# Pembuktian Kesulitan *Hamiltonian Cycle Problem* dengan Transformasi *3-Satisfiability Problem*

Arief Rahman 13511020
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
*13511020@std.stei.itb.ac.id*

***Abstrak*—Makalah ini akan membahas tentang pembuktian bahwa persoalan *Hamiltonian Cycle* merupakan persoalan NP-Complete. Pembuktian akan dilakukan dengan melakukan transformasi persoalan *3-Satisfiability* ke persoalan *Hamiltonian Cycle*. Pada makalah ini juga akan dibahas pembuktian bahwa persoalan *3-Satisfiability* merupakan persoalan NP-Complete, untuk memastikan bahwa pembuktian sifat NP-Complete dengan transformasi dari persoalan *3-Satisfiability* ke persoalan *Hamiltonian Cycle* valid.**

***Kata Kunci*—Graf, Kompleksitas, *NP-Completeness*, Satisfiability**

## I. Dasar Teori

### *A. Teori NP-Completeness*

Sebelum melakukan pembahasan lebih lanjut, akan diberikan definisi dasar terhadap istilah-istilah yang akan digunakan pada makalah ini terlebih dahulu.

**Definisi 1** (*persoalan keputusan*)
*Sebuah persoalan yang hanya memiliki jawaban IYA atau TIDAK.*

**Definisi 2** (*bahasa*)
*Himpunan dari seluruh input berupa string pada sebuah persoalan keputusan yang menghasilkan jawaban IYA.*

Input *string* didefinisikan sebagai sejumlah alfabet dari simbol. Misalnya, *string* biner terdiri dari alfabet {0, 1}. Contoh input *string* yang menggunakan *string* biner adalah 01010101101100011.

**Definisi 3** (*reducible polynomial*)
*Misalkan $L_1$ dan $L_2$ merupakan bahasa. $L_1$ disebut "polynomially reduced" terhadap $L_2$ (disimbolkan sebagai $L_1 \propto L_2$) jika ada sebuah algoritma dengan kompleksitas polinomial yang dapat mengubah semua instans input $i_1 \in L_1$* menjadi $i_2 \in L_2$.

*Reducability* bersifat asimetrik. Dengan kata lain, jika $L_1 \propto L_2$, tidak berarti $L_2 \propto L_1$. Walaupun begitu, *polynomial reducibility* memiliki karakteristik yang penting, yang akan diberikan pada teorema berikut:

**Teorema:** Jika $L_1 \propto L_2$ dan ada algoritma dengan kompleksitas polinomial untuk $L_2$, maka ada algoritma dengan kompleksitas polinomial untuk $L_1$. (Lihat [1], halaman 343 untuk pembuktian)

Dengan definisi-definisi di atas, dapat dibuat definisi untuk kelas-kelas kompleksitas persoalan.

**Definisi 4** (kelas P)
*P merupakan kelas bahasa (persoalan keputusan) L yang akan mengembalikan jawaban IYA dalam waktu polinomial dengan input x jika dan hanya jika $x \in L$.*

**Definisi 5** (kelas NP)
*NP merupakan kelas bahasa (persoalan keputusan) dengan proses pengecekan kebenaran input dapat dilakukan dalam waktu polinomial.*

Perhatikan bahwa pada definisi di atas tidak menyebutkan apapun tentang waktu yang diperlukan untuk mendapatkan jawaban dari persoalan; definisi di atas hanya menyatakan bahwa proses pengecekan kebenaran input hanya membutuhkan waktu polinomial.

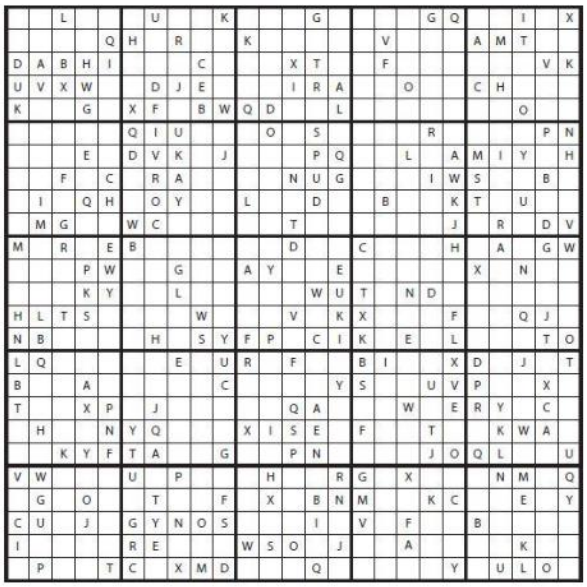
Dari definisi di atas juga dapat dilihat bahwa semua persoalan P juga merupakan persoalan NP, karena persoalan P dapat mengembalikan jawaban dalam waktu polinomial, sehingga pengecekan kebenaran jawaban juga hanya memerlukan waktu polinomial. Namun hingga saat ini belum dapat ditentukan apakah P = NP.

**Definisi 6** (kelas NP-Hard)
*Sebuah persoalan P merupakan NP-Hard jika semua persoalan lain di NP dapat direduksi ke P dalam waktu polinomial.*

**Definisi 7** (kelas NP-Complete)
*Sebuah persoalan P merupakan NP-Complete jika (1) $P \in NP$, dan (2) semua persoalan lain di NP dapat direduksi ke P dalam waktu polinomial.*

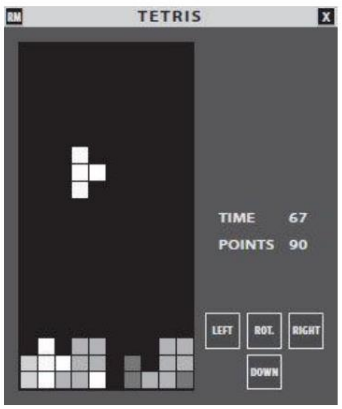
Persoalan NP-Complete dibatasi pada persoalan keputusan. Persoalan NP-Hard dapat berupa persoalan optimasi, yaitu persoalan yang meminta solusi optimal dari persoalan NP-Complete. Persoalan NP-Hard dan NP-Complete memiliki tingkat kesulitan yang sama. Saat ini, banyak persoalan yang ada telah terbukti merupakan permasalahan NP-Complete.

Hampir semua persoalan yang diketahui pada bidang *computer science* merupakan persoalan NP-Complete. Beberapa contoh persoalan yang diketahui merupakan persoalan NP-Complete adalah permainan Sudoku, Minesweeper, dan Tetris (Gambar 1, 2, dan 3)



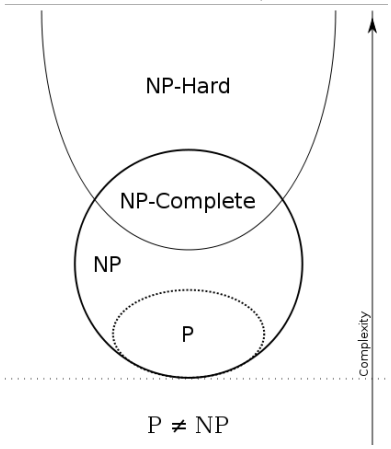
Gambar 1. Permainan Sudoku berukuran 25x25.


Gambar 2. Permainan Minesweeper berukuran 16x16.

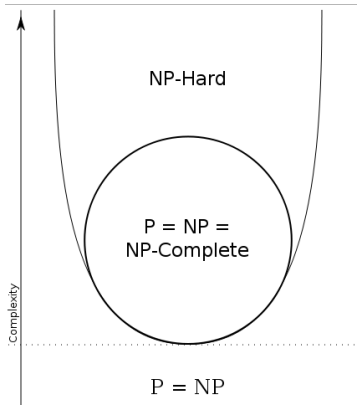



Gambar 3. Permainan Tetris

Gambar 1 dan 2 menjelaskan hubungan antara persoalan P, NP, NP-Complete, dan NP-Hard, dengan menggunakan asumsi P = NP dan P ≠ NP.




Gambar 1. Diagram hubungan antara persoalan P, NP, NP-Complete, dan NP-Hard dengan asumsi P ≠ NP benar.




Gambar 2. Diagram hubungan antara persoalan P, NP, NP-Complete, dan NP-Hard dengan asumsi P = NP benar.

## *B. Satisfiability Problem*

*Satisfiability Problem* (disingkat sebagai SAT) merupakan persoalan NP-Complete pertama yang ditemukan. Secara formal, persoalan SAT didefinisikan sebagai berikut.

*Persoalan* : *Satisfiability*
*Input* : Himpunan variabel V dan himpunan klausa C dengan menggunakan variabel yang ada di V.
*Output* : Apakah ada *truth assignment* yang memenuhi C; dengan kata lain, sebuah cara untuk mengeset variabel $v1, \ldots, vn$ bernilai *true* atau *false* sehingga setiap klausa di C memiliki setidaknya satu literal yang bernilai *true*?

Persoalan ini dapat dijelaskan dengan dua contoh. Misalkan ada himpunan klausa $C = \{\{v1, \bar{v}2\}, \{\bar{v}1, v2\}\}$ dengan himpunan variabel $V = \{v1, v2\}$. $\bar{v}i$ digunakan untuk menyatakan komplemen dari $vi$. Oleh karena itu, untuk memenuhi suatu himpunan klausa, diperlukan pembuatan *truth assignment* (dengan nilai *true* atau *false*) untuk setiap variabel di himpunan V untuk memenuhi semua klausa.

Contoh di atas dapat dipenuhi dengan mengeset $v1 = v2 = true$ atau $v1 = v2 = false$. Namun, misalkan himpunan klausa $C = \{\{v1, v2\}, \{v1, \bar{v}2\}, \{\bar{v}1\}$ . Untuk contoh ini, tidak ada *assignment* yang memenuhi, karena $v1$ harus bernilai *false* untuk memenuhi klausa ketiga, sehingga $v2$ harus bernilai *false* untuk memenuhi klausa kedua, sehingga klausa pertama tidak akan bisa terpenuhi.

Karena alasan teknis dan sosial, para mengakui bahwa SAT merupakan persoalan yang sulit; tidak memiliki algoritma penyelesaian polinomial untuk *worst-case*.

### *C. 3-Satisfiability Problem*

*3-Satisfiability Problem* (disingkat sebagai 3-SAT) merupakan instans khusus dari persoalan SAT. Berikut merupakan definisi formal dari persoalan 3-SAT.

*Persoalan* : *3-Satisfiability*
*Input* : Himpunan variabel V dan himpunan klausa C, dengan setiap klausa berisi tepat 3 literal, dengan menggunakan variabel yang ada di V.
*Output* : Apakah ada *truth assignment* yang memenuhi C; dengan kata lain, sebuah cara untuk mengeset variabel $v1, \ldots, vn$ bernilai *true* atau *false* sehingga setiap klausa di C memiliki setidaknya satu literal yang bernilai *true*?

Persoalan 3-SAT merupakan salah satu persoalan yang paling sering digunakan dalam pembuktian sifat NP-Complete dari persoalan lain, karena lingkup masalah yang dibahas di dalam persoalan ini jauh lebih sederhana (dengan kata lain sempit), sehingga proses reduksi lebih mudah, tidak seperti persoalan SAT yang lingkup masalahnya sangat beragam dan variatif.

### *D. Hamiltonian Cycle Problem*

*Hamiltonian Cycle Problem* (disingkat sebagai HCP untuk makalah ini) merupakan salah satu persoalan dalam teori graf. Sirkuit Hamilton sendiri merupakan suatu sirkuit pada graf G yang mengunjungi setiap simpul yang ada di G (kecuali simpul pertama dan simpul terakhir, yang merupakan simpul yang sama).

HCP dapat didefinisikan secara formal sebagai berikut.

*Persoalan* : *Hamiltonian Cycle*
*Input* : Graf G = (V,E) dengan V adalah himpunan simpul dan E adalah himpunan sisi.
*Output* : Apakah graf G mengandung sirkuit Hamilton; sirkuit yang mengunjungi setiap simpul tepat satu kali, kecuali simpul awal dan simpul akhir (yang merupakan simpul yang sama)?

Gambar 3 merupakan contoh sirkuit Hamilton yang valid.

Gambar 3. Contoh sirkuit Hamilton. Tur ditandai dengan garis hitam (*Sumber* : *Discrete Mathematics and Its Applications 4th Edition*)

## II. PEMBUKTIAN SIFAT NP-COMPLETE PADA 3-SAT

Sebelum membuktikan sifat NP-Complete dari HCP, akan dilakukan pembuktian bahwa persoalan 3-SAT merupakan persoalan NP-Complete terlebih dahulu. Persoalan 3-SAT merupakan persoalan NP-Complete jika memenuhi dua kondisi: (1) merupakan permasalahan NP, dan (2) dapat direduksi dari persoalan NP lain dalam waktu polinomial.

Kondisi pertama jelas benar, karena pengecekan kebenaran input (berupa *truth assignment* terhadap tiap variabel) dapat dilakukan dalam waktu polinomial dengan mengecek apakah setiap klausa mengandung setidaknya satu literal yang bernilai $true$.

Untuk memenuhi kondisi kedua, dapat digunakan suatu persoalan NP yang dapat direduksi menuju persoalan 3-SAT dalam waktu polinomial. Pada makalah ini, akan digunakan persoalan SAT sebagai persoalan yang akan direduksi.

Reduksi persoalan ini akan dilakukan dengan melakukan transformasi untuk setiap klausa secara independen berdasarkan panjangnya, dengan menambahkan klausa baru dan variabel baru untuk setiap pertambahan panjangnya. Misalkan sebuah klausa $Ci$ mengandung k literal:

- k = 1, dengan $Ci = \{z1\}$. Transformasi dilakukan dengan menambahkan dua variabel baru $v1, v2$ dan empat klausa (dengan 3 literal) baru: $\{v1, v2, z1\}, \{v1, \bar{v}2, z1\}, \{\bar{v}1, v2, z1\}, \{\bar{v}1, \bar{v}2, z1\}$. Satu-satunya cara untuk memenuhi keempat klausa ini sekaligus adalah dengan mengeset $z1 = true$. Hal ini akan membuat $Ci$ juga dapat terpenuhi.
- k = 2, dengan $Ci = \{z1, z2\}$. Transformasi dilakukan dengan menambahkan satu variabel baru $v1$ dan dua klausa (dengan 3 literal) baru: $\{v1, z1, z2\}, \{\bar{v}1, z1, z2\}$. Satu-satunya cara untuk memenuhi kedua klausa ini sekaligus adalah dengan mengeset $z1 = true$ atau $z2 = true$. Hal ini akan membuat $Ci$ juga dapat terpenuhi.
- k = 3, dengan $Ci = \{z1, z2, z3\}$. Pada kasus ini tidak dibutuhkan transformasi, karena $Ci$ sudah mewakili 3-SAT secara umum.
- k > 3, dengan $Ci = \{z1, z2, \ldots, zn\}$. Transformasi dilakukan dengan menambahkan $n - 3$ variabel n dan $n - 2$ klausa (dengan 3 literal) baru, dengan syarat-syarat:

$$2 \le j \le n-3$$
$$Ci,j = \{v_{i,j-1}, z_{j+1}, \bar{v}_{i,j}\}$$
$$C_{i,1} = \{z1, z2, \bar{v}_{i,1}\}$$
$$C_{i,n-2} = \{v_{i,n-3}, z_{n-1}, z_n\}$$

Untuk klausa yang besar, jika tidak ada literal awal yang ada di $Ci$ bernilai $true$, maka variabel-variabel yang baru tidak akan dapat memenuhi semua dari upaklausa yang baru. $C_{i,1}$ dapat dipenuhi dengan mengeset $v_{i,1} = false$, namun ini membuat $v_{i,2} = false$, dan seterusnya hingga akhirnya $C_{i,n-2}$ tidak dapat dipenuhi. Namun, jika ada satu literal $z1 = true$, maka akan ada n – 3 variabel bebas dan n – 3 klausa yang tersisa, sehingga setiap klausa dapat dipenuhi.

Proses reduksi ini memerlukan waktu sebesar $O(m + n)$ jika terdapat n klausa dan m literal dalam instans SAT. Karena suatu solusi SAT memenuhi instans 3-SAT dan suatu solusi 3-SAT juga menjelaskan bagaimanan cara mengeset variabel-variabel yang ada (jika diketahui sebuah solusi SAT-nya), maka persoalan yang telah direduksi ekivalen dengan persoalan awal.

Karena kedua kondisi telah terpenuhi, maka dapat disimpulkan bahwa persoalan 3-SAT merupakan persoalan NP-Complete.

## III. PEMBUKTIAN SIFAT NP-COMPLETE PADA HCP

Seperti pembuktian pada 3-SAT, untuk membuktikan bahwa HCP merupakan persoalan NP-Complete, harus dipenuhi dua kondisi: (1) merupakan permasalahan NP, dan (2) dapat direduksi dari persoalan NP lain dalam waktu polinomial.

Kondisi pertama jelas benar, karena pengecekan kebenaran input dapat dilakukan dalam waktu polinomial dengan mengecek apakah setiap simpul dikunjungi sekali (kecuali simpul awal yang dikunjungi dua kali) dan setiap sisi yang dilewati saat transisi simpul valid.

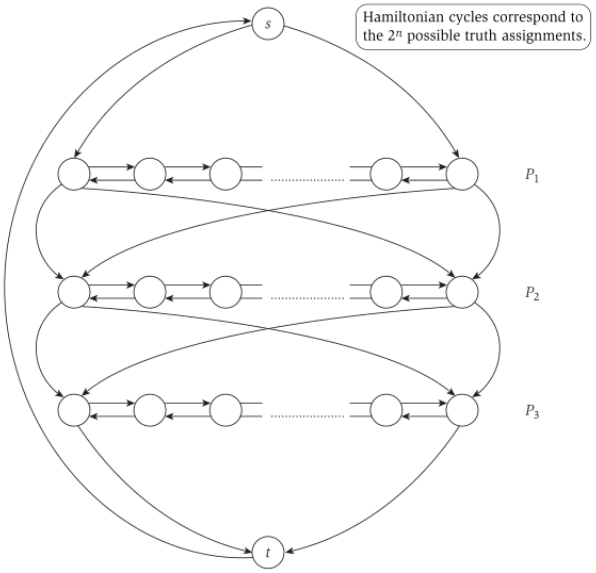
Untuk memenuhi kondisi kedua, dapat digunakan suatu persoalan NP yang dapat direduksi menuju persoalan HCP dalam waktu polinomial. Sesuai dengan judul makalah ini, persoalan yang akan digunakan untuk direduksi menuju HCP adalah persoalan 3-SAT. Persoalan 3-SAT lebih mudah untuk digunakan dibandingkan dengan persoalan SAT secara umum karena ruang lingkup masalahnya lebih kecil.

Misalkan ada sebuah instans 3-SAT, dengan variabel $x1, x2, \dots, xn$ dan klausa $C1, C2, \dots, Ck$ (dengan tiga literal).

Misalkan ada $2^n$ sirkuit Hamilton yang berbeda pada suatu graf. Hal ini mirip dengan fakta bahwa terdapat $2^n$ macam *truth assignment* yang dapat dilakukan terhadap variabel-variabel yang ada. Setiap batasan dari klausa yang ada akan dipetakan menjadi sebuah simpul di graf. Graf yang akan digunakan dalam reduksi ini merupakan graf berarah.

Untuk tahap awal, transformasi dilakukan dengan membentuk n jalur $P1, \dots, Pn$. $Pi$ terdiri dari simpul $v_{i1}, v_{i2}, \dots, v_{ib}$. b merupakan angka yang lebih besar dari jumlah klausa k. Misalkan $b = 3k + 3$. Ada sisi-sisi dari $v_{ij}$ ke $v_{i,j+1}$ dan sisi-sisi dari sisi berlainan (dari $v_{i,j+1}$ ke $v_{ij}$. Oleh karena itu, $Pi$ bisa ditelusuri dari "kanan ke kiri" maupun dari "kiri ke kanan".

Jalur-jalur ini kemudian dihubungkan dengan cara sebagai berikut. Untuk setiap $i = 1, 2, \dots, n-1$, didefinisikan sisi-sisi dari $v_{i1}$ ke $v_{i+1,1}$ dan ke $v_{i+1,b}$. Selain itu, juga dibuat dua simpul baru s dan t,dengan s memiliki sisi dari s ke $v_{i1}$ dan $v_{ib}$, dan t memiliki sisi dari $v_{n1}$ dan $v_{nb}$ ke t dan dari t ke s. Hasil reduksi sejauh ini dapat dilihat pada gambar 4.




Gambar 4. Proses reduksi pertama dari 3-SAT ke HCP.

Dari gambar ini dapat dilihat bahwa sirkuit Hamilton harus melewati sisi (t,s). Setelah memasuki s, sirkuit harus melewati P1 dari kiri ke kanan atau dari kanan ke kiri. Setelah itu P2 bisa dilewati dari kiri ke kanan atau dari kanan ke kiri, dan seterusnya hingga semua Pi berhasil dilewati dan memasuki t. Dengan kata lain, ada $2^n$ sirkuit Hamilton yang berbeda. Setiap sirkuit juga merepresentasikan n pilihan berbeda tentang bagaimana cara melewati $P_i$.

Secara umum ini memodelkan n cara untuk mengeset variabel $x1, \dots, xn$ pada instans persoalan 3-SAT. Oleh karena itu kita bisa mengidentifikasi setiap sirkuit Hamilton dengan unik dengan menggunakan *truth assignment* berikut: Jika sirkuit C melewati $P_i$ dari kiri ke kanan, maka $x_i$ diset menjadi $true$, selain itu, $x_i$ diset menjadi $false$.

Sekarang akan ditambahkan simpul baru untuk memodelkan semua klausa. Instans persoalan 3-SAT hanya akan terpenuhi jika dan hanya jika ada sebuah sirkuit Hamilton. Misalkan contoh klausa

$$C1 = x1\ OR\ \bar{x}2\ OR\ x3$$

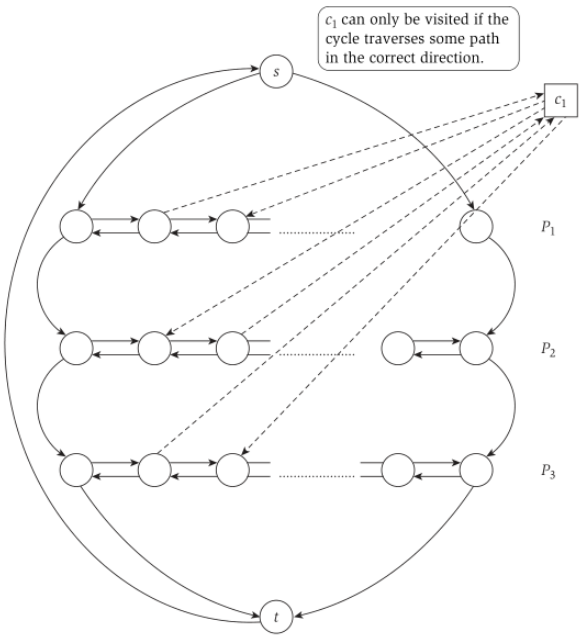
Dalam bahasa HCP, klausa ini menyatakan, "Sirkuit harus melewati $P_1$ dari kiri ke kanan, atau sirkuit harus melewati $P_2$ dari kanan ke kiri, atau sirkuit harus melewati $P_3$ dari kiri ke kanan." Oleh karena itu, akan ditambahkan simpul

$c_1$.

Untuk suatu nilai l, simpul $c_1$ akan memiliki sisi-sisi dari $v_{1l}, v_{2,l+1}$, dan $v_{3,l}$. Simpul $c_1$ juga akan memiliki sisi-sisi ke $v_{1,l+1}, v_{2,l}$, dan $v_{3,l+1}$. Oleh karena itu, $c_1$ bisa disambungkan dengan mudah ke sirkuit Hamilton apapun yang melewati P1 dari kiri ke kanan dengan mengunjungi $c_1$ antara $v_{1l}$ dan $v_{1,l+1}$. $c_1$ juga dapat disambungkan ke sirkuit Hamilton apapun yang melewati P2 dari kanan ke kiri, atau melewati P3 dari kiri ke kanan. Namun, c1 tidak dapat disambungkan ke sirkuit Hamilton yang dapat melakukan salah satu hal di atas.

Secara umum, didefinisikan sebuah simpul $c_j$ untuk setiap klausa $C_j$. Simpul pada 3j dan 3j + 1 akan di setiap jalur $P_i$ akan disimpan untuk variabel-variabel yang ada di klausa $C_j$. Misalkan ada klausa $C_j$ yang mengandung suatu literal t. Maka, jika $t = xi$, maka harus ditambahkan sisi ($v_{i,3j}$, $c_j$) dan ($c_j$, $v_{i,3j+1}$); jika $t = \bar{x}i$, maka harus ditambahkan sisi ($v_{i,3j+1}$, $c_j$) dan ($c_j$, $v_{i,3j}$).

Hasil akhir konstruksi graf berarah G dapat dilihat pada gambar 5.




Gambar 5. Graf hasil akhir reduksi dari 3-SAT ke HCP.

Sekarang, harus dibuktikan bahwa setiap instans dari persoalan 3-SAT dapat terpenuhi jika dan hanya jika graf G memiliki sebuah sirkuit Hamilton.

Pertama, misalkan ada sebuah *truth assignment* yang memenuhi instans persoalan 3-SAT. Maka, didefinisikan suatu sirkuit Hamilton berdasarkan penjelasan-penjelasan sebelumnya. Jika $x_i$ diset menjadi *true* pada *truth assignment* yang memenuhi, maka jalur $P_i$ akan ditelusuri dari kiri ke kanan; selain itu, jalur $P_i$ akan ditelusuri dari kanan ke kiri. Untuk setiap klausa $C_j$, karena klausa ini dipenuhi dengan *truth assignment* yang ada, maka dapat dipastikan akan ada paling sedikit satu jalur $P_i$ yang dapat pergi menuju simpul $c_j$, dan jalur yang sudah terbentuk dapat digabungkan dengan tur yang ada melalui sisi-sisi berdekatan di $v_{i,3j}$ dan $v_{i,3j+1}$.

Selanjutnya, misalkan ada sebuah sirkuit Hamilton C di graf berarah G. Jika C memasuki sebuah simpul $c_j$ dari sisi $v_{i,3j}$, maka tur harus keluar dari simpul tersebut melalui sisi $v_{i,3j+1}$. Jika tidak, maka $v_{i,3j+1}$ hanya akan memiliki satu simpul tetangga yang belum dikunjungi, yaitu $v_{i,3j+2}$. Hal ini membuat tur tersebut tidak dapat mengunjungi simpul ini ketika sifat sirkuit Hamilton masih terpenuhi.

Hal sebaliknya juga berlaku. Jika C memasuki sebuah simpul $c_j$ dari sisi $v_{i,3j+1}$, maka tur harus keluar dari simpul tersebut melalui sisi $v_{i,3j}$. Oleh karena itu, untuk setiap simpul $c_j$, simpul-simpul sebelum dan setelah $c_j$ di sirkuit c harus digabungkan dengan sebuah sisi e di G.

Maka, jika $c_j$ dikeluarkan dari sirkuit dan memasukkan sisi e untuk setiap j, maka akan didapat sebuah sirkuit Hamilton C' pada upagraf G – {c1, ..., ck}. Ini merupakan upagraf awal sebelum penambahan simpul. Seperti yang telah disebutkan sebelumnya, setiap sirkuit Hamilton pada upagraf ini harus melewati $P_i$ hanya satu arah (kiri ke kanan atau kanan ke kiri).

Oleh karena itu, C' merupakan sirkuit yang digunakan untuk mendefinisikan *truth assignment* di instans persoalan 3-SAT. Jika C' melewati $P_i$ dari kiri ke kanan, maka $x_i$ diset menjadi $true$, selain itu, $x_i$ diset menjadi $false$. Karena sirkuit C bisa mengunjungi setiap simpul klausa $c_j$, maka akan ada paling sedikit satu jalur yang terbentuk dengan arah benar relatif terhadap simpul $c_j$, sehingga *truth assignment* yang telah didefinisikan akan memenuhi semua klausa yang ada.

Karena dua kondisi di atas terpenuhi, dapat disimpulkan bahwa persoalan HCP merupakan persoalan NP-Complete.

## V. KESIMPULAN

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

## REFERENSI


[1] U. Manber. Introduction to Algorithms. Addison-Wesley, 1989.

[2] Skiena, Steven S.. The Algorithm Design Manual 2nd Edition. Springer.

[3] Kleinberg, Jon. Algorithm Design. Pearson – Addison-Wesley.

[4] Garey, Michael R.. Computers & Intractability : A Guide to the Theory of NP-Completeness. Bell Telephone Laboratories, Inc.. 1979.

[5] Greenwood, G.W.. Finding Solutions to NP-Problems : Philosophical Differences Between Quantum and Evolutionary Search Algorithms. Portland State University.

[6] Epp, Susanna S.. Discrete Mathematics with Applications 4th Edition. CENGAGE Learning. 2011.

[7] Fortnow, Lance. The Golden Ticket : P, NP, and the Search for the Impossible. Princeton University Press. 2013.

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, 29 April 2010

ttd

Arief Rahman, 13511020