ORIGINAL ARTICLE

Optimasi Fungsi Tak Berkendala Menggunakan Algoritma Genetika Terdistribusi dengan Pengkodean Real

Wayan Firdaus Mahmudy

Program Studi Ilmu Komputer, Fakultas MIPA, Universitas Brawijaya, Malang, Indonesia
wayanfm@ub.ac.id

Abstrak. *Algoritma Genetika dengan pengkodean biner biasa digunakan dalam masalah optimasi fungsi. Kelemahan pengkodean biner adalah jika range solusi berada daerah kontiyu. Pengkodean real bisa menyelesaikan masalah ini. Masalah berikutnya yang timbul adalah konvergensi dini yang terjadi karena populasi solusi terjebak dalam optimum lokal. Makalah ini membahas penggunaan algoritma genetika terdistribusi yang menggunakan beberapa subpopulasi dan tiap subpopulasi menggunakan operator genetika yang berbeda. Operator migrasi digunakan untuk memindahkan individu ke dalam subpopulasi lain. Dari hasil percobaan didapatkan populasi yang lebih beragam sehingga memperluas area pencarian solusi. Area pencarian solusi yang lebih luas meningkatkan akurasi hasil optimasi.*

Kata kunci: optimasi fungsi, algoritma genetika terdistribusi, operator genetika, migrasi

1. Pendahuluan

Optimasi merupakan masalah memaksimalkan atau meminimalkan suatu fungsi dengan kendala atau tanpa kendala. Dalam banyak kasus tidak mudah menyelesaikan masalah optimasi dengan fungsi tujuan non linier secara eksak. Pada pencarian solusi suatu masalah kadang-kadang dibutuhkan formulasi matematika yang kompleks untuk memberikan solusi yang pasti. Solusi optimum mungkin dapat diperoleh tetapi memerlukan proses perhitungan yang panjang dan tidak praktis. Untuk mengatasi kasus khusus seperti di atas dapat digunakan metode heuristik, yaitu suatu metode pencarian yang didasarkan atas intuisi atau aturan-aturan empiris untuk memperoleh solusi yang lebih baik daripada solusi yang telah dicapai sebelumnya [1]. Metode heuristik tidak selalu menghasilkan solusi terbaik tetapi jika dirancang dengan baik akan menghasilkan solusi yang mendekati optimum dalam waktu yang cepat. Algoritma genetika (*genetic algorithms / GAs*) adalah salah satu cabang evolutionary algorithms, yaitu suatu teknik optimasi yang didasarkan pada genetika alami. Dalam algoritma genetika untuk menghasilkan suatu solusi optimal, proses pencarian dilakukan di antara sejumlah alternatif titik optimal berdasarkan fungsi probabilistik [2].

Algoritma Genetika dengan pengkodean biner biasa digunakan dalam masalah optimasi fungsi. Kelemahan pengkodean biner adalah jika range solusi berada dalam daerah kontiyu. Pengkodean real (*real-coded genetic algorithms, RCGA*) bisa menyelesaikan masalah ini [3]. RCGA murni memberikan hasil kurang optimum pada pencarian dalam area yang kompleks. Konvergensi dini sering terjadi karena populasi solusi terjebak dalam optimum lokal. Algoritma genetika terdistribusi (*distributed genetic algorithms, DGA*) menggunakan beberapa subpopulasi dan tiap subpopulasi menggunakan operator

Mahmudy, WF 2009, 'Optimasi fungsi tak berkendala menggunakan algoritma genetika terdistribusi dengan pengkodean real', *Seminar Nasional Basic Science VI* FMIPA, Universitas Brawijaya, Malang, 21 February.

genetika (tukar silang/*crossover* dan mutasi) yang berbeda. Operator migrasi digunakan untuk memindahkan individu dari satu subpopulasi ke dalam subpopulasi lain.

2. Algoritma Genetika

Apabila dibandingkan dengan prosedur pencarian dan optimasi biasa, algoritma genetika berbeda dalam beberapa hal sebagai berikut [2]:

- Manipulasi dilakukan terhadap kode dari himpunan parameter, tidak secara langsung terhadap parameternya sendiri.
- Proses pencarian dilakukan dari suatu titik populasi, tidak dari satu titik saja.
- Proses pencarian menggunakan informasi dari fungsi tujuan.
- Pencariannya menggunakan stochastic operators yang bersifat probabilistik, tidak menggunakan aturan deterministik.

Masalah utama pada algoritma genetika adalah bagaimana memetakan satu masalah menjadi satu string kromosom. Siklus perkembangan algoritma genetik diawali dengan pembuatan himpunan solusi baru (initialization) yang terdiri atas sejumlah string kromosom dan ditempatkan pada penampungan populasi. Kemudian dilakukan proses reproduksi dengan memilih individu-individu yang akan dikembangbiakkan. Penggunaan operator-operator genetik seperti tukar silang (*crossover*) dan mutasi (*mutation*) terhadap individu-individu yang terpilih dalam penampungan individu akan menghasilkan keturunan atau generasi baru. Setelah proses evaluasi untuk perbaikan populasi, maka generasi-generasi baru ini akan menggantikan himpunan populasi asal. Siklus ini akan berlangsung berulang kali sampai tidak dihasilkan perbaikan keturunan, atau sampai kriteria optimum ditemukan.

Apabila P(t) dan C(t) merupakan parents dan children pada generasi ke-t, maka struktur umum algoritma genetika dapat dideskripsikan sebagai berikut:

```
procedure AlgoritmaGenetika
begin
  t = 0
  inisialisasi P(t)
  while (bukan kondisi berhenti) do
    evaluasi P(t)
    seleksi P(t)
    reproduksi C(t) dari P(t)
    bentuk P(t+1) dari P(t) dan C(t) terbaik
    t = t + 1
  end while
end
```

Inisialisasi Populasi

Siklus perkembangan algoritma genetika diawali dengan pembuatan himpunan solusi baru (initialization) yang terdiri atas sejumlah string kromosom dan ditempatkan pada penampungan populasi. Populasi awal sebagai daerah pencarian solusi optimal dibangkitkan secara acak. Representasi kromosom diperlukan untuk menjelaskan setiap individu dalam populasi. Setiap individu atau kromosom tersusun atas urutan gen dari suatu alphabet. Suatu alfabet dapat terdiri dari digit biner (0 dan 1), floating point, integer, simbol-simbol (seperti A, B, C), matriks, dan lain sebagainya [4].

Pada makalah ini digunakan representasi kromosom bilangan pecahan (real). Panjang string kromosom tergantung pada banyaknya variabel bebas/keputusan (x) yang digunakan.

Evaluasi

Untuk mengevaluasi nilai fitness dari kromosom dilakukan langkah-langkah berikut

– Ambil nilai real dari tiap individu dalampopulasi. $x^k = \left(x_1^k, ..., x_n^k\right)$; k=1,2,...,ukuran_populasi, n=banyaknya_variabel_keputusan.

– Evaluasi nilai fungsi tujuan $f(x^k)$

– Konversi nilai fungsi tujuan menjadi nilai *fitness*. Untuk masalah maksimasi nilai *fitness*= $f(x^k)$. Untuk masalah minimasi nilai *fitness*=z - $f(x^k)$; z adalah sembarang bilangan real.

Seleksi

Seleksi digunakan untuk memilih dua individu/kromosom sebagai induk untuk anak pada generasi berikutnya. Pada makalah ini digunakan metode pemilihan seragam, artinya setiap individu dalam populasi memiliki peluang yang sama untuk terpilih. Metode ini dipilih untuk menghemat waktu yang digunakan pada komputasi nilai probabilitas kumulatif yang digunakan pada metode seleksi *roulette-wheel*.

Crossover

Individu baru (*offspring*) terbentuk dari proses crossover. Pada makalah ini digunakan dua metode crossover yaitu *flat crossover* [5] dan *extended intermediate crossover* [6]. Misalkan $C_1 = (c_1^1 \ldots c_n^1)$ dan $C_2 = (c_1^2 \ldots c_n^2)$ adalah dua kromosom yang telah diseleksi untuk melakukan crossover, maka perbedaan dua metode crossover yang digunakan adalah sebagai berikut

Flat Crossover
Offspring H = $(h_1 \ldots, h_i, \ldots h_n)$ dibangkitkan dan h_i secara acak dipilih pada interval $[c_i^1, c_i^2]$.

Extended Intermediate Crossover
Offspring H = $(h_1 \ldots, h_i, \ldots h_n)$ dibangkitkan dan $h_i = c_i^1 + \alpha_i (c_i^2 - c_i^1)$, α_i dipilih secara acak pada interval [-0.25, 1.25].

Mutasi

Mutasi digunakan untuk mengexploitasi solusi pada area lokal. Pada makalah ini digunakan *random mutation*. $h_i' = h_i \pm a\,(max_i - min_i)$, a bilangan random pada interval [0, 1], max_i nilai maksimum dari x_i, min_i nilai minimum dari x_i.

3. Metode

Pada makalah ini digunakan dua fungsi untuk ujicoba perangkat lunak yang dibangun berdasarkan algoritma genetika. Fungsi-fungsi yang digunakan dalam pengujian adalah [3]

Fungsi	interval	optimum
$f_1(\vec{x}) = \sum_{i=1}^{n} x_i^2, \quad n=4$	$-5.12 \le x \le 5.12$	$f_1^* = (0,...,0) = 0$
$f_2(\vec{x}) = \sum_{i=1}^{n-1} \left(100.\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right), \quad n=4$	$-5.12 \le x \le 5.12$	$f_2^* = (1,...,1) = 0$

Perangkat lunak dijalankan sampai generasi 5000 dengan rancangan percobaan sebagai sebagai berikut:

Percobaan	Keterangan
C1	menggunakan *flat crossover*, populasi memuat 20 individu
C2	menggunakan *extended intermediate crossover*, populasi memuat 20 individu
DGA1	mengunakan 2 subpopulasi, subpopulasi pertama menggunakan *flat crossover*, subpopulasi pertama menggunakan *extended intermediate crossover*, masing-masing memuat 10 individu, pada generasi kelipatan 25 dipilih satu individu dari setiap subpopulasi kemudian ditukarkan (migrasi)
DGA2	sama seperti DGA1, migrasi dilakukan pada generasi kelipatan 50

4. Hasil dan Pembahasan

Untuk mengetahui keoptimalan hasil DGA dibandingkan GAs, masing-masing skenario percobaan dijalankan 5 kali. Tabel 1 dan Tabel 2 menunjukkan rata-rata nilai fungsi f_1 dan f_2 pada beberapa generasi untuk tiap skenario percobaan.

Tabel 1. Perbandingan Hasil Fungsi f_1

Generasi	C1	C2	DGA1	DGA2
50	0.03875	0.06450	0.078046	0.092206
100	0.03484	0.04750	0.053956	0.049282
500	0.01717	0.02169	0.026124	0.012490
1000	0.01157	0.01038	0.015302	0.010098
5000	0.00521	0.00719	0.005166	0.003602

Tabel 2. Perbandingan Hasil Fungsi f_2

Generasi	C1	C2	DGA1	DGA2
50	4.94577	3.68162	5.135998	3.451574
100	4.23221	3.44143	3.908094	3.19756
500	1.87598	1.36484	1.914862	1.71027
1000	1.24457	0.89586	1.216032	1.207514
5000	0.52245	0.52922	0.410762	0.37072

Dari Tabel 1 dan Tabel 2 terlihat pada generasi awal GA murni memberikan hasil yang lebih baik. Hal ini terjadi karena pada GA murni banyaknya individu pada populasi ditetapkan lebih banyak sehingga didapatkan area eksplorasi awal yang lebih lebih luas. Area explorasi yang lebih luas akan meningkatkan peluang ditemukannya solusi yang lebih mendekati optimum.

Pada generasi selanjutnya DGA secara konsisten menunjukkan hasil yang lebih baik meskipun banyaknya individu dalam satu subpopulasi lebih kecil. Kumpulan sedikit individu dalam satu subpopulasi meningkatkan peluang individu terjebak dalam daerah optimum lokal. Di sisi lain meningkatkan kemampuan eksploitasi untuk mencari titik optimum lokal.

Penggunaan metode crossover yang berbeda pada tiap subpopulasi membuat individu yag dihasilkan tiap subpopulasi juga berbeda. Ketika proses regenerasi terus berjalan, akhirnya individu-individu dalam satu subpopulasi menggerombol dalam daerah optimum lokal dan tidak mampu menghasilkan solusi yang lebih lebih. Penggunaan operator migrasi akan memasukkan individu baru yang berbeda ke dalam satu subpopulasi. Ketika individu baru ini terpilih sebagi induk pada proses crossover maka akan dihasilkan anak(*offspring*) yang lebih bervariasi. Hal ini akan meningkatkan kemampuan eksplorasi pencarian solusi pada subpopulai dan tidak lagi terjebak pada daerah optimum lokal.

5. Kesimpulan

- DGA memberikan hasil yang lebih optimum dibandingkan GAs pada generasi lanjut.
- Pada dua fungsi uji, migrasi pada generasi kelipatan 50 memberikan hasil yang lebih baik dibandingkan pada generasi kelipatan 25.

Daftar pustaka

[1] Dimyati, T,T, dan Dimyati, A. (2003), Operations Research, Model-Model Pengambilan Keputusan, Sinar Baru Algensindo, Bandung,

[2] Michalewicz, Zbigniew. (1996). Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, Heidelberg.

[3] Herrera, F. Lozano, M. & Verdegay, J.L. (1998). Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. Artificial Intelligence Review 12: 265–319.

[4] Houck, Christoper R, Jeffrey K dan Michael G Kay. (1999), A Genetic Algorithms for Function Optimization: A Matlab Implementation. http://www.dai.ed.ac.uk/groups/evalg/eag_local_copies_op_papers_body.html

[5] Radcliffe N.J. (1991). Equivalence Class Analysis of Genetic Algorithms. Complex Systems 5(2), 183–205.

[6] M¨uhlenbein H. & Schlierkamp-Voosen D. (1993). Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization. Evolutionary Computation 1, 25–49.