Strategi V1

Dibuat Oleh:

Friskadini Ismayanti (1301198496)
Muhammad Faisal Amir (1301198497)
Ridho Maulana Cahyudi (1301198515)

TELKOM UNIVERSITY
FAKULTAS INFORMATIKA
2020

Algoritma Divide And Conquer

Pengertian

Algoritma Divide And Conquer Merupakan Algoritma Yang Sangat Populer Di Dunia Ilmu Komputer. Divide And Conquer Merupakan Algoritma Yang Berprinsip Memecah-Mecah Permasalahan Yang Terlalu Besar Menjadi Beberapa Bagian Kecil Sehingga Lebih Mudah Untuk Diselesaikan. Langkah-Langkah Umum Algoritma Divide And Conquer :

Divide : Membagi Masalah Menjadi Beberapa Upa-Masalah Yang Memiliki Kemiripan Dengan Masalah Semula Namun Berukuran Lebih Kecil (Idealnya Berukuran Hampir Sama).

Conquer : Memecahkan (Menyelesaikan) Masing-Masing Upa-Masalah (Secara Rekursif).

Combine : Menggabungkan Solusi Masing-Masing Upa-Masalah Sehingga Membentuk Solusi Masalah Semula.

Objek Masalah Yang Di Bagi Adalah Masukan (Input) Atau Instances Yang Berukuran N: Tabel (Larik), Matriks, Dan Sebagainya, Bergantung Pada Masalahnya. Tiap-Tiap Upa-Masalah Mempunyai Karakteristik Yang Sama (The Same Type) Dengan Karakteristik Masalah Asal, Sehingga Metode Divide And Conquer Lebih Natural Diungkapkan Dalam Skema Rekursif. Sesuai Dengan Karakteristik Pembagian Dan Pemecahan Masalah Tersebut, Maka Algoritma Ini Dapat Berjalan Baik Pada Persoalan Yang Bertipe Rekursif (Perulangan Dengan Memanggil Dirinya Sendiri).

Dengan Demikian, Algoritma Ini Dapat Diimplementasikan Dengan Cara Iteratif (Perulangan Biasa), Karena Pada Prinsipnya Iteratif Hampir Sama Dengan Rekursif. Salah Satu Penggunaan Algoritma Ini Yang Paling Populer Adalah Dalam Hal Pengolahan Data Yang Bertipe Array (Elemen Larik). Mengapa ? Karena Pengolahan Array Pada Umumnya Selalu Menggunakan Prinsip Rekursif Atau Iteratif.

Penggunaan Secara Spesifik Adalah Untuk Mencari Nilai Minimal Dan Maksimal Serta Untuk Mengurutkan Elemen Array. Dalam Hal Pengurutan Ini Ada Empat Macam Algoritma Pengurutan Yang Berdasar Pada Algoritma Divide And Conquer, Yaitu Merge Sort, Insert Sort, Quick Sort, Dan Selection Sort. Merge Sort Dan Quick Sort Mempunyai Kompleksitas Algoritma $O(N\ ^{2}Log\ N)$. Hal Ini Lebih Baik Jika Dibandingkan Dengan Pengurutan Biasa Dengan Menggunakan Algoritma Brute Force.

Skema Umum Algoritma Divide And Conquer :

```
prosedure DIVIDE_n_CONQUER(input n : integer)
{ Masukan: masukan yang berukuran n
 Keluaran: solusi dari masalah semula
}
Deklarasi
        r, k : integer
Algoritma
    if n ≤ n0 then { masalah sudah cukup kecil }
        SOLVE sub-masalah yang berukuran n ini
    else
        Bagi menjadi r sub-masalah, masing-masing
            Berukuran n/k
        for masing-masing dari r upa-masalah do
            DIVIDE_n_CONQUER(n/k)
        endfor
        COMBINE solusi dari r sub-masalah menjadi
            solusi masalah semula
    endif
```

Penerapan Algoritma

Pemecahan Masalah Convex Hull Dengan Algoritma Divide And Conquer

Pada Penyelasaian Masalah Pencarian Convex Hull Dengan Menggunakan Algoritma Divide And Conquer, Hal Ini Dapat Dipandang Sebagai Generalisasi Dari Algoritma Pengurutan Merge Sort. Berikut Ini Merupakan Garis Besar Gambaran Dari Algoritmanya:

- Pertama-Tama Lakukan Pengurutan Terhadap Titik-Titik Dari Himpunan S Yang Diberika Berdasarkan Koordinat Absis-X, Dengan Kompleksitas Waktu O(N Log N).
- Jika |S| ≤ 3, Maka Lakukan Pencarian Convex Hull Secara Brute-Force Dengan Kompleksitas Waktu O(1). (Basis).
- Jika Tidak, Partisi Himpunan Titik-Titik Pada S Menjadi 2 Buah Himpunan A Dan B, Dimana A Terdiri Dari Setengah Jumlah Dari |S| Dan Titik Dengan Koordinat Absix-X Yang Terendah Dan B Terdiri Dari Setengah Dari Jumlah |S| Dan Titik Dengan Koordinat Absis-X Terbesar.
- Secara Rekursif Lakukan Penghitungan Terhadap HA = Conv(A) Dan HB = Conv(B).
- Lakukan Penggabungan (Merge) Terhadap Kedua Hull Tersebut Menjadi Convex Hull, H, Dengan Menghitung Da Mencari Upper Dan Lower Tangents Untuk HA Dan HB Dengan Mengabaikan Semua Titik Yang Berada Diantara Dua Buah Tangen Ini.

Permasalahan Convex Hull Adalah Sebuah Permasalahan Yang Memiliki Aplikasi Terapan Yang Cukup Banyak, Seperti Pada Permasalahan Grafika Komputer, Otomasi Desain, Pengenalan Pola (Pattern Recognition), Dan Penelitian Operasi. Divide And Conquer Adalah Metode Pemecahan

Masalah Yang Bekerja Dengan Membagi Masalah Menjadi Beberapa Upa-Masalah Yang Lebih Kecil, Kemudian Menyelesaikan Masing-Masing Upa-Masalah Tersebut Secara Independent, Dan Akhirnya Menggabungkan Solusi Masing-Masing Upa-Masalah Sehingga Menjadi Solusi Dari Masalah Semula.

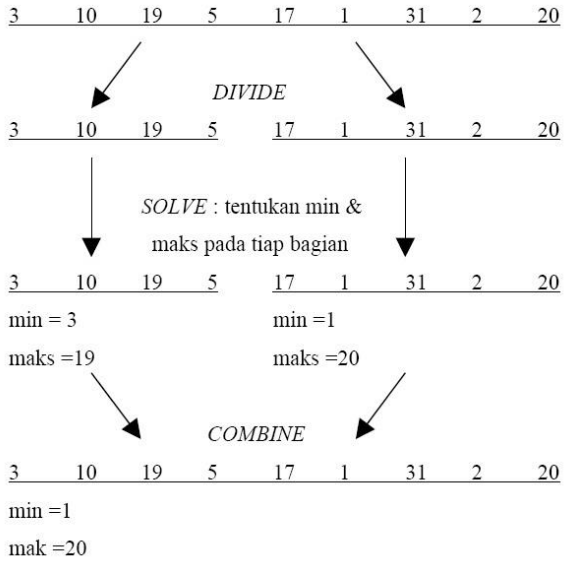
Algoritma Divide And Conquer Merupakan Salah Satu Solusi Dalam Penyelesaian Masalah Convex Hull. Algoritma Ini Ternyata Memiliki Kompleksitas Waktu Yang Cukup Kecil Dan Efektif Dalam Menyelesaikan Permasalahan Ini (Jika Dibandingkan Algoritma Lain). Selain Itu Juga, Algoritma Ini Dapat Digeneralisasi Untuk Permasalahan Convex Hull Yang Berdimensi Lebih Dari 3.

Persoalan Minimum Dan Maksimum (Minmaks)

Persoalan : Misalnya Diketahui Table A Yang Berukuran N Eleman Sudah Berisi Nilai Integer. Kita Ingin Menentukan Nilai Minimum Dan Nilai Maksimum Sekaligus Di Dalam Table Tersebut. Misalkan Tabel A Berisi Elemen-Elemen Sebagai Berikut :

3 10 19 5 17 1 31 2 20

Ide Dasar Algoritma Secara Divide And Conquer :



Ukuran Table Hasil Pembagian Dapat Dibuat Cukup Kecil Sehingga Mencari Minimum Dan Maksimum Dapat Diselesaikan (SOLVE) Secara Lebih Mudah. Dalam Hal Ini, Ukuran Kecil Yang Dipilih Adalah 1 Elemen Atau 2 Elemen.

Algoritma Minmaks :

Untuk Kasus N = 1 Atau N = 2,

SOLVE : Jika N = 1, Maka Min = Maks = An. Jika N = 2, Maka Bandingkan Kedua Elemen Untuk Menentukan Min Dan Maks.

Untuk Kasus N > 2,

DIVIDE : Bagi Dua Table A Secara Rekursif Menjadi Dua Bagian Yang Berukuran Sama, Yaitu Bagian Kiri Dan Bagian Kanan.

CONQUER : Terapkan Algoritma Divide And Conquer Untuk Masing-Masing Bagian, Dalam Hal Ini Min Dan Maks Dari Table Bagian Kiri Dinyatakan Dalam Peubah Min1 Dan Maks1, Dan Min Dan Maks Dari Table Bagian Kanan Dinyatakan Dalam Peubah Min2 Dan Maks2.

COMBINE : Bandingkan Min1 Dan Min2 Untuk Menentukan Min Table A, Serta Bandingkan Maks1 Dan Maks2 Untuk Menentukan Maks Table A.

Optimasi Konversi Bilangan Desimal Ke Biner

Salah Satu Cara Optimasi Yang Bias Kita Lakukan Adalah Membagi Bilangan Decimal Yang Hendak Diubah Dengan Angka 8 (Bukan 2). Di Sinilah Prinsip Algoritma Divide And Conquer Kita Gunakan Untuk Melakukan Optimasi. Kita Pecah-Pecah Angka Decimal Yang Akan Kita Gunakan Dengan Cara Membaginya Dengan Angka 8 Secara Berulang. Angka-Angka Sisa Pembagian Yang Kita Peroleh Kemudian Kita Ubah Ke Dalam Bilangan Biner Sebelum Kita Gabungkan Menjadi Hasil Jawaban.

Karena Angka Pembagi Yang Kita Pakai Adalah 8 (23), Maka Kita Dapat Mengurangijumlah Pembagian Yang Kita Lakukan Menjadi ± 1/3 Dari Jumlah Semula. Hal Ini Tentu Saja Akan Sangat Berpengaruh Pada Kinerja Dan Waktu Yang Diperlukan Oleh Computer Mengingat Proses Pembagian Merupakan Salah Satu Proses Yang Cukup Rumit.

Tentu Saja Optimasi Ini Harus Kita Bayar Dengan Menangani Konversi Bilangan Octal Ke Biner. Akan Tetapi Jika Kita Gunakan Teknik Perbandingan (Tanpa Harus Melakukan Konversi Secara Manual), Maka Proses Ini Akan Menjadi Sangat Cepat Dan Mudah. Penerapan Algoritma Ini Adalah Dengan Menggunakan Sintaks Case Of. Begitu Juga Dengan Permasalahan Pemakaian Memori (Kompleksitas Ruang) Yang Lebih Besar Yang Muncul Akibat Penggunaan Algoritma Rekursif. Karena Pada Proses Rekursif-Nya Kita Tidak Banyak Menggunakan Variable Yang Memerlukan Tempat Yang Begitu Besar, Maka Hal Ini Bias Kita Abaikan. Dengan Penggunaan Optimasi Ini, Maka Seharusnya Proses Konversi Akan Lebih Cepat Karena Pemangkasan Jumlah Pembagian Yang Dilakukan.

Skema Procedur Utama Konversi Dengan Optimasi

```
procedure KONVERSI_2(input angka : long)
{ Keluaran digunakan string karena pada umumnya kapasitas
  untuk bilangan sangat terbatas
}
Deklarasi
        string hasil ;
Algoritma
        hasil = "" ;
        KONVERSI_DnC(angka, hasil) ;
```

Skema Procedur Rekursif Dengan Menerapkan Algoritma Divide And Conquer

```
procedure KONVERSI_DnC( input angka : long,
                input/output hasil : string )
Deklarasi
        temp1, temp2  : long ;
Algoritma
        if  (angka <= 8)  then
          case  (angka)  of  :
                0 : hasil <- hasil + "000" ;
                1 : hasil <- hasil + "001" ;
                2 : hasil <- hasil + "010" ;
                3 : hasil <- hasil + "011" ;
                4 : hasil <- hasil + "100" ;
                5 : hasil <- hasil + "101" ;
                6 : hasil <- hasil + "110" ;
                7 : hasil <- hasil + "111" ;
      else
          temp1 <- angka div 8
                KONVERSI_DnC(temp1, hasil) ;
          temp2 <- angka mod 8
                KONVERSI_DnC(temp2, hasil) ;
```

Kompleksitas Waktu Algoritma :

T(N) = O(N/3)

Dengan N Menyatakan Eksponen Terkecil Dari 2 Yang Mempunyai Nilai 2n Lebuh Besar Dari Angka Decimal.

Algoritma Konversi System Bilangan Dengan Menggunakan Algoritma Dengan Optimasi Yang Menerapkan Algoritma Divide And Conquer Lebih Mangkus Daripada Algoritma Konversi Dengan Metode Pembagian Sisa Biasa Jika Dilihat Dari Segi Kompleksitas Waktunya. Hanya Saja Optimasi Ini Diimbangi Dengan Kenaikan Pada Kompleksitas Ruangnya, Meskipun Pengaruhnya Tidak Sebesar Optimasi Yang Kita Lakukan.

Mencari Pasangan Titik Yang Jaraknya Terdekat (Closest Pair)

Persoalan : Diberikan Himpunan Titik, P, Yang Terdiri Dari N Buah Titik, (Xi,Yi), Pada Bilangan 2-D. Tentukan Jarak Terdekat Antara Dua Buah Titik Di Dalam Himpunan P. Jarak Dua Buah Titik P1 = (X1, Y1) Dan P2 = (X2, Y2) :

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Penyelesaian Dengan Algoritma Divide And Conquer :

A. Asumsi : N = 2k Dan Titik-Titik Diurut Berdasarkan Absis (X)

B. Algoritma Closest Pair :

- SOLVE : Jika N = 2, Maka Jarak Kedua Titik Dihitung Langsung Dengan Rumus Euclidean.
- DIVIDE : Bagi Titik-Titik Itu Ke Dalam Dua Bagian, Pleft Dan Pright, Setiap Bagian Mempunyai Jumlah Titik Yang Sama
- CONQUER :Secara Rekursif, Terapkan Algoritma D-And-C Pada Masingmasing Bagian.
- Pasangan Titik Yang Jaraknya Terdekat Ada Tiga Kemungkinan Letaknya :
    1. Pasangan Titik Terdekat Terdapat Di Bagian Pleft.
    2. Pasangan Titik Terdekat Terdapat Di Bagian Pright.
    3. Pasangan Titik Terdekat Dipisahkan Oleh Garis Batas L, Yaitu Satu Titik Di Pleft Dan Satu Titik Di Pright.

Jika Kasusnya Adalah (C), Maka Lakukan Tahap COMBINE Untuk Mendapatkan Jarak Dua Titik Terdekat Sebagai Solusi Persoalan Semula.

Kompleksitas Problem

Masalah Penugasan Yang Diperumum Masalah Penugasan Yang Diperumum Dideskripsikan Sebagai Masalah Pemasangan Sejumlah Tugas Kepada Sejumlah Agen Dengan Ketentuan Setiap Agen Dapat Dipasangkan Pada Lebih Dari Satu Tugas Dan Terdapat Batasan Sumber Daya Untuk Setiap Agen. Model Optimisasi Masalah Penugasan Adalah Sebagai Berikut.

Meminimumkan:

$$z = \sum_{i \in I} \sum_{j \in J} c_{ij}\, x_{ij}$$

Terhadap:

$$\sum_{j \in J} r_{ij}\, x_{ij} \le b_i\,,\ i \in I\,,$$

$$\sum_{i \in I} x_{ij} = 1\,,\ j \in J\,,$$

$$x_{ij} \in \{0,1\}.$$

Metode Penyelesaian Setiap Masalah Penugasan Pada Dasarnya Dapat Diselesaikan Dengan Menggunakan Algoritma Branch-And-Bound, Karena Algoritma Branch-And-Bound Biasa Digunakan

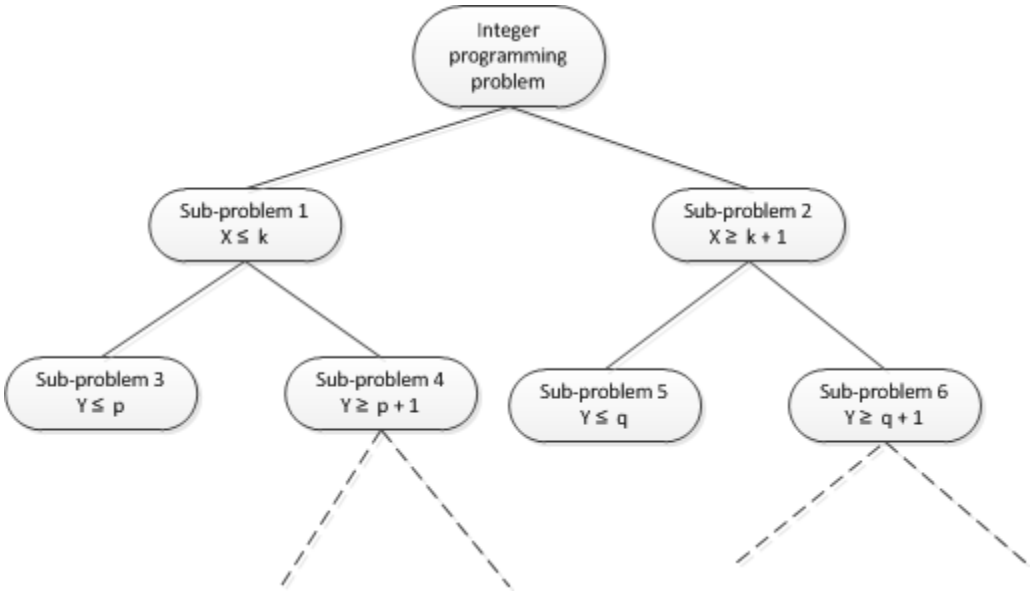
Untuk Menyelesaikan Masalah Integer Linear Programming (ILP) Melalui LP Relaksasi Dengan Membuat Sub - Sub Masalah Yang Harus Diselesaikan. Namun Seringkali, Penyelesaian Masalah ILP Dengan Metode Branch-And-Bound Membutuhkan Percabangan Yang Sangat Banyak Sampai Diperoleh Solusi Optimal Akibatnya Dibutuhkan Waktu Yang Lama Untuk Menemukan Solusi Optimal. Dalam Penelitian Ini Dilakukan Sejumlah Revisi Pada Langkah Penyelesaian Model. Tujuan Dari Penggunaan Algoritma Branch-And-Bound Yang Direvisi Ini Adalah Untuk Efisiensi Waktu Penyelesaian Masalah Penugasan Sehingga Solusi Optimal Dapat Diperoleh Dengan Waktu Yang Lebih Cepat. Algoritma Branch-And-Bound Yang Direvisi Ini Pada Dasarnya Adalah Salah Satu Tipe Algoritma Branch-And-Bound Biasa Di Mana Nilai Batasnya Dihitung Dengan Menyelesaikan Masalah Knapsack Biner Dalam Sub – Sub Masalah Yang Ada (G.T. Ross Dan R.M. Soland, 1973, Hlm.93).

Analisa Order Growth

Algoritma Branch And Bound

Selain Programa Linier, Bidang Operations Research Mempunyai Satu Model Umum Yang Sangat Sering Dipakai. Model Ini Adalah Model Programa Linier Dengan Tambahan Kendala Integer Yang Biasa Disebut Dengan Integer Programming (IP). Kendala Integer Yang Dimaksud Adalah Kendala Di Mana Beberapa (Atau Semua) Variabel Keputusan Harus Bernilai Angka Integer.

Metode Dasar Yang Dapat Digunakan Untuk Menyeleseikan Permasalahan Programa Integer Adalah Dengan Algoritma Branch And Bound. Algoritma Ini Disebut Algoritma Branch Karena Algoritma Ini Akan Membagi Permasalahan Prorgama Integer Menjadi Cabang-Cabang Permasalahan Seperti Ditunjukkan Pada Gambar Di Bawah Ini. Cabang-Cabang Permasalahan Akan Berbentuk Programa Linear. Algoritma Ini Juga Membatasi (Bound) Pencarian Pada Percabangan Yang Menjanjikan Saja.



Percabangan Dilakukan Dengan Memeriksa Variabel Mana Yang Belum Memenuhi Kendala Integer. Jika Variabel Seperti Ini Ditemui, Maka Sub-Problem Akan Dibuat Dengan Menyalin Parent-Problem Ditambah Dengan Kendala Upper Bound Atau Lower Bound Dengan Nilai Integer Bagi Variabel Tersebut.

Misalkan Kita Mempunyai Permasalahan Sebagai Berikut:

$$max\ 3x + 4y$$
$$s.t.$$
$$2x + 7y \leq 17$$
$$5x + 2y \leq 21$$
$$x, y \in\ integer$$

Jika Kita Selesaikan Dengan Mengabaikan Kendala Integer, Maka Ia Memiliki Solusi Optimum Dengan Nilai Objective 16.48, X = 3.64 Dan Y = 1.38.

Ternyata Solusi Tersebut Belum Memenuhi Kendala Integer Sehingga Kita Perlu Membuat Percabangan. Sebagai Langkah Awal, Kita Ingin Memenuhi Persyaratan Integer Untuk Variabel X, Sehingga Kita Membuat Dua Sub-Problem Berikut:

Sub-Problem 1:

$$max\ 3x + 4y$$
$$s.t.$$
$$2x + 7y \leq 17$$
$$5x + 2y \leq 21$$
$$x \leq 3$$

Dan Sub-Problem 2:

$$max\ 3x + 4y$$
$$s.t.$$
$$2x + 7y \leq 17$$
$$5x + 2y \leq 21$$
$$x \geq 4$$

Semua Percabangan Yang Menjanjikan Akan Diperiksa Sehingga Metode Ini Dapat Dipastikan Dapat Memeriksa Kondisi Keoptimalan Suatu Permasalahan Programa IP. Metode Ini Akan Cukup Memakan Waktu Komputasi Yang Tinggi Karena Percabangan Tumbuh Secara Eksponensial Seiring Dengan Bertambahnya Kendala Integer.

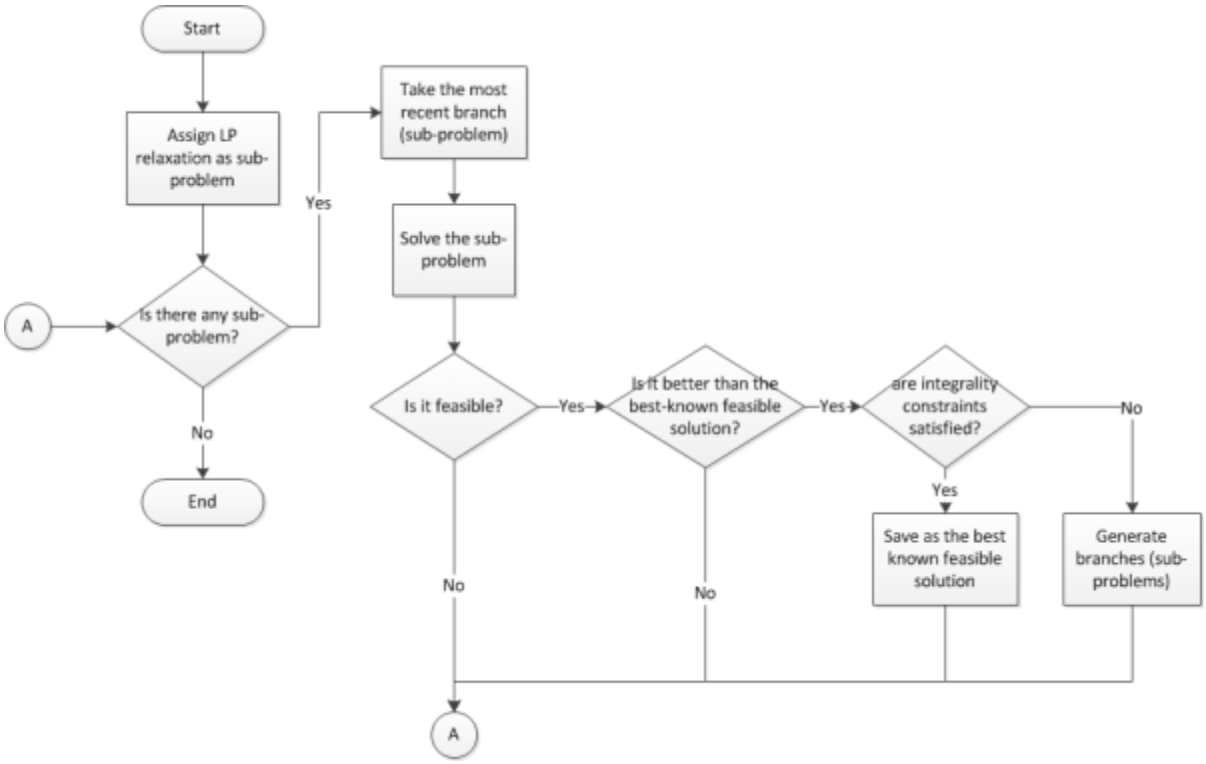
Agar Lebih Efisien, Tidak Semua Percabangan Akan Diperiksa. Ada Beberapa Kaidah Yang Akan Memberi Panduan Apakah Percabangan Akan Diteruskan Untuk Level Berikutnya Atau Tidak. Tiga Keadaan Di Mana Percabangan Tidak Akan Diteruskan Pada Level Berikutnya Adalah:

Sub-Problem Tersebut Tidak Feasibel Atau Unbounded

Sub-Problem Tersebut Tidak Lebih Baik Dari Pada Solusi Feasible (Feasible Artinya Memenuhi Kendala Integer) Yang Sudah Diketahui

Sub-Problem Tersebut Memenuhi Kendala Integer. Jika Sub Problem Ini Lebih Baik Dari Solusi Incumbent, Maka Solusi Incumbent Akan Digantikan Dengan Solusi Yang Baru Ini.

Untuk Lebih Lengkapnya, Kami Menyediakan Bagan Algoritma Branch And Bound Berikut:



Kompleksitas Problem

Pemecahan Masalah Convex Hull Dengan Algoritma Divide And Conquer

Pada Penyelasaian Masalah Pencarian Convex Hull Dengan Menggunakan Algoritma Divide And Conquer, Hal Ini Dapat Dipandang Sebagai Generalisasi Dari Algoritma Pengurutan Merge Sort. Berikut Ini Merupakan Garis Besar Gambaran Dari Algoritmanya:

- Pertama-Tama Lakukan Pengurutan Terhadap Titik-Titik Dari Himpunan S Yang Diberika Berdasarkan Koordinat Absis-X, Dengan Kompleksitas Waktu O(N Log N).
- Jika $|S| \leq 3$, Maka Lakukan Pencarian Convex Hull Secara Brute-Force Dengan Kompleksitas Waktu O(1). (Basis).
- Jika Tidak, Partisi Himpunan Titik-Titik Pada S Menjadi 2 Buah Himpunan A Dan B, Dimana A Terdiri Dari Setengah Jumlah Dari $|S|$ Dan Titik Dengan Koordinat Absix-X Yang Terendah Dan B Terdiri Dari Setengah Dari Jumlah $|S|$ Dan Titik Dengan Koordinat Absis-X Terbesar.
- Secara Rekursif Lakukan Penghitungan Terhadap HA = Conv(A) Dan HB = Conv(B).
- Lakukan Penggabungan (Merge) Terhadap Kedua Hull Tersebut Menjadi Convex Hull, H, Dengan Menghitung Da Mencari Upper Dan Lower Tangents Untuk HA Dan HB Dengan Mengabaikan Semua Titik Yang Berada Diantara Dua Buah Tangen Ini.

Permasalahan Convex Hull Adalah Sebuah Permasalahan Yang Memiliki Aplikasi Terapan Yang Cukup Banyak, Seperti Pada Permasalahan Grafika Komputer, Otomasi Desain, Pengenalan Pola

(Pattern Recognition), Dan Penelitian Operasi. Divide And Conquer Adalah Metode Pemecahan Masalah Yang Bekerja Dengan Membagi Masalah Menjadi Beberapa Upa-Masalah Yang Lebih Kecil, Kemudian Menyelesaikan Masing-Masing Upa-Masalah Tersebut Secara Independent, Dan Akhirnya Menggabungkan Solusi Masing-Masing Upa-Masalah Sehingga Menjadi Solusi Dari Masalah Semula.

Analisa Order Growth

Referensi

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2006-2007/Algoritma%20Branch%20and%20Bound.pdf

https://ejournal.upi.edu/index.php/JEM/article/viewFile/9596/5892

https://www.researchgate.net/publication/319273356_Aplikasi_Algoritma_Branch_and_Bound_Untuk_Optimasi_Jalur_Pemadam_Kebakaran_Kota_Yogyakarta

https://andikafisma.wordpress.com/algoritma-divide-and-conquer/

http://aren.cs.ui.ac.id/sda/resources/sda2010/04_analisis.pdf