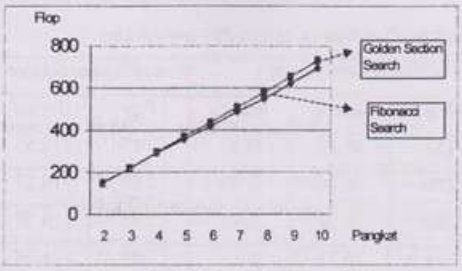
Tabel . Hasil perbandingan untuk 300 percobaan

Pangkat	Fibonacci Search			Golden Section Search		
	flop	iterasi	Waktu	flop	iterasi	waktu
2	152	15.23	0.57	147	16.55	0.53
3	219	15.26	0.73	219	16.57	0.73
4	286	15.29	0.93	292	16.59	0.9
5	354	15.32	1.1	365	16.64	1.3
6	419	15.22	1.27	437	16.58	1.47
7	490	15.36	1.63	510	16.63	1.67
8	554	15.26	1.83	580	16.55	2
9	620	15.21	2.2	652	16.53	2.4
10	687	15.23	2.53	727	16.60	2.77

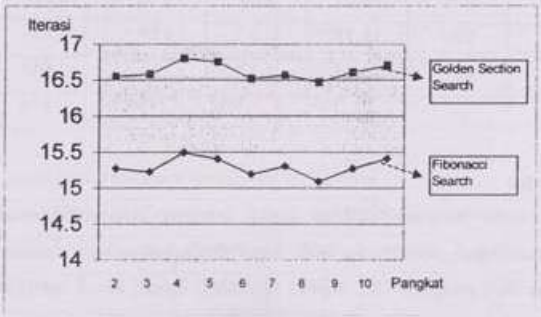
5. Grafik

Dalam program, fasilitas grafik memang tidak disediakan, namun hasil perbandingan yang berupa tabel dapat digunakan sebagai data untuk selanjutnya digunakan program lain, seperti Microsoft Excel, untuk mengubahnya menjadi berbentuk grafik.

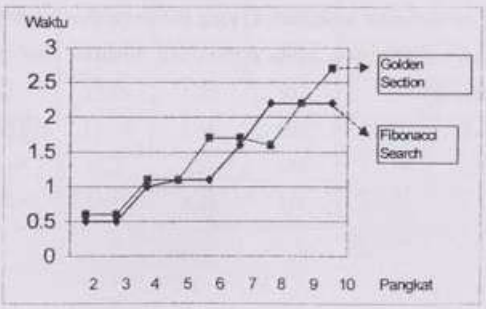
Ada tiga grafik yang dapat dibuat, yaitu perbandingan flop, iterasi dan waktu. Dengan melihat tabel sebelumnya, yang menampakkan kemiripan untuk jumlah percobaan kelipatan 100, maka grafik yang disajikan disini adalah untuk 100 percobaan saja.



Gambar . Grafik Perbandingan Flop



Gambar . Grafik Perbandingan Iterasi



Gambar 4.8 Grafik Perbandingan Waktu

5. Kesimpulan

Berdasarkan apa yang telah dibahas pada bab-bab sebelumnya, selanjutnya dapat ditarik beberapa kesimpulan :

1. Semakin besar derajat suatu Polinom, semakin besar pula flop yang ada dalam algoritmanya, tetapi belum tentu demikian untuk banyak iterasi dalam program optimisasinya.
2. Dalam tiap kali percobaan, pada umumnya semakin tinggi pangkat suatu polinom, semakin besar perbedaan flop antara kedua algoritma.
3. Untuk algoritma Fibonacci Search, banyaknya iterasi selalu lebih sedikit dari pada Golden Section Search.
4. Untuk kecepatan atau waktu proses, kebanyakan yang terjadi adalah semakin tinggi pangkat, maka semakin lama proses. Dan ini mudah untuk dipahami. Tetapi beberapa percobaan tidak mendukung kesimpulan ini. Jadi disini dapat disimpulkan bahwa untuk pengukuran waktu menggunakan sistem operasi Windows adalah kurang efektif, karena sistem operasi ini bersifat multitasking, sehingga bisa dipengaruhi oleh proses-proses yang lain meskipun semua program yang lain sudah ditutup.

DAFTAR PUSTAKA

Divisi Penelitian dan Pengembangan LPKBM MADCOMS – MADIUN. *Panduan Lengkap pemrograman Borland Delphi 5*. Penerbit ANDI Yogyakarta, 2001.

R. Bronson. *Seri Buku Schaum's Teori dan Soal-soal Operation Research*. Penerbit Erlangga, 1994 . (Terjemahan)

Ravindran, Philips, Solberg. *Operation Research Principles and Practice*. John Wiley and Sons Inc. 1987