

Penentuan Solusi Satisfiability (SAT) Problem Dengan Metode Kohonen Self-Organizing Map (KSOM)

by John Doe

Submission date: 03-Jun-2020 02:54AM (UTC+0700)

Submission ID: 1336062197

File name: Publikasi_Ilmiyah_16523073-Revisi_2.pdf (317.03K)

Word count: 1603

Character count: 10174

Penentuan Solusi Satisfiability (SAT) Problem Dengan Metode Kohonen Self-Organizing Map (KSOM)

Abstract— SAT Problem merupakan salah satu permasalahan NP-Complete yang sangat sulit untuk dipecahkan dengan cara konvensional. Sudah banyak aplikasi (SAT Solver) untuk menyelesaikannya. Tujuan dari penelitian ini untuk melakukan pemodelan SAT Solver menggunakan salah satu Jaringan Syarat Tiruan, Kohonen Self Organizing Map .

Kata Kunci : SAT Problem, Kohonen Self Organizing Map (K-SOM)

I. PENDAHULUAN

Setiap kehidupan dalam diri manusia di dunia ini, pasti memiliki berbagai macam ragam masalah. Namun, setiap permasalahan tersebut dapat diatasi. Beberapa dari kita dapat menangani masalah tersebut dengan cara sistematis. Salah metode yang bisa menyelesaikannya adalah dengan SAT Problem. Di dalam logika matematika, SAT merupakan konsep dasar *semantic* yang dapat menyelesaikan masalah dengan menentukan permasalahan kombinasi nilai variabel dalam sebuah klausa logika proposisi sehingga membuat klausa tersebut bernilai benar[1].

Oleh karena itu, untuk mempermudah proses penyelesaian dengan metode SAT *problem*, dibutuhkan SAT *solver*. SAT *solver* merupakan software yang bisa memecahkan SAT *problem*, yang berfungsi sebuah formula dari logika proposisi bernilai *satisfiable* atau *unsat*.

Dalam permasalahan yang kompleks, terkadang SAT solver mendapatkan kesulitan. Hal ini terjadi karena SAT solver merupakan perangkat lunak yang dibuat dengan algoritma basic dan dapat menyelesaikan masalah yang tergolong mudah/simple. Oleh karenanya, tingkat optimasi algoritma didalam membentuk SAT Solver merupakan hal yang sangat penting. Semakin tinggi peningkatan SAT solver, maka semakin baik kualitas solusi yang didapatkan, kemudian proses komputasi yang dibutuhkan memiliki waktu yang lebih sedikit dan *resource* yang dibutuhkan juga lebih rendah dari SAT Solver terdahulu.

Selama beberapa tahun terakhir tentang penelitian pembentukan algoritma SAT *solver* yang efisien untuk menyelesaikan SAT *problem* berkembang dengan sangat cepat. Perkembangan ini sudah sangat banyak berkontribusi khususnya pada kemajuan teknologi khususnya pada manusia yang secara otomatis dapat memecahkan masalah yang mengimplikasikan puluhan atau ratusan ribu variabel dan

jutaan klausa [1]. Salah satu contoh di dunia nyata yaitu dalam permasalahan *Electronic Design Automation* atau yang biasa disebut dengan EDA. Permasalahan yang ada pada EDA meliputi *formal equivalence checking, model, checking, formal verification of microprocessors*[3], dan lain sebagainya. Oleh karena itu, SAT Solver merupakan salah satu komponen penting untuk merancang hal tersebut.

Kohonen. Self organizing Map adalah satu JST yang *unsupervised learning* yang ditemukan Teuvo Kohonen. Tujuan dari metode ini adalah memvisualisasikan data dengan mengurangi dimensi data dalam bentuk low-dimensional data. Menurut Haykin [4], terdapat 3 komponen terpenting dalam metode SOM, yaitu *Competition, Cooperation, dan Syanptic Adaption*.

Sudah banyak SAT Solver yang dikembangkan dengan algoritma berbeda seperti WalkSAT [5], Satz [6], dan lain-lain. Namun, sebagian SAT Solver tersebut masih membutuhkan waktu komputasi yang memakan waktu cukup lama dalam proses eksekusinya yang dikarenakan solusi yang ditawarkan dalam bentuk eksak. Penelitian ini bertujuan untuk mengembangkan algoritma SAT dengan melakukan pemodelan dengan menggunakan metode K-Som untuk menyelesaikan SAT Problem

II. LANDASAN TEORI

A. Tinjauan Pustaka

Dasar teori dalam menyelesaikan pemodelan SAT dengan menggunakan metode KSOM ini adalah dengan menggunakan beberapa teori untuk dijadikan rujukan dalam melakukan penelitian ini. Diantaranya :

- a. Penelitian dari Carlos Ansotegui, Maria Luisa Bonet, Jordi Levy yang berjudul “SAT-based MaxSAT Algorithms” Penelitian ini dilakukan untuk melakukan percobaan mengatasi masalah industrial atau masalah yang sebenarnya menggunakan beberapa algoritma berbasis Max-SAT yang dirancang khusus untuk masalah tersebut [10]
- b. Penelitian yang berasal dari Alaa Ali Hameed, Bekir Karlik, Mohammad Shukri Salman, Gulden Eleyau yang berjudul “Robust adaptive learning approach to

self organizing maps". Tujuan penelitian ini untuk pendekatan pembelajaran yang adaptik dengan menggunakan metode *self organizing map*. [11]

B. Dasar Teori

a. Logika Proposisi & Formula CNF

Salah satu bentuk formula dalam format Logika proposisi yang dibuat dari kalimat proposisi, yaitu kalimat yang telah ditentukan dengan nilai kebenaran. Logika proposisi dinyatakan didalam sebuah variable proposisi hanya memiliki 2 kemungkinan nilai, *true* atau *false*

Sedangkan CNF adalah bentuk formula proposisi yang sudah memenuhi syarat formula dalam bentuk normal form. Syarat dari CNF adalah [7]:

1. Operatornya hanya \wedge , \vee , dan \neg .
2. Operator \neg diterapkan terhadap proposisi.

Contoh CNF :

- $l \wedge m$
- $l \wedge (m \vee \neg n)$
- $l \wedge m \wedge (\neg l \vee \neg m)$
- $(l \vee \neg m) \wedge m \wedge (\neg l \vee \neg n)$
- $(l \vee \neg m) \wedge (m \vee \neg n) \wedge (n \vee \neg l)$
- $(l \vee \neg m) \vee (n \vee \neg o)$
- $(l \vee \neg m) \vee (n \vee \neg o) \wedge (m \vee o)$

b. SAT Problem & SAT Solver

SAT Problem adalah proses menentukan formula tersebut bernilai *satisfiable*, atau *unsatisfiable*. Sebuah formula dikatakan *satisfiable* apabila terdapat kombinasi nilai kebenaran variable pembentuknya yang membuat formula tersebut bernilai benar (*true*), dan sebaliknya jika penyusun formulanya *false*, maka disebut *unsatisfiable*.

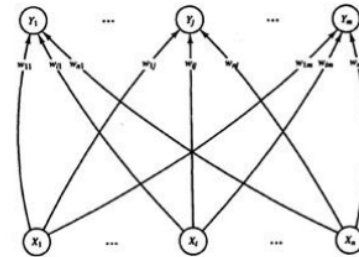
Misal :

- $l \wedge m$: *satisfiable, true* jika $l=T$ dan $m=T$.
- $l \wedge (m \vee \neg n)$: *satisfiable, true* jika $l=T$ dan $n=F$
- $l \wedge m \wedge (\neg l \vee \neg m)$: tidak *satisfiable*
- $(l \vee \neg m) \wedge m \wedge (\neg l \vee \neg n)$: *satisfiable* karena akan bernilai T jika salah satu adalah $m = T$, $l = F$, dan $n=F$

16

c. Kohonen Self Organizing Map (K-SOM)

K-SOM adalah JST yang bersifat *unsupervised* atau tanpa pembelajaran yang ditemukan oleh peneliti asal Finlandia yang bernama Teuvo Kohonen. Fungsi dari KSOM ini adalah untuk memvisualisasi data dengan mengurangi dimensi data. Setiap bobot vector dari K-SOM menjadi contoh dari input pola yang terkait dengan cluster tersebut^{[1]b}. KSOM memiliki lapisan layer input dan lapisan layer output, Setiap neuron pada lapisan input terkoneksi satu sama lain neuron pada lapisan output.



Gambar 1. Arsitektur K-SOM (Fausett,1993)

d. Python

Merupakan Bahasa interpretatif yang mendukung multi paradigma pemrograman namun tidak dibatasi pada pemrograman berorientasi objek, pemrograman imperatif dan pemrograman fungsional.^[10] Bahasa pemrograman python memiliki bahasa yang lebih dinamis dan memiliki tatabahasa yang mudah untuk dipelajari [12].

e. Microsoft Visual Studio Code

Merupakan salah satu IDE terkini saat ini yang memiliki fitur yang sangat dan banyak dan lengkap, salah satunya adalah *syntax coloring* dan *bracket matching*. Banyak Bahasa pemrograman yang mendukung fitur tersebut termasuk Python, Visual Studio Code juga memiliki fitur IntelliSense yang membantu menulis kode program melalui Popup yang muncul secara otomatis saat mengetik dan menampilkan saran sintaks [13].

III. HASIL DAN PEMBAHASAN

Dalam melakukan pengembangan makalah tentu membutuhkan metodologi penelitian agar penelitian yang sedang dilakukan berjalan dengan semestinya. Metodologi penelitian digunakan untuk Menyusun makalah terdiri dari identifikasi masalah, analisis kebutuhan serta pemodelan.

A. Identifikasi Masalah

Melakukan Identifikasi masalah , kebutuhan , cara kerja serta ruang lingkup aplikasi dengan cara :

- Melakukan analisis terhadap permasalahan SAT problem.
- Melakukan analisis terhadap Kohonen Self Organizing Map dan menyelesaikan SAT Problem.
- Membaca Paper dari penelitian sebelumnya.

B. Analisis Kebutuhan

Melakukan analisis apa yang diperlukan untuk membuat aplikasi berbasis Python yang dapat menyelesaikan permasalahan SAT Problem dengan menggunakan metode K-SOM

C. Pemodelan

Pada tahap ini dilakukan pemodelan KSOM terhadap SAT Problem. Berikut adalah contoh pemodelan SAT Problem, terdapat SAT Problem dengan Formula CNF :

$$(p \vee q) \wedge (r \vee \neg s) \wedge (r) \wedge (s)$$

1. Inisiasi Vector Input

Langkah yang pertama dilakukan adalah menentukan jumlah vector input Wij, semisal nilai yang ditentukan yaitu 4 vector input.

- Vector Input X1 (p,q,r,s) = 1,1,0,0
- Vector Input X2 (p,q,r,s) = 0,0,0,1
- Vector Input X3 (p,q,r,s) = 1,0,0,0
- Vector Input X4 (p,q,r,s) = 0,0,1,1

2. Inisialisasi Neuron Output

SAT Problem memiliki dua kemungkinan yaitu bernilai variable 0 yang mempresentasikan pemberian false (unsatisfiable) atau bernilai variable 1 yang mempresentasikan pemberian true (satisfiable).

3. Inisialisasi bobot serta learning rate

Menginisialisasi weight (bobot) neuron output dengan nilai antara x-min dan x-max secara acak. Kemudian menentukan learning rate (α) sebesar 0.6

Y0	Y1
$\begin{bmatrix} 0.2 \\ 0.6 \\ 0.5 \\ 0.9 \end{bmatrix}$	$\begin{bmatrix} 0.8 \\ 0.4 \\ 0.7 \\ 0.3 \end{bmatrix}$

Gambar 2. Bobot Neuron Output

4. Untuk tiap vector input, melakukan Langkah 5-7

5. Untuk setiap j, melakukan perhitungan dibawah ini :

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

kemudian temukan index J yang memiliki nilai D(J) terkecil

Vector Input X1 (p.,q.,r.,s.) = 1,1,0,0

$$D(1) = (0.2 - 1)^2 + (0.6 - 1)^2 + (0.5 - 0)^2 + (0.9 - 0)^2 = 1.86$$

$$D(2) = (0.8 - 1)^2 + (0.4 - 1)^2 + (0.7 - 0)^2 + (0.3 - 0)^2 = 0.98$$

Vector Input X2 (p.,q.,r.,s.) = 0,0,0,1

$$D(1) = (0.2 - 0)^2 + (0.6 - 0)^2 + (0.5 - 0)^2 + (0.9 - 1)^2 = 0.66$$

$$D(2) = (0.92 - 0)^2 + (0.76 - 0)^2 + (0.28 - 0)^2 + (0.12 - 1)^2 = 2.28$$

Vector Input X3 (p.,q.,r.,s.) = 1,0,0,0

$$D(1) = (0.08 - 1)^2 + (0.24 - 0)^2 + (0.2 - 0)^2 + (0.96 - 0)^2 = 1.87$$

$$D(2) = (0.92 - 1)^2 + (0.76 - 0)^2 + (0.28 - 0)^2 + (0.12 - 0)^2 = 0.68$$

Vector Input X4 (p.,q.,r.,s.) = 0,0,1,1

$$D(1) = (0.08 - 0)^2 + (0.24 - 0)^2 + (0.2 - 1)^2 + (0.96 - 1)^2 = 0.71$$

$$D(2) = (0.97 - 0)^2 + (0.31 - 0)^2 + (0.12 - 1)^2 + (0.05 - 1)^2 = 2.73$$

6. Menemukan Index J yang memiliki nilai D(J) terkecil.

Vector Input X1 (p,q,r,s) = 1,1,0,0

-D(j) minimum untuk j = 1

Vector Input X2 (p.,q.,r.,s.) = 0,0,0,1

-D(j) minimum untuk j = 0

Vector Input X3 (p.,q.,r.,s.) = 1,0,0,0

-D(j) minimum untuk j = 1

Vector Input X4 (p.,q.,r.,s.) = 0,0,1,1

-D(j) minimum untuk j = 0

7. Memperbaharui semua bobot tiap unit j dengan rumus :

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \alpha[x_i - w_{ij}(\text{old})]$$

Vector Input X1. (p.,q.,r.,s.) = 1,1,0,0

$$\begin{aligned} W21 &= 0.8 + 0.6(1 - 0.8) = 0.92 \\ W22 &= 0.4 + 0.6(1 - 0.4) = 0.76 \\ W23 &= 0.7 + 0.6(0 - 0.7) = 0.28 \\ W24 &= 0.3 + 0.6(0 - 0.3) = 0.12 \end{aligned}$$

Vector Input X2 (p.,q.,r.,s.) = 0,0,0,1

$$\begin{aligned} W11 &= 0.2 + 0.6(0 - 0.2) = 0.08 \\ W12 &= 0.6 + 0.6(0 - 0.4) = 0.24 \\ W13 &= 0.5 + 0.6(0 - 0.5) = 0.2 \\ W14 &= 0.9 + 0.6(1 - 0.9) = 0.96 \end{aligned}$$

Vector Input X3 (p.,q.,r.,s.) = 1,0,0,0

$$\begin{aligned} W21 &= 0.92 + 0.6(1 - 0.92) = 0.97 \\ W22 &= 0.76 + 0.6(0 - 0.76) = 0.31 \\ W23 &= 0.28 + 0.6(0 - 0.28) = 0.12 \\ W24 &= 0.12 + 0.6(0 - 0.12) = 0.05 \end{aligned}$$

Vector Input X4 (p.,q.,r.,s.) = 0,0,1,1

$$\begin{aligned} W11 &= 0.08 + 0.6(0 - 0.08) = 0.04 \\ W12 &= 0.24 + 0.6(0 - 0.24) = 0.09 \\ W13 &= 0.20 + 0.6(1 - 0.20) = 0.68 \\ W14 &= 0.96 + 0.6(1 - 0.96) = 0.98 \end{aligned}$$

8. Memperbarui *learning rate* dengan mengalikan *learning rate* yang lama dengan 0.5

$$\begin{aligned} \alpha(0) &= 0.6 \\ \alpha(t+1) &= 0.5 \alpha(t) \\ \alpha(\text{baru}) &= 0.5(0.6) = 0.3 \end{aligned}$$

9. Mengupdate bobot sampai bobot tersebut bersifat konvergen.

Tabel 1. Hasil perhitungan

X1	X2	X3	X4	KLUSTER
1	1	0	0	1
0	0	0	1	0
1	0	0	0	1
0	0	1	1	0

Dari hasil perhitungan diatas menunjukkan bahwa vector input X₁ dan X₃ masuk kedalam kluster 1 dan vector input X₂ dan X₄ masuk kedalam kluster 0

D. Implementasi Pemodelan

Pada tahap ini, dilakukan implementasi pemodelan sebelumnya Pemodelan SAT dengan menggunakan metode *K-Self Organizing Map* dalam menyelesaikan SAT *problem* dengan menggunakan bahasa pemograman Python 3.7.6

1. Pembuatan Vector Input

```
# Menentukan Vector Input
training_patterns_input = [[1, 1, 0, 0],
                           [0, 0, 0, 1],
                           [1, 0, 0, 0],
                           [0, 0, 1, 1]]
```

2. Penentuan Bobot (weight)

```
# Menentukan Weight (Bobot)
self.w = [[0.2, 0.6, 0.5, 0.9],
          [0.8, 0.4, 0.7, 0.3]]
```

3. Neuron Output

```
# Neuron Ouput
MAX_CLUSTERS = 2
```

4. Perhitungan $D(j) = \sum_i (w_{ij} - x_i)^2$ pada setiap J

```
# Hitung setiap j
def compute_input(self, vectorNumber, trainingTests):
    self.mD[0] = 0.0
    self.mD[1] = 0.0

    for i in range(self.maxClusters):
        for j in range(self.mVectors):
            self.mD[i] += math.pow((self.w[i][j] - trainingTests[vectorNumber][j]), 2)
```

5. Memperbarui Update bobot dari setiap unit pemenang

```
# Mengupdate bobot dari unit pemenang
for j in range(self.mVectors):
    self.w[dMin][j] = self.w[dMin][j] + (self.mAlpha * (patterns[1][j] - self.w[dMin][j]))
```

6. Mengurangi Learning rate

```
# Reduce the learning rate.
self.mAlpha = self.decayRate * self.mAlpha
```

7. Hasil Perhitungan

```
Vector Input ( 1, 1, 0, 0, ) Masuk Kedalam Kategori 1
Vector Input ( 0, 0, 0, 1, ) Masuk Kedalam Kategori 0
Vector Input ( 1, 0, 0, 0, ) Masuk Kedalam Kategori 1
Vector Input ( 0, 0, 1, 1, ) Masuk Kedalam Kategori 0

Bobot Untuk Node 0 koneksi :
0.000, 0.000, 0.508, 1.000,

Bobot Untuk Node 1 koneksi :
1.000, 0.492, 0.000, 0.000,
```

IV. KESIMPULAN

Telah berhasil dibentuk suatu pemodelan SAT dengan metode dari Jaringan Syaraf Tiruan yaitu Kohonen Self-Organizing Map. Pada pembahasan kali ini, hanya mampu memodelkan SAT dengan kasus yang masih sangat sederhana. Untuk cara kerja SAT ini belum bisa dibandingkan dengan SAT Solver, karena masih dalam tahap pemodelan. oleh karena itu masih perlu pengembangan dari SAT Solver. Untuk kedepannya, perlu dilakukan pengembangan lebih lanjut terhadap SAT Solver dengan Metode K-SOM.

V. REFERENSI

- [1] D. P. Pratama, "Tugas Akhir," *PENYELESAIAN BOOLEAN SATISFIABILITY PROBLEM DENGAN ALGORITMA DAVIS PUTNAM LOGEMANN LOVELAND (DPLL) MENGGUNAKAN JAVA*, p. 1, 2018
- [2] S. O. Haykin, *Neural Networks: A Comprehensive Foundation*, England: Pearson Education., 1999.
- [3] L. V. Fauset, *Fundamentals of Neural Networks Architectures Algorithms and Applications*, Prentice Hall, 1st edition, 1993.
- [4] T. Hidayat, *Logika Proposisi*, Yogyakarta: Dar Fiqrin, 2014.
- [5] T. Hidayat and A. B. Irhasni, "SAT Solver dengan DPLL dalam pemograman Deklaratif," 2017.
- [6] M. Huth and M. Ryan, *Logic in Computer Science: Modelling and Reasoning about Systems*, New York: Cambridge University Press, 2004.
- [7] M. S. Joao P and A. S. Karem, "GRASP: A New Search," *Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design*, pp. 220-227.
- [8] P.Utama, "https://prasetautamacv.wordpress.com/2016/11/20/self-organizing-maps-kohonen-networks/," [Online].
- [9] *Discovering Knowledge in Data (Introduction to Data Mining)*, Chapter 8, Daniel T. Larose, Wiley, 2004
- [10] C. Ansotegui, M. L. Bonet and J. Levy, "SAT-based MaxSAT algorithms," *Artificial Intelligence*, vol.196, pp. 77-105, 2013.
- [11] A. A. Hameed , B. Karlik, M. S. Salman and G. Elevant, "Robust adaptive learning approach to self-organizing maps," *Knowledge-Based Systems*, vol. 171, pp 25-36, 2019.
- [12] Wikipedia, "Python (bahasa pemrograman)".[Online]. Available: [https://id.wikipedia.org/wiki/Python_\(bahasa_pemrograman\)](https://id.wikipedia.org/wiki/Python_(bahasa_pemrograman))
- [13] Bagus Aji Santoso, "Visual Studio Code, Editor Baru dari Microsoft Untuk Windows, OS X, dan Linux". 05 Mei 2015.[Online].Available. <https://www.codepolitan.com/visual-studio-code-editor-baru-dari-microsoft-untuk-windows-os-x-dan-linux>

Penentuan Solusi Satisfiability (SAT) Problem Dengan Metode Kohonen Self-Organizing Map (KSOM)

ORIGINALITY REPORT

18%

SIMILARITY INDEX

9%

INTERNET SOURCES

6%

PUBLICATIONS

14%

STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Islam Indonesia Student Paper	4%
2	www.codepolitan.com Internet Source	2%
3	Submitted to Universitas Dian Nuswantoro Student Paper	2%
4	Submitted to Universiti Teknologi Malaysia Student Paper	1%
5	Antonio Morgado, Federico Heras, Joao Marques-Silva. "Model-Guided Approaches for MaxSAT Solving", 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, 2013 Publication	1%
6	Submitted to University of Essex Student Paper	1%
7	Martin Babka, Tomáš Balyo, Ondřej Čepek, Štefan Gurský, Petr Kučera, Václav Vlček.	1%

"Complexity issues related to propagation completeness", Artificial Intelligence, 2013

Publication

8	socs.binus.ac.id Internet Source	1%
9	tel.archives-ouvertes.fr Internet Source	1%
10	link.springer.com Internet Source	1%
11	Ansótegui, Carlos, Joel Gabàs, Yuri Malitsky, and Meinolf Sellmann. "MaxSAT by improved instance-specific algorithm configuration", Artificial Intelligence, 2016. Publication	1%
12	Submitted to Universitas Brawijaya Student Paper	1%
13	Submitted to Universitas Pendidikan Indonesia Student Paper	<1%
14	Michael Behrman, Roland Linder, Amir H. Assadi, Brett R. Stacey, Misha-Miroslav Backonja. "Classification of patients with pain based on neuropathic pain symptoms: Comparison of an artificial neural network against an established scoring system", European Journal of Pain, 2007 Publication	<1%

15

apriadiwika.blogspot.com

Internet Source

<1%

16

Submitted to Universitas 17 Agustus 1945

Surabaya

Student Paper

<1%

Exclude quotes Off

Exclude matches Off

Exclude bibliography On

Penentuan Solusi Satisfiability (SAT) Problem Dengan Metode Kohonen Self-Organizing Map (KSOM)

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5
