Penerapan Algoritma *Divide and Conquer* pada Perhitungan Nilai Eigen

M. Faizal Hitobeli - 13506057[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]if16057@students.if.itb.ac.id

Abstract — Secara formal, algoritma adalah langkah-langkah dalam menyelesaikan suatu permasalahan secara terstruktur sehingga dapat ditemukannya penyelesaian. Pada dasarnya, dalam ilmu komputer konteks divide and conquer tidak jauh berbeda dengan maknanya dalam bidang politik maupun militer. Terdapat korelasi dalam memecah kekuatan musuh ke dalam bagian-bagian yang lebih kecil. Namun dalam sudut pandang ilmu komputer, kekuatan musuh yang dimaksud adalah masalah yang dihadapi. Dengan pendekatan algoritma divide and conquer, kita dapat mencoba untuk menyelesaikan permasalahan perhitungan nilai eigen. Berikut akan dibahas mengenai penerapan algoritma divide and conquer pada perhitungan nilai eigen. Pada penerpaan algoritma divide and conquer pada perhitungan nilai eigen, didapat hasil yang cenderung variatif, bergantung kepada spesifikasi komputer yang digunakan, namun penggunaan algoritma ini pada proses penghitungan nilai eigen memerlukan waktu yang relatif cepat. Implementasi algoritma dibantu dengan library LAPACK yang ditulis dengan bahasa Fortran 90.

Index Terms — Algortima, divide and conquer, nilai eigen, LAPACK.

I. PENDAHULUAN

Dalam berbagai aktifitas keseharian, seseorang seringkali berhadapan dengan berbagai masalah. Banyak cara yang dapat digunakan untuk memecahkan masalah-masalah tersebut. Namun, pada dasarnya teknik pemecahan masalah yang dihadapi dapat dibagi menjadi tiga kategori, yaitu dengan menggunakan metode heuristik, algoritmik, serta menggunakan gabungan kedua metode yang ada.

Metode heuristik memungkinkan seseorang untuk menemukan solusi dari suatu masalah berdasarkan ketersediaan informasi atas solusi masalah yang sedang dihadapi tersebut. Di lain pihak, metode algoritmik memungkinkan seseorang untuk mendapatkan solusi dari suatu masalah berdasarkan langkah-langkah tertentu. Tidak menutup kemungkinan untuk menggabungkan kedua metode tersebut dalam menyelesaikan suatu persoalan.

Secara formal, algoritma adalah langkah-langkah dalam menyelesaikan suatu permasalahan secara terstruktur sehingga dapat ditemukannya penyelesaian. Sering algoritma dikorelasikan dengan penulisan kode dalam proses pembuatan perangkat lunak komputer. Akan tetapi, algoritma tak hanya dapat digunakan untuk membuat sebuah perangkat lunak komputer saja, melainkan dapat pula menyelesaikan persoalan tertentu dalam suatu lingkup permasalahan.

Dalam bidang ilmu informatika, program yang baik bukan hanya program yang dapat menyelesaikan masalah. Akan tetapi program tersebut juga layaknya dapat menyelesaikan masalah dengan menggunakan algoritma yang efektif. Salah satu algoritma yang cukup sering digunakan untuk menyelesaikan masalah adalah algoritma *Divide and Conquer*. Algoritma ini bekerja dengan cara memecah suatu masalah menjadi beberapa masalah yang lebih kecil, sehingga memungkinkan untuk diselesaikan secara efektif.

Dalam bidang ilmu matematika, sering ditemukan istilah atau konsep yang disebut dengan nilai eigen (*eigenvalue*), dan vektor eigen (*eigenvector*). Nilai eigen dan vektor eigen merupakan konsep yang ada dalam bidang aljabar linear. Salah satu kajian pada aljabar linear adalah transformasi linear, direpresentasikan oleh matriks yang bekerja terhadap vektor.

Nilai eigen dan vektor eigen merupakan salah satu properti yang dimiliki oleh matriks bujur sangkar. Matriks bujur sangkar adalah matriks yang memiliki jumlah baris dan kolom yang sama, contoh matriks M_{2x2}, A_{3x3}. Nilai eigen berperan sangat besar dalam studi persamaan diferensial. Selain itu, nilai eigen juga digunakan secara luas dalam berbagai aplikasi ilmu fisika.

Oleh karena aplikasinya yang cukup luas, perhitungan nilai eigen secara tepat dari sebuah matriks dapat sangat membantu dalam penerapannya. Dengan pendekatan algoritma *divide and conquer*, kita dapat mencoba untuk menyelesaikan permasalahan perhitungan nilai eigen. Berikut akan dibahas mengenai penerapan algoritma *divide and conquer* pada perhitungan nilai eigen.

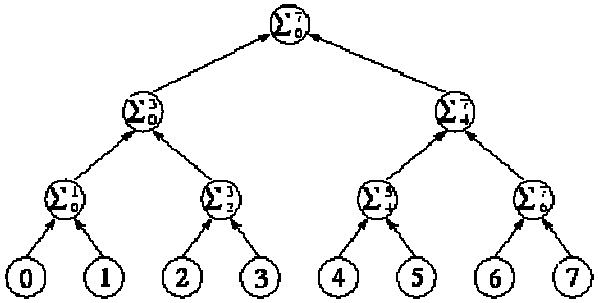
II. LANDASAN TEORI

A. Algoritma *Divide and Conquer*

Istilah *divide and conquer* berasal dari istilah *divide etimpera* yang berasal dari bahasa latin yang sering digunakan dalam konteks militer ataupun politis. Dalam konteks tersebut, *divide and conquer* bermakna bahwa suatu kekuatan yang terkonsentrasi harus dipecah terlebih dahulu agar bisa ditaklukkan. Dengan demikian, kekuatan musuh yang lebih besar daripada kekuatan kita dapat dikalahkan dengan cara memecah kekuatan musuh yang terkonsentrasi tersebut sehingga menjadi kekuatan kecil yang terpisah.

Pada dasarnya, dalam ilmu komputer konteks *divide and conquer* tidak jauh berbeda dengan maknanya dalam bidang politik maupun militer. Terdapat korelasi dalam memecah kekuatan musuh ke dalam bagian-bagian yang lebih kecil. Namun dalam sudut pandang ilmu komputer, kekuatan musuh yang dimaksud adalah masalah yang dihadapi.

Masalah yang didefinisikan dipecah ke dalam dua upa masalah dengan tipe yang sama. Selanjutnya, kedua upa masalah tersebut dipecah masing-masing menjadi dua upa masalah yang lain dan seterusnya sampai masalah tersebut menjadi sangat sederhana, sehingga memungkinkan untuk diselesaikan, atau tidak dapat dipecah lagi. Jika dianalogikan ke dalam bentuk pohon, simpul akar adalah masalah utama, yang selanjutnya memiliki dua cabang pada tingkat 1. Dua cabang tersebut kemudian masing-masing memiliki dua daun, dan begitu seterusnya. Berikut ilustrasi algoritma *divide and conquer* dengan representasi pohon.



Gambar 1 Representasi pohon algoritma *divide and conquer*

Algoritma *divide and conquer* mempunyai tiga elemen dalam pendekatannya. Pertama, *divide*, yaitu memecah masalah yang ada menjadi upa masalah. Kedua, *conquer*, yaitu menyelesaikan masalah. Ketiga, *combine*, menyatukan hasil masalah yang didapat dari setiap upa masalah untuk mendapatkan solusi akhirnya.

Berikut contoh *pseudo code* algoritma *divide and conquer*:

```
procedure divide (general parameters)
  if (trivial) then //condition of triviality
        solve it
  else //not trivial then
        divide into sub-problems
        solve them recursively //call this procedure
        until it becomes trivial
        combine them to get the final solution
  end if
end procedure
```

B. Definisi Nilai Eigen

Definisi nilai eigen secara formal adalah, jika A adalah transformasi linear, vektor non-null x adalah vektor eigen dari A jika ada skalar λ sedemikian sehingga

$$\mathbf{Ax} = \lambda\mathbf{x}.$$

Maka, λ dikatakan nilai eigen dari A sesuai dengan x, dan untuk semua vektor eigen berperilaku seperti x.

Dalam aljabar linear, terdapat dua macam objek, yaitu skalar yang merupakan hanya angka, dan vektor yang memiliki panjang dan arah (atau lebih tepatnya vektor adalah anggota ruang vektor). Vektor dapat dianggap sebagai anak panah. Pada fungsi biasa aljabar, fungsi yang paling penting dalam aljabar linier disebut transformasi linear, dan transformasi linear biasanya diberikan oleh matriks, array dari angka-angka. Jadi suatu fungsi pada aljabar linear direpresentasikan tidak dengan peulisan f (x), melainkan dengan penulisan M (v) atau hanya Mv, dimana M adalah matriks dan v adalah vektor.

Berikut contoh pencarian nilai eigen pada suatu matriks. Misal terdapat suatu matriks A sebagai berikut:

$$A = \begin{bmatrix} 2 & -4 \\ -1 & -1 \end{bmatrix}$$

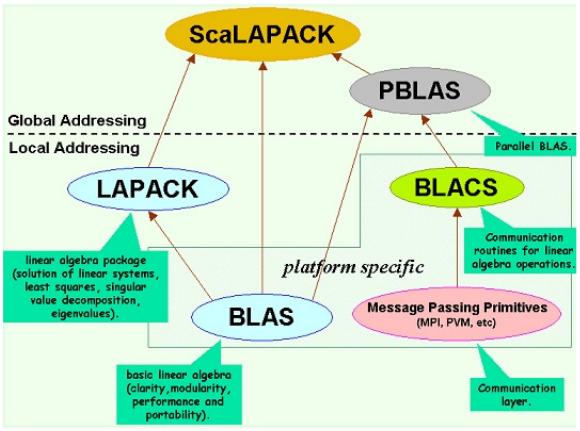
Maka, nilai eigen matriks tersebut dapat dicari dengan perhitungan:

$$\begin{aligned} p(\lambda) &= \det \begin{bmatrix} 2-\lambda & -4 \\ -1 & -1-\lambda \end{bmatrix} \\ &= (2-\lambda)(-1-\lambda)-(-4)(-1) \\ &= \lambda^2 - \lambda - 6 \\ &= (\lambda-3)(\lambda+2). \end{aligned}$$

Jadi, nilai eigen yang didapat adalah: $\lambda_1 = 3$ dan $\lambda_2 = -2$

C. LAPACK

Liniear Algebra PACKage (LAPACK) adalah suatu *library* perangkat lunak yang disediakan untuk melayani perhitungan aljabar linear numerik. *Library* ini menyediakan kumpulan *routines* untuk memecahkan sistem persamaan linear, nilai eigen, dan dekomposisi nilai singular. LAPACK ditulis dalam bahasa Fortran 77. Pada perkembangannya LAPACK sekarang ditulis dalam bahasa Fortran 90. Berikut merupakan ilustrasi susunan komponen di dalam *library* LAPACK.



Gambar 2 Komponen *library* LAPACK

LAPACK dapat dilihat sebagai kelanjutan dari *linear equation package*. LAPACK bergantung pada *Basic Linear Algebra Subprogram* (BLAS) yang secara efektif mengeksploitasi *cache* pada arsitektur komputer modern yang berbasis *cache*. LAPACK berada di baah lisensi *three-clause* BSD, yaitu sebuah lisensi perangkat lunak gratis dengan beberapa batasan.

III. PENERAPAN ALGORITMA *DIVIDE AND CONQUER*

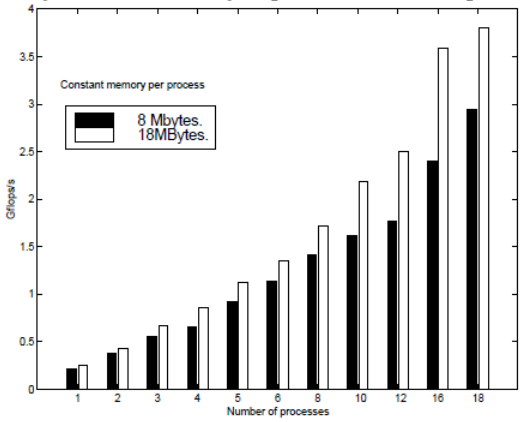
Algoritma *divide and conquer* diimplementasikan dalam kode yang menggunakan LAPACK *serial routines*. Penulis menggunakan PBLAS (BLAS paralel) routines PxGEMM untuk melakukan perkalian paralel pada matriks. Kode ini memiliki biaya komunikasi yang tumbuh dengan akar kuadrat dari jumlah proses, sehingga mengarah kepada efisiensi dan skalabilitas yang. Semua komunikasi dilakukan dengan menggunakan BLACS (Basic Linear Algebra Communication Subprograms), yang bertujuan untuk memberikan lapisan, portabel linier-aljabar-khusus untuk komunikasi. BLACS tersedia untuk berbagai mesin memori terdistribusi dan untuk kedua PVM (Paralel Virtual Machine) dan MPI (Message Passing Interface). Hal ini memungkinkan kode algoritma *divide and conquer* yang tertulis dapat dijalankan secara portabel pada platform paralel, termasuk jaringan workstation yang mendukung PVM atau MPI.

Berikut potongan implementasi kode algoritma *divide and conquer* dengan *library* LAPACK dengan bahasa Fortran 90.

```
subroutine PxSTEDC( N, NB, D, E, Q, ... )
subroutine PxLAED0( N, NB, D, E, Q, ... )
 TSUBPBS = (N-1)/NB +1
 while (TSUBPBS > 1 )
       for i = 1:TSUBPBS/2
           call PxLAED1(N, NB, i, TSUBPBS, D, Q, ...)
       end
 TSUBPBS = TSUBPBS / 2
 end
subroutine PxLAED1( N, NB, i, TSUBPBS, D, Q, ...)
call PxLAED2( N, NB, 1 TSUBPBS, K, D, Q, Z, ... )
call PxLAED3( N, K, D, Z, U, ... )
call PxGEMM( Q1, U1, ...)
call PxGEMM( Q2, U2, ...)
```

IV. ANALISIS HASIL PENERAPAN

Dengan menerpakan algoritma *divide and conquer* pada perhitungan nilai eigen, didapat hasil yang cenderung variatif, bergantung kepada spesifikasi komputer yang digunakan. Berikut ilustrasi hasil perhitungan dalam bentuk graf pada dua buah komputer.



Gambar 3 Perhitungan nilai eigen pada dua buah komputer

Penggunaan algoritma ini pada proses penghitungan nilai eigen memerlukan waktu yang relatif cepat. Dalam hal ini, satu komputer menggunakan memori proses yang dipatok pada 8 Mbyte dan 18 Mbyte pada aplikasi.

V. KESIMPULAN

Untuk saat ini, algoritma *divide and conquer* merupakan salah satu algoritma yang tercepat untuk menemukan semua nilai eigen dan vektor eigen dari suatu matriks simetris dengan yang ukuran relatif besar dan padat. Dalam tulisan ini, algoritma *divide and conquer* dapat secara efisien paralelkan pada mesin dengan memori terdistribusi. Dengan menggunakan pendekatan teorema Lowner, eigendecompositions numerik yang baik diperoleh pada semua situasi. Dari sudut pandang waktu eksekusi, hasil menunjukkan waktu eksekusi yang lebih baik untuk kebanyakan kasus bila dibandingkan dengan waktu eksekusi paralel QR dan pembelahan, diikuti oleh iterasi inverse yang tersedia pada *library* ScaLAPACK.

Hasil Kinerja pada dua komputer menunjukkan skalabilitas dan portabilitas algoritma yang diimplementasikan. efisiensi yang baik terutama diperoleh dengan mengeksploitasi paralelisme data yang melekat pada algoritma ini Untuk ini, kami memusatkan usaha kami pada implementasi yang baik dari proses transformasi kembali untuk mencapai kecepatan maksimum-up untuk perkalian matriks.

REFERENSI

[1] Munir, Rinaldi. 2005. *Diktat Kuliah Strategi Algoritmik IF2251 Strategi Algoritmik*. Departemen Teknik Informatika ITB.

[2] J. Rutter, 1994. *A Serial Implementation of Cuppen's Divide and Conquer Algorithm for the Symmetric Eigenvalue Problem*, Technical Report CS-94-225, Department of Computer Science, University of Tennessee, Knoxville, TN, LAPACK Working Note 69.

[3] F. Tisseur and J. J. Dongarra, 1998. *Parallelizing the Divide and Conquer Algorithm for the Symmetric Tridiagonal Eigenvalue Problem on Distributed Memory Architectures*, Technical Report CS-98-381, Department of Computer Science, University of Tennessee, Knoxville, TN, LAPACK Working Note 132.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 7 Desember 2010

M. Faizal Hitobeli
13506057