PENCARIAN SOLUSI PEMROGRAMAN NON LINIER MENGGUNAKAN ALGORITMA BRANCH-AND-BOUND

Victor Hariadi
Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya
Gedung Teknik Informatika, Kampus ITS, Jl. Raya ITS, Sukolilo, Surabaya - 60111
e-mail: victor@its-sby.edu

ABSTRAK

Pemrograman nonlinier adalah satu bagian penting dari permasalahan optimasi, baik dari sudut pandang matematis maupun aplikasi. Banyak permasalahan dunia nyata yang dapat direpresentasikan ke dalam bentuk permasalahan pemrograman non linier. Dalam pemrograman nonlinier diperlukan metode untuk mencari nilai optimal global agar tidak terjebak pada pencapaian nilai optimal local. Pada penelitian ini dicoba untukmengaplikasikan algoritma branch-and-bound melalui proses relaxation pada permasalahan untuk mendapatkan solusi optimal global. Agar pertumbuhan jumlah sub permasalahan dapat dikendalikan maka kami memanfaatkan reformulasi kondisi (Karush-Kuhn Tucker) KKT. Uji coba dilakukan dengan menggunakan permasalahan pemrograman kuadratik. Dari percobaan yang dilakukan menunjukkan bahwa pembatasan sub permasalahan melalui reformulasi KKT sangat membantu pencapaian solusi optimal dengan algortima branch-and-bound.

Kata kunci: permaslahan optimasi, pemrograman non linier, algoritma branch-and-bound, reformulasi kondisi KKT.

I. PENDAHULUAN

Mencari solusi optimal pada permasalahan pemrograman nonlinier, dalam hal ini pada bentuk permasalahan konveks, diperlukan suatu algoritma untuk mendapatkan nilai solusi optimal global, dan tidak terjebak dalam perolehan solusi optimal lokal [5].

Secara natural, algoritma branch-and-bound akan membagi sebuah permasalahan menjadi sub-sub permasalahan yang berukuran kecil. Kemudian masing-masing sub permasalahan tersebut akan diselesaikan melalui metode-metode penyelesaian permasalahan optimasi non linier yang tersedia dengan memperhitungkan batasan yang ada dari permasalahan asli [2].

Implementasi algoritma branch-and-bound yang digunakan dalam penelitian ini dimulai dengan menentukan nilai batas atas (*upper bound*) dan batas bawah (*lower bound*). Sementara untuk memperhitungkan pembatasan dalam setiap sub permasalahan dimulai dengan reformulasi kondisi KKT di dalam pemrograman kuadratik. Tujuannya agar tidak banyak muncul sub permasalahan. Atau yang selanjutnya kita sebuat sebagai node.

II. PEMROGRAMAN NON LINIER

A. Bentuk Dasar Pemrograman Non Linier

Permasalahan *Pemrograman Non Linier* dengan pembatas linear mempunyai bentuk dasar sebagai berikut [1]:

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && Ax = b, \\ & && x \geq 0 \end{aligned} \qquad \textbf{(1)}$$

$f(x)$ dapat diturunkan terus menerus dan *convex* dari R^n sampai R.

dimana,

R^n = himpunan bilangan real berdimensi n
R^{mxn} = himpunan bilangan real berdimensi mxn
$A \in R^{mxn}$ = matriks A yang anggota himpunannya bilangan real dimensi mxn
dan $x, b \in R^m$

Apabila fungsi obyektif berbentuk $f(x) = ½\, x^T Q x + c^T x$ dimana $Q \in R^{nxn}$ bersifat semidefinit positif dan $c \in R^n$ maka permasalahan pemrograman non linier akan menjadi bentuk permasalahan kuadratik dengan sifat konveks dari R^n sampai R, $A \in R^{mxn}$, dan $b \in R^m$.

Sehingga bentuk permasalahan optimasi konveks ini menjadi:

$$\begin{aligned} &\text{minimize} && f(x) = ½\, x^T Q x + c^T x \\ &\text{subject to} && Ax = b, \\ & && x \geq 0 \end{aligned} \qquad \textbf{(2)}$$

dimana Q harus semidefinit positif untuk memastikan agar persoalan tersebut mempunyai penyelesaian yang unik.

B. Kondisi Optimal

Berdasarkan kondisi KKT, x^* merupakan solusi optimal dari permasalahan tersebut jika dan hanya jika terdapat $y^* \in R^m$ se-hingga (x^*, y^*) harus memenuhi kondisi berikut:

$$\begin{aligned} &Ax - b = 0, \\ &\nabla f(x) - A^T y \geq 0, \\ &x^T(\nabla f(x) - A^T y) = 0 \end{aligned} \qquad \textbf{(3)}$$

Ketiga kondisi di atas ekivalen dengan :
Ax – b = 0

$(x\text{-}x^*)^T\ (\nabla f(x) - A^T y) \geq 0$,
$\forall x \geq 0$ **(4)**

III. PEMROGRAMAN KUADRATIK

A. Bentuk Dasar Pemrograman Kuadratik

Bentuk umum Pemrograman Kuadratik adalah sebagai berikut [6]:

Bentuk Primal :

$$\text{minimize} : \frac{1}{2}x^T Hx + c^T x$$
$$\text{subject to} : Ax \leq b, x \geq 0 \quad \textbf{(5)}$$

dimana $x \in \mathbb{R}^n$ adalah variable dan $(H, A, b, c) \in \mathbb{R}^{n \times n} \times \mathbb{R}^{m \times n} \times \mathbb{R}^m \times \mathbb{R}^n$ adalah data.

Sementara bentuk Dual :

$$\text{maximize} : -\frac{1}{2}x^T Hx + b^T y$$
$$\text{subject to} : -Hx + A^T y + z = c, z \geq 0 \quad \textbf{(6)}$$

Pembatas berupa pertidaksamaan dapat dijadikan bentuk standar yaitu bentuk persamaan linier, dengan cara menambahkan variabel yang sesuai dengan jenis tanda pertidaksamaan tersebut [1]:

- Untuk tanda $\leq$ dapat diubah ke bentuk normal dengan menambahkan *slack variable* sehingga pada persamaan pembatasnya ditambahkan variabel $+W$.
- Untuk tanda $\geq$ dapat diubah ke bentuk normal dengan mengurangkan *surplus variable* sehingga pada persamaan pembatasnya ditambahkan variabel $-S$.

Notasi variabel yang ditambahkan di atas dapat sesuai dengan notasi variabel disain, dari permsalahan yang akan dipecahkan.

Contoh bentuk persamaan pemrograman kuadratik dan penjabaran matriks dari permasalahan yang ada :

$$\begin{aligned}\text{minimize} \quad & f(x) = x_1^2 + 0.1x_2^2 + x_3^2 - x_1x_3 - x_2 \\ \text{subject to} \quad & x_1 + x_2 + x_3 \geq 1 \\ & x_1, x_2, x_3 \geq 0\end{aligned}$$

Diubah ke bentuk standar :

$$\begin{aligned}\text{minimize} \quad & f(x) = x_1^2 + 0.1x_2^2 + x_3^2 - x_1x_3 - x_2 \\ \text{subject to} \quad & x_1 + x_2 + x_3 - x_4 = 1 \\ & x_1, x_2, x_3 \geq 0\end{aligned}$$

Penjabaran matriks dari fungsi oyektif dan fungsi-fungsi pembatasnya adalah sebagai berikut :

$$H = \begin{bmatrix} 2 & 0 & -1 & 0 \\ 0 & 0.2 & 0 & 0 \\ -1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, c = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \end{bmatrix},$$
$$b = [1]^T, A = [1 \quad 1 \quad 1 \quad -1]$$

Dengan nilai x_0 adalah sebagai berikut :

1. Pada iterasi awal dideklarasikan nilai x yaitu sebagai berikut : $x_0 = [0 \quad 0 \quad 0 \quad -1]^T$.
2. Pada akhir iterasi akan didapatkan solusi optimal dari permasalahan di atas yaitu : $x^* = [0 \quad 5 \quad 0]^T$.

B. Matriks Semidefinit Positif

Suatu matrik dikatakan semidefinit positif jika nilai $x^T Ax \geq 0$ untuk semua nilai x, atau semua nilai *eigen* matrik A adalah non negatif [5]. Jika dilakukan eliminasi *Gaussian* terhadap A agar matriks A menjadi matriks segitiga atas, U, maka akan diperoleh semua elemen diagonal U adalah non negatif. Pembuktian definit matrik bisa dilihat pada contoh berikut:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \rightarrow x^T Ax = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$
$$= 2x_1^2 + 2x_2^2 + 2x_1x_2$$

Untuk nilai,

$x = [3 \quad 0]$

$\rightarrow x^T Ax = 2*3^2 + 2*0^2 + 2*3*0 = 18$

$x = [0 \quad -2]$

$\rightarrow$

$x^T Ax = 2*0^2 + 2*(-2)^2 + 2*0(-2) = 4$

$x = [1 \quad -3]$

$\rightarrow \quad x^T Ax = 2*1^2 + 2*(-3)^2 + 2*1*(-3) = 14$

Bahwa untuk nilai x sembarang, maka diperoleh $x^T Ax \geq 0$.

Sementara nilai *eigen* A diperoleh dari penyelesaian det $(A - \lambda.I) = 0$

Nilai *eigen* untuk $A \rightarrow \begin{matrix} \lambda = 1 \\ \lambda = 3 \end{matrix}$

Nilai *eigen* yang diperoleh adalah non negatif.

Jika dilakukan eliminasi *Gaussian*, diperoleh :

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \rightarrow U = \begin{bmatrix} 2 & 1 \\ 0 & 1.5 \end{bmatrix}$$

Dari matrik segitiga atas U yang dibentuk, diperoleh semua elemen diagonal U adalah non negatif. Dari semua pembuktian tersebut, matrik A bisa digolongkan ke dalam matrik semidefinit positif.

IV. ALGORITMA BRANCH-AND-BOUND PADA PEMROGRAMAN KUADRATIK

A. Pendekatan Branch and Bound

Di dalam permasalahan pemrograman kuadratik (*quadratic programming / QP*), aplikasi branch-and-bound akan secara berulang membagi permasalahan induk menjadi node-node yang juga disebut *convex feasible sets* [2]. Dan untuk mengontrol pertumbuhan jumlah pencabangan tersebut, maka digunakan finite KKT-*branching* (yang merupakan perluasan dari *first order* KKT) [3].

Bentuk dasar dari permasalahan yang memaksimalkan tujuan obyektif fungsi kuadratik:

$$\begin{aligned} \max \quad & \tfrac{1}{2}x^T Q x + c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned} \qquad \textbf{(7)}$$

Dimana $x \in R^n$ adalah variabel dan $(Q, A, b, c) \in R^{n\times n} \times R^{m\times n} \times R^m \times R^n$ adalah datanya. Yang mengacu pada *feasible set* dari QP sebagai:

$$P := \{x \in R^n : Ax \leq b, x \geq 0\} \qquad \textbf{(8)}$$

Semua pemrograman kuadratik dengan batasan linier dapat menggunakan bentuk standar seperti **(7)**, diasumsikan bahwa Q adalah simetris dan semidefinit positif.

B. Reformulasi KKT pada Pemrograman Kuadratik

Dengan mengenalkan pengali nonnegatif y dan z untuk batasan $Ax \leq b$ dan $x \geq 0$ dalam P [4], secara berturut-turut semua solusi optima lokal x dari (QP) harus memiliki property dari set:

$$\begin{aligned} G_x &:= \{(y,z) \geq 0 : A^T y - z = Qx + c\} \\ C_x &:= \{(y,z) \geq 0 : (b - Ax) \circ y = 0, x \circ z = 0\} \end{aligned} \qquad \textbf{(9)}$$

memenuhi $G_x \cap C_x \neq \emptyset$. Dengan kata lain, G_x adalah set pengali dimana gradien *Lagrangian* hilang, dan C_x terdiri dari pengali yang memenuhi batasan x. Kondisi $G_x \cap C_x \neq \emptyset$ menetapkan bahwa x adalah *first-order KKT point*.

Berikut adalah properti dari KKT point:

Proposisi 3.1 (*Giannessi and Tomasin*) [7] *Jika diasumsikan* $x \in P$ *dan* $(y,z) \in G_x \cap C_x$. *Maka* $x^T Q x + c^T x = b^T y$.

Bukti bahwa $(y,z) \in C_x$ menyatakan (secara tidak langsung) $(Ax)^T y = b^T y$ dan $x^T z = 0$. Berikutnya, sebelum mengalikan persamaan $A^T y - z = Qx + c$ dengan x^T menghasilkan $x^T Q x + c^T x = (Ax)^T y - x^T z = b^T y$.

Persamaan **(9)** dapat direformulasi sebagai program linier dengan batasan pelengkap:

$$\begin{aligned} \max \quad & \tfrac{1}{2}b^T y + \tfrac{1}{2}c^T x \\ \text{s.t.} \quad & x \in P \quad (y,z) \in G_x \cap C_x \end{aligned} \qquad \textbf{(10)}$$

Dengan menurunkan $(y,z) \in C_x$ didapat *Linear Programming* (LP) relaxation alami [7]:

$$\begin{aligned} \max \quad & \tfrac{1}{2}b^T y + \tfrac{1}{2}c^T x \\ \text{s.t.} \quad & x \in P \quad (y,z) \in G_x \end{aligned} \qquad \textbf{(11)}$$

Reformulasi KKT (RKKT) akan menghasilkan pendekatan pencabangan yang berhingga yang menghasilkan 4 himpunan indeks

$F^x, F^z \subseteq \{1, \cdots, n\}$ dan $F^{b-Ax}, F^y \subseteq \{1, \cdots, m\}$, yang memenuhi $F^x \cap F^z = \emptyset$ dan $F^{b-Ax} \cap F^y = \emptyset$.

Berikut adalah hasil pembatasan KKT:

$$\begin{aligned} \max \quad & \tfrac{1}{2}b^T y + \tfrac{1}{2}c^T x \\ \text{s.t.} \quad & x \in P \quad (y,z) \in G_x \cap C_x \\ & x_j = 0, \; j \in F^x \\ & z_j = 0, \; j \in F^z \\ & A_i x = b_i, \; i \in F^{b-Ax} \\ & y_i = 0, \; i \in F^y \end{aligned} \qquad \textbf{(12)}$$

Pada node yang diberi, algoritma branch-and-bound akan menyelesaikan relaksasi konveks dari **(12)**.

C. Finite Branch-and-Bound

Berikut diperkenalkan dasar *semidefinite programming* (SDP) relaxation. Matrik variabel Y yang terkait dengan $x \in P$ dengan persamaan kuadratik berikut [3]:

$$Y = \begin{pmatrix} 1 \\ x \end{pmatrix}\begin{pmatrix} 1 \\ x \end{pmatrix}^T = \begin{pmatrix} 1 & x^T \\ x & xx^T \end{pmatrix} \qquad \textbf{(13)}$$

Dari persamaan **(13)** diperoleh :

- Y adalah simetrik dan semidefinit positif. Yaitu, $Y \in S^{1+n}_+$ atau secara sederhana ($Y \succcurlyeq 0$);
- Jika batasan dikalikan $Ax \leq b$ dari P dengan beberapa x_i, didapat pertidaksamaan kuadratik $bx_i - Axx_i$ dimana valid untuk P; pertidaksamaan ini dapat ditulis dalam istilah Y sebagai:
$$(b|-A)Y_{e_i} \geq 0 \quad \forall i = 1,\cdots,n$$
- Fungsi objektif dari **(4.1)** dapat dimodelkan dalam bentuk Y sebagai:

$$\frac{1}{2}x^TQx + c^Tx = \frac{1}{2}\begin{pmatrix}0 & c^T\\ c & Q\end{pmatrix} \bullet \begin{pmatrix}1 & x^T\\ x & xx^T\end{pmatrix} = \frac{1}{2}\begin{pmatrix}0 & c^T\\ c & Q\end{pmatrix} \bullet Y \quad \textbf{(14)}$$

Untuk kemudahan didefinisikan
$$K := \{(x_0,x) \in \mathbb{R}^{1+n} : Ax \leq x_0 b, (x_0,x) \geq 0$$

dengan
$$\bar{Q} = \frac{1}{2}\begin{pmatrix}0 & c^T\\ c & Q\end{pmatrix} \quad \textbf{(15)}$$

yang mana dapat dituliskan properti kedua dan ketiga diatas secara lebih sederhana menjadi $Y_{e_i} \in K$ dan $x^TQx + c^Tx = \bar{Q} \bullet Y$ .Sebagai tambahan M_+ menandakan set dari semua Y memenuhi property pertama dan kedua:

$$M_+ := \{Y \succcurlyeq 0 : Ye_i \in K \quad \forall i = 1,\cdots,n\}$$

Kemudian didapat formulasi yang sama dari **(7)** berikut ini:

$$\begin{aligned} \max \quad & \bar{Q} \bullet Y \\ \text{s.t.} \quad & Y = \begin{pmatrix}1 & x^T\\ x & xx^T\end{pmatrix} \in M_+, \; Ye_0 = (1;x) \\ & x \in P \end{aligned} \quad \textbf{(16)}$$

Akhirnya, dengan menurunkan kolom *n* terakhir dari persamaan **(13)**, berikut ini adalah SDP relaxation linier dari **(7)** :

$$\begin{aligned} \max \quad & \bar{Q} \bullet Y \\ \text{s.t.} \quad & Y \in M_+, \quad Ye_0 = (1;x) \\ & x \in P \end{aligned} \quad \textbf{(17)}$$

Kemudian digabungkan dasar SDP relaksasi ini dengan LP relaksasi **(14)** untuk menghasilkan relaksasi SDP dari **(10)**. Secara rinci dipertimbangkan dua obyektif yang disamakan menggunakan batasan tambahan:

$$\begin{aligned} \max \quad & \bar{Q} \bullet Y \\ \text{s.t.} \quad & Y \in M_+, \quad Ye_0 = (1;x) \\ & x \in P, \quad (y,z) \in G_x \\ & \bar{Q} \bullet Y = (b^Ty + c^Tx)/2 \end{aligned} \quad \textbf{(18)}$$

Permasalahan optimasi ini adalah relaksasi yang valid dari **(10).** Dan jika batasan **(12)** adalah valid untuk semua titik pada area yang mungkin untuk **(10),** maka (x,y,z) dapat menjadi suatu titik dan definisikan Y menurut **(13)**. Maka (Y,x,y,z) akan memenuhi empat batasan pertama dari **(4.12)** dengan konstruksi dan batasan **(12)** memenuhi Proposisi 3.1 dan **(13)**

Jika diasumsikan $(\bar{Y},\bar{x},\bar{y},\bar{z})$ adlah himpunan solusi optimal dari SDP_1 **(18)**, maka:

$$\bar{Q} \bullet \bar{Y} = \frac{1}{2}b^T\bar{y} + \frac{1}{2}c^T\bar{x} \quad \textbf{(19)}$$

Karena $(\bar{y},\bar{z}) \in G_{\bar{x}} \cap C_{\bar{x}}$. Proposisi 3.1 menyatakan (secara tidak langsung) $(b^T\bar{y} + c^T\bar{x})/2 = \bar{x}^TQ\bar{x} + c^T\bar{x}$. Sehingga $\bar{Q} \bullet \bar{Y}$ sama dengan nilai fungsi pada $\bar{x}$. Karena $\bar{Q} \bullet \bar{Y}$ adalah batas atas dalam nilai optimal pada QP **(7)**, diikuti $\bar{x}$ adalah solusi optimal.

Jika solusi dari relaxation menghasilkan KKT poin $(\bar{x},\bar{y},\bar{z})$, maka $\bar{x}$ adalah solusi optimal dari **(12)**.

D. Implementasi Branch-and-Bound

Algoritma branch-and-bound dimulai dari evaluasi *root node* dan mengevaluasi node-node di bawahnya, untuk kemudian menambahkan atau menghapus node dari tree pada tiap stage. Mengevaluasi node melibatkan penyelesaian relaksasi SDP dan kemudian *fathomed* node (jika mungkin) atau mencabangkan node. Algoritma berakhir ketika semua node yang di-*generate* oleh algoritma yang telah di-*fathomed*.

Fathoming node (yaitu, menghapusnya dari pertimbangan lebih lanjut) adalah diperbolehkan hanya jika subproblem **(12)** tidak berisi solusi optimal dari **(7)** Selain itu, pertama, jika relaksasi adalah *infeasible* maka **(12)** adalah *infeasible*, sehingga tidak ada solusi optimal. Kedua, kita memperoleh *fathom* jika nilai objektif yang direlaksasi pada $(\bar{x},\bar{y},\bar{z})$ adalah sama dengan atau kurang dari objektif dari $x \in P$. Karena nilai **(7)** dari $\bar{x}$ tidak bisa lebih besar dari nilai yang direlaksasi, *fathoming* terjadi dalam kaitan $\bar{x}$ hanya ketika dua nilai adalah sama atau tidak memiliki *gap*.

Kriteria pencabangan: pilih subproblem dengan menggunakan strategi best-bound-first, yang

berarti bahwa subproblem dengan batas bawah terkecil (*smallest lower bound*) akan dipilih untuk dibagi dalam setiap iterasi.

V. UJI COBA

Uji coba dilakukan pada permasalahan pemrograman convex khususnya pada permasalahan pemrograman kuadratik. Data yang digunakan untuk uji coba program ini adalah data matriks dari permasalahan yang ada. Dari permasalahan yang ada dalam bentuk awal diubah ke bentuk standar. Untuk memudahkan penyelesaian permasalahan, fungsi permasalahan diubah ke bentuk matriks, yaitu Q, A, b, c. sedangkan matriks x,y,z digunakan sebagai nilai awal untuk penyelesaian permasalahan. Berikut akan diberikan contoh uji coba.

Bentuk awal :

min $f(x) = x_1^2 + 4x_2^2 - 8x_1 - 16x_2$

Subject to $x_1 + x_2 \geq 5$

$x_1 \geq 3$

$x_1, x_2 \geq 0$.

Bentuk standar :

min $f(x) = x_1^2 + 4x_2^2 - 8x_1 - 16x_2$

Subject to $x_1 + x_2 + x_3 = 5$

$x_1 + x_4 = 3$

$x_1, x_2, x_3, x_4 \geq 0$.

Bentuk matriks :

$$Q = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad c = \begin{bmatrix} -8 \\ -16 \\ 0 \\ 0 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 5 \\ 3 \end{bmatrix}.$$

$x = [2 \quad 1 \quad 2 \quad 1]^T$

$y = [-8.0001 \quad -0.0001]^T$

$z = [4.0002 \quad 0.0001 \quad 8.0001 \quad 0.0001]^T$.

Dari percobaan yang dilakukan pada didapatkan hasil sebagai berikut :

Nilai optimal :

$x_1 = 3.106$, $x_2 = 1.858$.

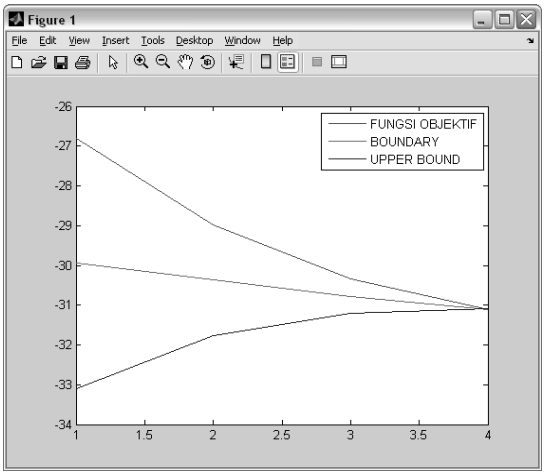
Nilai optimal fungsi objektif = -31.1201

Waktu = 0.168728 seconds.

Dari hasil uji coba dapat dilihat yaitu nilai optimal untuk vector x adalah $x_1 = 3.106$, $x_2 = 1.858$. dengan perkiraan nilai fungsi objektif yang optimal adalah -31.1201. Waktu yang diperlukan untuk pencapaian solusi optimal adalah 0.168728 seconds.

Pada uji coba dimasukkan nilai parameter *error* = 0.2. Nilai error yang dimaksud adalah selisih absolut antara nilai fungsi objektif dan nilai *boundary*. Nilai 0.2 adalah kedekatan nilai fungsi objektif optimal dengan nilai *boundary*. Semakin kecil nilai error maka akurasi hasil mendekati optimal. Karena fungsi dari nilai *boundary* adalah sebagai pembatas pada tiap permasalahan yang diselesaikan.

Berikut ini akan ditampilkan gambar grafik trayektori fungsi objektif, nilai *boundary* dan nilai batas atas (*upper bound*). Dari Gambar 5-1 dapat dilihat bahwa nilai fungsi objektif semakin mendekati nilai *boundary* dan nilai batas atas (*upper bound*). Ini menunjukkan nilai *boundary* mendekati perkiraan nilai fungsi objektif yang optimal.



Gambar 1. Grafik trayektori data percobaan

Hasil dari uji coba tidak hanya menampilkan nilai optimal yang dicapai disertai gambar grafik trayektori nilai fungsi objektif, nilai *boundary*, nilai batas atas (*upper bound*), tetapi juga menampilkan tabel yang berisi iterasi, nilai x, nilai fungsi objektif, nilai *boundary*, dan nilai batas atas (*upper bound*). Kolom iterasi disini maksudnya adalah penyelesaian peramasalahan pada sub problem. Sehingga bisa diartikan sebagai jumlah sub problem yang aktif untuk dipertimbangkan sebagai sub problem yang memiliki perkiraan nilai fungsi objektif yang optimal dan nilai x yang optimal dari permasalahan.

Tabel 1. Hasil uji coba

iterasi	x1	x2	f_cost	upper	bound
1	1.95	1.5	-26.7975	-33.1025	-29.95
2	2.37	1.6975	-28.9771	-31.7629	-30.37
3	2.78	1.7975	-30.3476	-31.2124	-30.78
4	3.106	1.858	-31.1201	-31.0919	-31.106

Dari Tabel 1 dapat dilihat bahwa jumlah node yang aktif (dilihat pada kolom iterasi) sebanyak tiga. Dari Tabel 1, fungsi objektif (pada kolom f_cost) yang dicari tidak pernah melebihi nilai *boundary* (pada kolom bound) dan nilai batas atas (pada kolom

upper). Karena jika melebihi maka proses akan dihentikan. Pada kolom Nilai_x menunjukkan nilai vector dari x yang dicari nilai optimalnya.

VI. KESIMPULAN

Berdasarkan aplikasi yang telah dibuat dan beserta uji coba yang telah dilakukan, maka dapat ditarik kesimpulan sebagai berikut :

1. Fungsi pembatas dalam permasalahan kuadratik yang diangkat mempengaruhi jumlah sub problem yang aktif (dapat dilihat melalui jumlah iterasi).
2. Nilai fungsi *boundary* sangat membantu dalam membatasi fungsi objektif dari permasalahan, sehingga lebih mudah didapatkan solusi perkiraan optimal.
3. Pada permasalahan tertentu tidak dapat digambarkan grafik trayektori fungsi objektif karena pada awal iterasi sudah dicapai solusi perkiraan optimal.

PUSTAKA

[1] Bazaraa, M.S, Sherali, H.D, Shetty, C.M, (2006), *Non linear theory and algorithm Third Edition*, Wiley-Interscience

[2] Boyd, S., Gosh, A., Magnani, A., (2003), *Branch and bound method*, Stanford University

[3] Burer S.,Vandenbussche D., (2006), *A Finite branch and bound algorithm for nonconvex quadratic programming via semidefinite relaxation*, Springer- Verlag

[4] Deb, K.; Tewari, R.; Dixit, M.; Dutta, J., (2007), *Finding trade-off solutions close to KKT points using evolutionary multi-objective optimization*, IEEE Congress on Evolutionary Computation, Page(s):2109 - 2116

[5] Nocedal, J., Wright, Stephen J., (1999), *Numerical Optimization*, Springer – Verlag

[6] Taha, A., Hamdy, (1994), *Operation Research An Introduction*, Mc Graw Hill

[7] Wolkowicz H., Anstreicher K., (1998), *On lagrangian relaxation of quadratic matrix constraints*, Department of Combinatories & Optimization, University of Waterloo.