Penerapan Algoritma *Branch and bound* pada Masalah *Integer Knapsack*

Muhammad Fiqri Muthohar - 13506084

Program Studi Teknik Informatika , Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung
Jl. Ganesha 10 Bandung 40132
e-mail: fiqri@arc.itb.ac.id

ABSTRAK

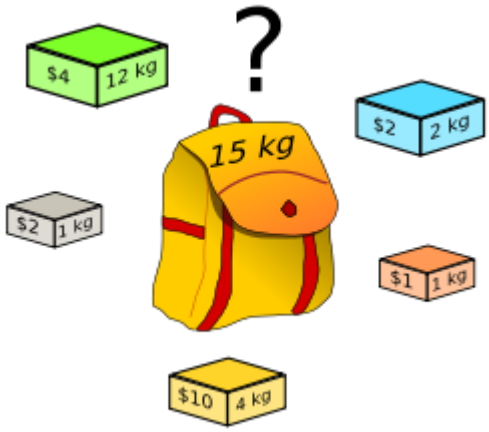
Masalah *integer knapsack* adalah masalah di mana kita dihadapkan pada persoalan optimasi pada pemilihan benda yang dapat dimasukkan ke dalam sebuah wadah yang memiliki keterbatasan ruang dan daya tampung namun benda yang dimasukkan ke dalam wadah tersebut haruslah tetap utuh satu benda, tidak dapat berupa fraksi dari benda tersebut. Dengan adanya optimasi dalam pemilihan benda yang akan dimasukkan ke dalam wadah tersebut diharapkan dapat dihasilkan keuntungan yang maksimum dari pemilihan benda yang dilakukan. Benda-benda dalam permasalahan ini memiliki nilai berat, volum, harga, atau sebuah nilai yang utuh yang digunakan sebagai suatu alat untuk menentukan prioritasnya dalam proses pemilihan tersebut. Dan wadah yang dimaksud di sini juga memiliki nilai konstanta yang merupakan nilai pembatas untuk benda-benda yang akan dimasukkan ke dalam wadah tersebut. Sehingga harus diambil sebuah cara memasukkan benda-benda tersebut ke dalam wadah sehingga menghasilkan hasil optimum tetapi tidak melebihi kemampuan wadah untuk menampungnya.
Dalam makalah ini akan dibahas penerapan salah satu cara untuk menyelesaikan masalah *integer knapsack* yang ada yaitu dengan algoritma *branch and bound*.

Kata kunci: *integer knapsack, branch and bound*.

1. PENDAHULUAN

Masalah integer knapsack adalah masalah di mana kita dihadapkan dengan persoalan optimasi pemilihan benda untuk dimasukkan ke dalam sebuah wadah yang memiliki keterbatasan ruang dan daya tamping tetapi benda yang akan dimasukkan ke dalam wadah tersebut haruslah tetap dalam keadaan utuh bukan merupakan fraksi dari benda tersebut. Masing-masing benda yang ada memiliki sebuah nilai berupa berat, volume, harga, atau nilai lainnya yang dapat dipakai sebagai penentu dalam proses pemilihannya. Sedangkan wadah memiliki sebuah nilai konstanta yang dimilikinya dan merupakan sebuah pembatas dalam proses pemilihan benda untuk dapat dimasukkan ke dalam wadah tersebut. Pada akhir proses diinginkan hasil yang optimum di mana memiliki nilai keuntungan yang paling besar yang dapat dicapai dengan benda yang berada pada wadah tersebut. Sehingga harus diambil sebuah cara memasukkan benda-benda ke dalam wadah sehingga dapat menghasilkan hasil yang optimum.

Dalam makalah ini akan dibahas penerapan salah satu cara untuk menyelesaikan masalah *integer knapsack* yang ada yaitu dengan algoritma *branch and bound*.



Gambar 1. Ilustrasi integer knapsack problem

2. METODE

Pada bab ini akan dibahas secara lebih mendalam hal-hal yang berkaitan dengan integer knapsack dan metode algoritma *branch and bound*.

2.1 Integer Knapsack

Dalam permasalah ini diumpamakan terdapat sebuah benda terhingga sejumlah *n* buah benda, b_1 sampai b_n, dan sebuah wadah yang memiliki daya tampung maksimal senilai *K*. Setiap benda memiliki memiliki bobot w_i dengan nilai keuntungan p_i. Objektif dari permasalahan ini adalah bagaimana memilih objek-objek yang dimasukkan ke dalam wadah sehingga tidak melebihi kapasitas yang

dimiliki oleh wadah namun memaksimalkan total keuntungan yang diperoleh.[1]

Solusi permasalahan ini dapat dinyatakan sebagai vector *n*-tupel:

$$X = \{ x_1, x_2, \ldots, x_n \} \quad (1)$$

yang dalam hal ini, $x_i = 1$ jika benda ke-*i* dimasukkan ke dalam wadah, atau $x_i = 0$ jika benda ke-*i* tidak dimasukkan ke dalam wadah, karena itulah maka persoalan ini dinamakan *0/1 knapsack*. Sebagai contoh, X = {1, 0, 0, 1, 0} adalah sebuah solusi yang memasukkan benda ke-1 dan 4 ke dalam wadah, sedangkan benda ke-2, 3, dan 5 tidak dimasukkan ke dalam wadah.

Secara matematis, persoalan *0/1 knapsack* dapat dirumuskan sebagai berikut :

Maksimasi

$$F = \sum_{i=0}^{n} x_i p_i \quad (2)$$

dengan kendala (*constraint*):

$$\sum_{i=1}^{n} w_i x_i \leq K \quad (3)$$

2.2 Algoritma *branch and bound*

Algoritma branch and bound adalah sebuah metode pencarian di dalam ruang solusi secara sistematis. Ruang solusi pada algoritma branch and bound diorganisasikan ke dalam pohon ruang status. Pembentukan pohon ruang status pada algoritma branch and bound menggunakan skema dari algoritma BFS (*Breadth First Search*). Untuk mempercepat pencarian ke simpul solusi, maka setiap simpul diberi sebuah nilai ongkos (*cost*). Simpul berikutnya yang akan diekspansi tidak lagi berdasarkan urutan pembangkitannya (sebagaimana pada BFS murni), tetapi simpul yang memiliki ongkos paling kecil di antara simpul-simpul hidup lainnya (*least cost search*). Nilai ongkos pada setiap simpul *i* menyatakan taksiran ongkos termurah 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 dari status *i*. Ongkos ini dihitung dengan suatu nilai pembatas. Fungsi pembatas ini digunakan untuk membatasi pembangkitan simpul yang tidak mengarah ke simpul solusi.[1]

Untuk masalah knapsack ini, pengurutan cost dilihat dari total nilai keuntungan yang dimiliki dari setiap pengambilan langkahnya. Sedangkan nilai pembatas adalah apabila dia telah melebihi batas muat yang dimiliki oleh wadah tersebut. Pencarian yang dilakukan dirubah sedemikian rupa sehingga bukan mencari nilai terkecil namun mencari nilai terbesar namun tidak melebihi beban maksimum yang dimiliki oleh wadah.

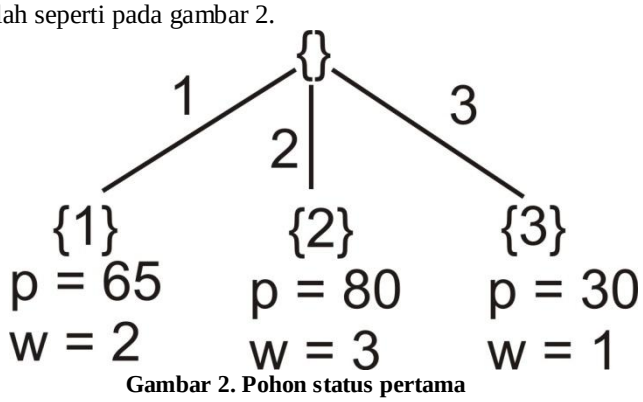
Aplikasi pada contoh soal, dimisalkan ada permasalahan sebagai berikut:

Tabel 1. Bobot dan keuntungan barang (n = 3)

Benda ke-*i*	w_i	p_i
1	2	65
2	3	80
3	1	30

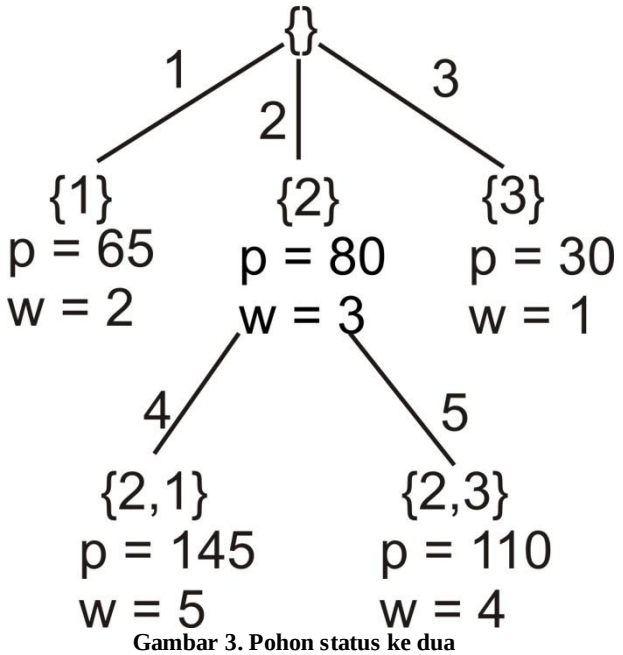
Pada contoh tersebut wadah memiliki batas beban senilai K =5. Pemilihan langkah yang ada didasarkan pada nilai keuntungan yang paling besar yang paling ada. Akar baru tidak dibangkitkan jika melebihi fungsi pembatas yang ada atau dalam hal ini melebihi kapasitas beban yang dimiliki oleh wadah. Penjelasan perlangkah akan dijelaskan di bawah ini.

Pada langkah pertama pohon solusi yang terbentuk adalah seperti pada gambar 2.



Gambar 2. Pohon status pertama

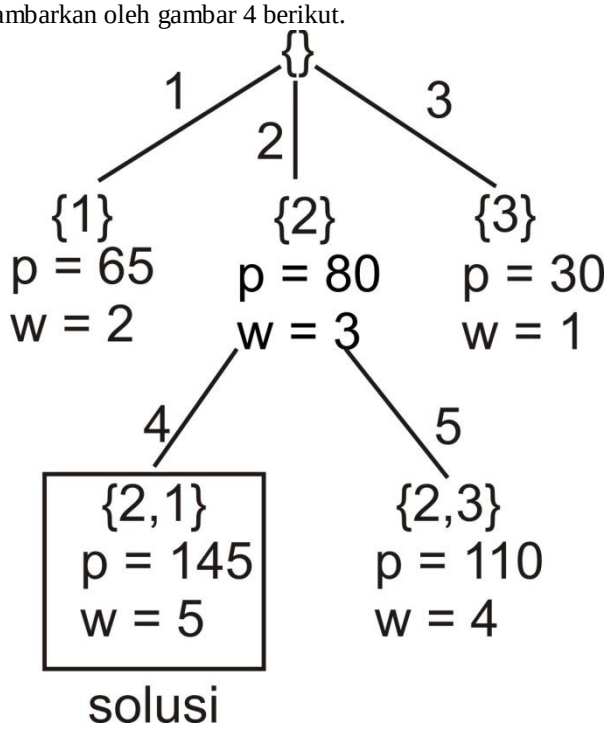
Pada langkah berikutnya pohon status dibangkitkan lagi dari nilai keuntungan yang paling besar. Dalam hal ini simpul yang diekspansi adalah simpul 2 yang beranggotakan benda 2 yang memiliki beban total 3 dengan nilai keuntungan 80. Seperti ditunjukkan pada gambar 3.



Gambar 3. Pohon status ke dua

Pada langkah berikutnya pohon status yang dibangkitkan dari nilai yang paling maksimum melebihi nilai batas sehingga pada simpul tersebut tidak dapat dieskpansi lagi lebih jauh. Pada simpul tersebut pula hasil paling optimum telah dicapai sehingga pencarian solusi selesai. Simpul 4 tersebut memiliki anggota benda 1 dan 2 yang memiliki bobot total sebesar 5 dan memiliki total keuntungan sebanyak 145.

Sehingga kondisi terakhir dimana solusi ditemukan digambarkan oleh gambar 4 berikut.



Gambar 4. Solusi pohon status dari contoh permasalahan

3. KESIMPULAN

Metode branch and bound dapat digunakan dalam pencarian solusi masalah integer knapsack. Dalam hal ini tentu saja ada beberapa hal yang sedikit dimodifikasi dari metode branch and bound yang murni dimana biasanya mencari nilai terkecil, namun pada permasalahan ini diganti menjadi mencari nilai terbesar yang ada. Sehingga dapat membentuk solusi dari masalah integer knapsack ini menjadi solusi yang optimal.

REFERENSI

[1] Munir, Rinaldi. 2007. *Diktat Kuliah Strategi Algoritmik.*

[2] Levitin, Anany. 2003 . *Introduction to the Design and Analysis of Algorithm*. Addison Wesley.

[3]Wikipedia,http://en.wikipedia.org/wiki/Knapsack_problem, waktu akses : 20 Mei 2008, pukul 10.00