Penerapan Algoritma *Branch and Bound* untuk Menentukan Arah Evolusi

Mohamad Firda Fauzan, Tri Aji Nugroho, Ivan Sugiarto Widodo

Laboratorium Ilmu dan Rekayasa Komputasi
Departemen Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
if14127@students.if.itb.ac.id, if14141@students.if.itb.ac.id, if14149@students.if.itb.ac.id

Abstraksi

Akhir-akhir ini, banyak hama pengganggu yang mulai berevolusi untuk kebal terhadap suatu racun tertentu. Hal ini, tentu saja sangat mengganggu para konsumen yang memakai insektisida tertentu. Kami ingin memprediksi arah dan waktu dari perkembangan spesies agar produsen insektisida juga dapat "Berevolusi" mengikuti perkembangan dari spesies yang ia basmi.
Kami menemukan bahwa alur dari evolusi mengikuti dari algoritma Branch and Bound (B&B) karena pada evolusi terdapat variasi dan seleksi alam. Pada tahap Branching mirip seperti tahap variasi dan Bounding function mirip seperti seleksi alam.
Maka dari itu, kami menarik hipotesis bahwa algoritma B&B dapat kami gunakan untuk memprediksi arah dari evolusi suatu spesies. Namun, karena kami masih awam pada permasalahan biologi. Pada makalah ini kami mencobanya terlebih dahulu dengan simulasi terhadap tumbuhan.

I. Latar belakang

Suatu hari, kami mendapatkan pertanyaan yang menggugah hati kami terkait dengan fenomena spesies-spesies yang terjadi dialam, seperti makin kebalnya nyamuk terhadap obat antiserangga dan juga makin kebalnya penyakit tertentu terhadap antibiotika jenis tertentu. Hasil dari fenomena ini, membuat beberapa orang menjadi kecewa dengan pestisida atau obat-obat antibiotika yang beredar dipasaran. Variasi species berlangsung dengan cepat tanpa pernah kita ketahui hasil dari variasi-variasi hasil dari seleksi alam.
Jikalau produsen antiserangga dapat memprediksi arah dari perkembangan variasi species yang produknya basmi, pasti para konsumen tidak akan kecewa dengan keampuhan dari produknya tersebut. Atau jikalau seorang ahli penyakit dapat memprediksi perkembangan dari bakteri yang ia perangi dengan obat tertentu pasti ia dapat memformulasikan obat yang ampuh terhadap satu jenis bakteri tertentu sebelum bakteri tersebut berevolusi menjadi bakteri yang imun terhadap obat yang sekarang ia berikan. Para ahli botani juga dapat memperkirakan arah dari perkembangan dari sebuah tumbuhan yang merugikan sehingga ia dapat menciptakan metoda yang baik untuk menghambat laju dari tumbuhan tersebut.

II. Dasar Teori

Ide dasar yang menjadi titik tolak dari ide kami ini adalah, pertama Teori Evolusi yang dikemukakan oleh Darwin dan Algoritma Branch and Bound. Kami merasa pada Evolusi terdapat seleksi alam yang mirip dengan fungsi pembatas (Bounding Function) pada algoritma tersebut. Suatu varietas dari sebuah spesies dapat ditentukan kemungkinan bertahan hidupnya sampai dia dapat kawin dan kemudian varietas yang mempunyai kemungkinan hidup yang terbesar dengan keadaan alam yang telah kami prediksi sebelumnya, dibangkitkan kemungkinan keturunannya. Varietas yang paling dominan dalam suatu spesies akan menentukan arah perkembangan dari suatu spesies. Sehingga kami mengambil hipotesis bahwa arah dari perkembangan **spesies dapat ditentukan dengan algoritma branch and bound**. Hal ini, akan kami uraikan secara detail sebagai berikut.

a. Variasi dan Seleksi Alam.

Pada buku *The Origin of Species*, Darwin menyatakan bahwa perkembangan dari sebuah species sangat bergantung pada dua hal, yaitu variasi dan seleksi alam. Makhluk hidup, menurutnya melakukan persilangan-persilangan antar varietas dalam suatu species untuk membuat variasi baru secara acak yang kemudian terseleksi alam. Organisme yang bertahan hidup sampai, paling tidak, untuk kawin akan mendapatkan kemungkinan yang besar untuk meneruskan bentuk morfologinya dan membentuk spesies[1]. Pernyataan Darwin tentang variasi morfologi kemudian diperkuat

[1] Dalam hal ini, ia menyatakan pada kasus variasi burung merpati dan anjing yang teliti di Inggris

oleh Mendel melalui teori hereditas. Ia menyatakan bahwa dalam setiap bentuk morfologi dibawa oleh gen yang bersifat dominan, resesif atau intermediasi (campuran)[2] , dan gen gen terebut berpasang-pasangan[3]. Sifat-sifat dari gen resesif akan diturunkan tetapi dalam pembentukkan morfologi akan tertutupi oleh gen yang dominan. Sementara, pada kasus intermediasi, gen-gen yang berlawanan akan bercampur untuk menentukan bentuk morfologis.

b. Algoritma *Branch and Bound*

Algoritma B&B merupakan metode pencarian di dalam ruang solusi secara sistematis. Ruang solusi diorganisasikan kedalam pohon ruang status. Pada B&B, untuk mempercepat pencarian ke simpul solusi, maka setiap simpul diberi seuh nilai ongkos (cost). Simpul berikutnya yang akan diekspansi tidak lagi berdasarkan uruan pembangkitannya (sebagaimana pada BFS murni) tetapi simpul yang memiliki ongkos yang paling kecil di antara simpul-simpul hidup lainnya. Nilai ongkos pada setiap simpul I menyatakan taksiran oengkos termurah lintasan dari simpul i ke simpul solusi (goal node):

$\hat{C}(i)$ = nilai taksiran lintasan termurah dari simpul status *i* ke status tujuan.

Dengan kata lain, $\hat{C}(i)$ menyatakan batas bawah (*lower bound*) dari ongkos pencarian solusi dari status *i.* Ongkos ini dihitung dengan suatu fungsi pembatas. Tetapi pada penelitian kami ini, kami mengubah $\hat{C}(i)$ menjadi menyatakan batas atas karena kami akan menggenerasikan varietas dengan kemungkinan hidup terbesar[4].

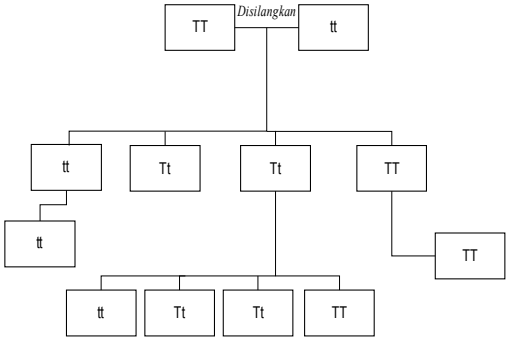
III Perumusan Masalah

Pada makalah kami ini, kami akan melakukan dua proses utama untuk memperdiksi dari perkembangan gen suatu spesies. Dua tahap itu adalah, Penggenerasian, Branch and Bound.

Spesies yang kami gunakan untuk mencoba algoritma kami adalah tumbuhan karena pada tumbuhan ia dapat melakukan penyerbukan sendiri sehingga akan menyempitkan ruang kemungkinan. Tetapi untuk awal, kami akan menyilangkan varietas yang merupakan dua varietas mayoritas dalam spesies untuk mendapatkan campuran-campuran gen. Untuk jenis gen kami menggunakan jenis gen dominan atau resesif sehingga akan menyempitkan bentuk morfologis menjadi hanya yang berlawanan saja[5]. Masalah tautan gen juga tidak kami perhitungkan dalam makalah ini. Dan juga kami tidak memperhitungkan masalah mutasi genetis karena kami belum mengetahuinya secara baik.

a. Penggenerasian

Tahapan penggenerasian sendiri kami bagi menjadi dua tahap terpisah, yaitu penggenerasian awal, dan penggenerasian umum. Pada tahap penggenerasian awal, kami melakukan persilangan varietas yang mempunyai gen-gen yang bebeda misal, tumbuhan dengan gen TT dengan tt atau Tt dengan tt. Kemudian pada penggenerasian umum, kami mengasumsikan bahwa tumbuhan tersebut melakukan penyerbukan sendiri misal pada persilangan dari gen TT dengan tt mempunyai anak Tt yang kemudian melakukan penyerbukan sendiri dan menghasilkan anak yang merupakan kombinasi gen Tt. Untuk lebih jelasnya lihat gambar 1.



Gambar 1 skema penggenerasian

b. *Branch and Bound*

[2] Sumber http://en.wikipedia.org/wiki/Heredity (10.00, 9 maret 2006)

[3] terkadang sifat morfologi disusun atas dua gen yang berlawanan, contohnya pada kasus ketinggian tanaman. Terkadang oleh tiga gen, contoh pada darah dan warna mata

[4] Munir, Rinaldi. *Diktat Kuliah IF2251 Strategi Algoritmik,*hal 150

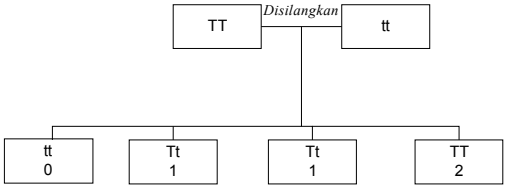
[5] Maksudnya adalah, pada tumbuhan ada tinggi rendah dua sifat ini kami anggap saling menutupi yang tinggi menutupi yang rendah. Karena kami tidak mengerti sifat dari intermediasi

Tahapan Branch and Bound dimulai dengan melakukan pembobotan yang urut berdasarkan kemungkinan hidup paling tinggi. Contohnya

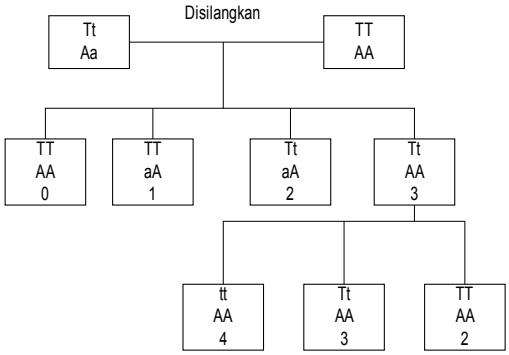
besar dari yang pendek, maka ia mempunyai bobot yang lebih tinggi dari yang berbatang pendek pendek. Kemudian, untuk tumbuhan dengan gen yang mampu bertahan hidup lebih banyak diberikan nilai lebih besar.
Skema ini, akan berakhir ketika didapatkan gen yang homogen dan mampu beradaptas dengan lingkungannya. Atau bila tidak didapatkan juga, bisa dibatasi sampai keturunan ke berapa atau dengan kata lain dibatasi oleh waktu.

IV. Simulasi Sederhana

Misal, kami mengasumsikan bahwa terjaadi persilangan antara dua varietas mayoritas dari spesies yang mempunyai fenotip tinggi dan rendah dengan umur reproduksi 3 tahun. Kami mengasumsikan bahwa dalam jangka waktu 5 tahun kondisi alam tempat tumbuhan itu hidup, ditumbuhi oleh banyak tumbuhan lain yang dapat menutupi sinar matahari, sehingga kami berasumsi bahwa tumbuhan yang mempunyai batang yang tinggilah yang akan bisa bertahan. Kemudian data-data gen tersebut dan data gen yang akan bertahan, kami masukkan kedalam simulator. Kemudian, simulator menghasilkan pohon yang dapat dilihat pada gambar 2..



Gambar 2
Dari sini, kami simpulkan bahwa gen yang akan bertahan adalah gen TT
Untuk contoh dua gen, dengan tumbuhan jenis yang sama dan keadaan alam dalam 8 tahun adalah penurunan permukaan air tanah, sehingga hanya yang memiliki akar yang panjang dan batang yang pendek akan selamat. Kemudian kami mengasumsikan bahwa dua jenis kombinasi gen yang terbanyak adalah TtAa dan TTAA dan kami mengasumsikan bahwa dua gen tersebut terjadi perkawinan pada pohon, pohon dengan batang yang tinggi pada suatu waktu memiliki kemungkinan untuk hidup sampai ia mampu kawin lebih silang. Kami memasukkan data-data tersebut kedalam mesi simulasi dan mendapatkan pohon BFS seperti berikut,



Gambar 3
Dari gambar diatas, pohon dengan gen tt AA mempunyai kemungkinan untuk hidup lebih tinggi daripada yang lain yaitu, 8 maka kami dapat menyimpulkan bahwa pohon yang dapat bertahan adalah pohon dengan gen tt AA. Sampai pada tahap ini, perhitungan kami terbukti masih tepat.

V. Simpulan

Dengan melakukan induksi dari data-data simulasi, kami menarik hipotesis kami dapat memprediksi arah perkembangan dari suatu spesies. Asalkan gen-gen mayoritas yang terlibat sudah diketahui. Kemudian, dengan mengetahui gen-gen yang resisten terhadap suatu keadaan alam, umur dari suatu spesies. Perkembangan varietas mayoritas dari suatu spesies dapat diketahui dengan menggunakan algoritma *Branch and Bound*.
Kami juga merasa masih banyak hal yang dapat dikembangkan dari makalah ini, seperti penambahan efek dari mutasi dan juga efek dari tautan genetik. Namun, karena untuk melakukan keduah hal tersebut diperlukan pemetaan gen secara menyeluruh maka kami memerlukan bantuan dari jurusan lain untuk menyempurnakan algoritma ini.
Pada makalah kami sekarang ini, masih banyak asumsi yang kami gunakan dalam mengimplementasikan algoritma ini dan masih belum terbukti nyata di lapangan. Oleh karena itu, kami hanya mampu untuk menyebutnya sebagai hipotesis semata.

Daftar Pustaka

[0] Darwin, Charles. *The Origin of Species.* Ikon Teralitera, 2002
[1] Munir, Rinaldi. *Diktat Kuliah IF2251 Strategi Algoritmik*, STEI-ITB, 2006

Lampiran Kode Program
(Dalam C#**)**

```
using System;
using System.Collections.Generic;
using System.Text;
//Hanya untuk gen dengan jenis kurang dari 2
namespace HereditasBB
{
class Hereditas
{
public static void GenerateGenod(String Gen1, String Gen2,ref List<String> Gn1,ref
List<String> Gn2)
{
int x =
System.Convert.ToInt32(Math.Pow(System.Convert.ToDouble(2),System.Convert.ToDouble((Gen1.
Length)/2)));
String [] Genod1 = new String[x];
int pil = Gen1.Length / 2;
int count = 0;
switch (pil)
{
case 2 :
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
Genod1[count]=Gen1.Substring(i, 1) + Gen1.Substring(j + 2, 1);
count++;
}
}
break;
case 3 :
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
for (int k = 0; k < 2; k++)
{
Genod1[count] = Gen1.Substring(i, 1) + Gen1.Substring(j + 2, 1) + Gen1.Substring(k+4,1);
count++;
}
}
}
break;
case 4 :
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
for (int k = 0; k < 2; k++)
{
for (int l = 0; l < 2; l++)
{
Genod1[count] = Gen1.Substring(i, 1) + Gen1.Substring(j + 2, 1) + Gen1.Substring(k + 4,
1) + Gen1.Substring(l + 6, 1);
count++;
}
}
}
}
break;
case 5:
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
for (int k = 0; k < 2; k++)
{
for (int l = 0; l < 2; l++)
{
for (int m = 0; m < 2; m++)
{
```

```
Genod1[count] = Gen1.Substring(i, 1) + Gen1.Substring(j + 2, 1) + Gen1.Substring(k + 4,
1) + Gen1.Substring(l + 6, 1) + Gen1.Substring(m + 8, 1);
count++;
}
}
}
}
}
break;
}
for (int i = 0; i < x ; i++)
{
if (!Gn1.Contains(Genod1[i]))
{
Gn1.Add(Genod1[i]);
}
}
//============================================================================
x =
System.Convert.ToInt32(Math.Pow(System.Convert.ToDouble(2),System.Convert.ToDouble((Gen2.
Length)/2)));
String [] Genod2 = new String[x];
pil = Gen2.Length / 2;
count = 0;
switch (pil)
{
case 2 :
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
Genod2[count]=Gen2.Substring(i, 1) + Gen2.Substring(j + 2, 1);
count++;
}
}
break;
case 3 :
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
for (int k = 0; k < 2; k++)
{
Genod2[count] = Gen2.Substring(i, 1) + Gen2.Substring(j + 2, 1) + Gen2.Substring(k+4,1);
count++;
}
}
}
break;
case 4 :
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)
{
for (int k = 0; k < 2; k++)
{
for (int l = 0; l < 2; l++)
{
Genod2[count] = Gen2.Substring(i, 1) + Gen2.Substring(j + 2, 1) + Gen2.Substring(k + 4,
1) + Gen2.Substring(l + 6, 1);
count++;
}
}
}
}
break;
case 5:
for (int i = 0; i < 2; i++)
{
for (int j = 0; j < 2; j++)

{
for (int k = 0; k < 2; k++)
```

```
{
for (int l = 0; l < 2; l++)
{
for (int m = 0; m < 2; m++)
{
Genod2[count] = Gen2.Substring(i, 1) + Gen2.Substring(j + 2, 1) + Gen2.Substring(k + 4,
1) + Gen2.Substring(l + 6, 1) + Gen2.Substring(m + 8, 1);
count++;
}
}
}
}
}
break;
}
for (int i = 0; i < x; i++)
{
if (!Gn2.Contains(Genod2[i]))
{
Gn2.Add(Genod2[i]);
}
}
}
public static void Silangkan(List<String> Gn1, List<String> Gn2,List<String> Anak)
{
String temp = "";
for (int i = 0; i < Gn1.Count; i++)
{
for (int j = 0; j < Gn2.Count; j++)
{
for (int k = 0; k < Gn1[0].Length; k++)
{
temp = temp + Gn1[i].Substring(k, 1) + Gn2[j].Substring(k, 1);
}
Anak.Add(temp);
temp = "";
}
}
}
public static void GetValue(String Border,List<String> Anak,List<int> AnakVal)
{
int val = 0;
for (int i = 0; i < Anak.Count; i++)
{ val = 0;
for (int j=0;j< Border.Length;j++)
{
if (Anak[i].Substring(j, 1) == Border.Substring(j,1))
{
val++;
}
}
AnakVal.Add(val);
}
}
public static void FindMaxProbability(List<String> Anak, List<int> AnakVal,ref String
Candidat)
{
int max=-999;
int idxmax = -999;
for (int i = 0; i < AnakVal.Count; i++)
{
if (AnakVal[i] > max)
{
max = AnakVal[i];
idxmax = i;
}
}

Candidat = Anak[idxmax];
}
}
}
```