Pemecah Sudoku Interaktif Berbasis Python

Proposal Tugas Akhir

Kelas TA 1

Muhammad Zakaria Musa
NIM: 1103130047

Program Studi Sarjana Teknik Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2017

Lembar Persetujuan

Pemecah Sudoku Interaktif Berbasis Python

A Python-Based Interactive Sudoku Solver.

Muhammad Zakaria Musa
NIM: 1103130047

Proposal ini diajukan sebagai usulan pembuatan tugas akhir pada
Program Studi Sarjana Teknik Informatika
Fakultas Informatika Universitas Telkom

Bandung, 1 November 2017
Menyetujui

Calon Pembimbing 1	Calon Pembimbing 2
Yanti Rusmawati, Ph.D.	Muhammad Arzaki, M.Kom.
NIP: 15711785-1	NIP: 15871701-2

Abstrak

Abstrak . . . .

Kata Kunci: metode formal, SAT solver, Sudoku, pengujian kotak hitam . . . .

Daftar Isi

Bab I

Pendahuluan

1.1 Latar Belakang

Sudoku berasal dari kata *Sūji wa dokushin ni kagiru* yang berarti angkanya harus tunggal[1] adalah suatu *puzzle* (teka-teki) yang direpresentasikan oleh sebuah matriks (*array* dua dimensi) berukuran $n^2 \times n^2$ yang dibangun dari n^2 dengan submatriks (atau blok) yang berukuran $n \times n$. Sudoku merupakan *puzzle* yang biasanya dimainkan oleh satu orang. Pada awal permainan, terdapat beberapa sel yang telah terisi yang disebut dengan pemberian (*givens*). Untuk menyelesaikan permainan ini, seorang pemain harus mengisi setiap sel yang belum terisi dengan angka di antara 1 sampai n^2 sedemikan sehingga setiap baris, setiap kolom, dan setiap blok (submatriks berukuran $n \times n$) memuat tepat satu bilangan di antara 1 sampai n^2. Biasanya suatu sudoku didesain agar tepat memiliki satu kemungkinan solusi. Hal ini juga mengakibatkan sudoku dapat diselesaikan hanya dengan mengandalkan penalaran yang sederhana. Pengisian suatu sel dapat dilakukan dengan meninjau kemungkinan dari isi sebuah sel.

Banyak pemain yang terjebak pada teka-teki sudoku dan tidak dapat melanjutkan permainan. Hal ini terjadi karena pemain mengisi nilai yang salah atau sudoku tidak memiliki solusi. Oleh karena itu banyak pemain yang membutuhkan pemecah sudoku yang dapat memberikan keterpenuhan dari sebuah sudoku. Sehingga pemain dapat mengetahui sebuah sudoku memiliki solusi atau tidak.

Hingga saat ini, sudah banyak penelitian yang membahas penyelesaian sudoku secara matematis maupun komputasional. Salah satu metode yang cukup dikenal adalah penyelesaian sudoku dengan memanfaatkan masalah keterpenuhan formula proposisional (*propositional satisfiability problem*). Dengan pendekataan ini, syarat-syarat yang harus dipenuhi oleh suatu sudoku dimodelkan dengan satu atau lebih formula logika proposisi[2, 3]. Keterpenuhan (*satisfiability*) dari himpunan formula yang memodelkan syarat-syarat ini akan menjamin bahwa suatu sudoku memiliki suatu solusi.

Pada tugas akhir ini, penulis akan membuat pemecah sudoku interaktif berbasis python. Bahasa python dipilih karena implementasi SAT solver dalam

bahasa python masih masih sedikit dibandingkan pada bahasa C dan C++[4]. Aplikasi akan menggunakan pengujian kotak hitam untuk menilai kualitas aplikasi[5].

1.2 Perumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah pada tugas akhir ini adalah bagaimana cara membuat SAT solver bebasis python untuk pemecah sudoku yang interaktif.

1.3 Batasan Masalah

Batasan pada tugas akhir ini terbatas pada:

1. Penulis hanya membangun aplikasi sudoku solver berukuran (4×4), (9×9), (16×16).
2. Program mengeluarkan keterpenuhan dari sudoku.
3. Program mengeluarkan solusi dari sudoku jika pengguna mengiginkan.
4. Aplikasi akan diuji dengan metode pengujian kotak hitam (*black box testing*).

1.4 Tujuan

Tujuan yang ingin dicapai pada tugas akhir ini adalah untuk membuat pemecah sudoku yang interaktif.

1.5 Rencana Kegiatan

Pada pengerjaan tugas akhir ini beberapa hal yang akan saya lakukan adalah sebagai berikut:

1. Analisa spesifikasi dan Kebutuhan sistem.
2. Translasi spesifikasi sistem.
3. Pembuatan aplikasi.
4. Pengujian aplikasi.
5. Analisis sistem.
6. Penulisan laporan.

1.6 Jadwal Kegiatan

Jadwal pengerjaan tugas akhir sesuai dengan alur yang telah dibuat.

Tabel 1.1: Jadwal Kegiatan.

No	Jenis Kegiatan	Bulan						
		November 2017	Desember 2017	Januari 2018	Februari 2018	Maret 2018	April 2018	Mei 2018
1	Analisa spesifikasi dan kebutuhan sistem							
2	Translasi spesifikasi sistem							
3	Pembuatan aplikasi							
4	Pengujian aplikasi							
5	Analisis sistem							
6	Penulisan Laporan							

Bab II

Kajian Pustaka

Pada bab ini penulis akan menjelaskan teori yang digunakan selama pengerjaan tugas akhir.

2.1 Sejarah Sudoku

Sebelum sudoku modern berkembang. *Puzzle latin square* terlebih dahulu berkembang yang di repretasikan oleh sebuah matriks berukuran $n \times n$ yang memiliki angka $1hinggan$ yang harus diisi oleh pemain. untuk menyelesakain *puzzle* setiap baris dan kolom pada *latin square* harus memiliki nilai angka yang berbeda. *Latin square* pertama kali diciptakan oleh Euler pada tahun 1783[6]. Pada 19 November 1892 *Le Siècle* sebuah surat kabar di perancis mempublikasikan *magic square* yaitu sebuah teka-teki yang di repretasikan oleh sebuah matriks berukuran 9×9 yang memiliki dua buah submatriks berukuran $3hingga3$. Tujuan dari *magic square* adalah pemain mengisi baris dan kolom sehingga nilai penjumlahan setiap baris dan kolom itu sama. Sudoku modern sendiri lahir dari Howard Garns yang dipublikasikan di *Dell Magazines* pada 1979[7].

2.2 Sudoku

Sudoku berasal dari kata *Sūji wa dokushin ni kagiru* yang berarti angkanya harus tunggal[1]. Sudoku dipopulerkan di jepang oleh *Nikoli in the paper Monthly Nikolist* pada April 1984. Sudoku adalah suatu *puzzle* (teka-teki) yang direpresentasikan oleh sebuah matriks (*array* dua dimensi) berukuran $n^2 \times n^2$ yang dibangun dari n^2 dengan submatriks (atau blok) yang berukuran $n \times n$. Sudoku merupakan *puzzle* yang biasanya dimainkan oleh satu orang. Pada awal permainan, terdapat beberapa sel yang telah terisi yang disebut dengan pemberian (*givens*). Untuk menyelesaikan permainan ini, seorang pemain harus mengisi setiap sel yang belum terisi dengan angka di antara 1 sampai n^2 sedemikan sehingga setiap baris, setiap kolom, dan setiap blok (submatriks berukuran $n \times n$) memuat tepat satu bilangan di antara 1 sampai n^2. Biasanya suatu sudoku didesain agar tepat memiliki satu kemungkinan solusi. Hal ini juga mengakibatkan sudoku dapat diselesaikan hanya dengan mengandalkan penalaran yang sederhana. Pengisian suatu sel dapat dilakukan dengan

meninjau kemungkinan dari isi sebuah sel.

2.3 Metode Formal

Metode formal adalah teknik yang digunakan untuk memodelkan suatu sistem yang kompleks [8]. Metode formal dapat digunakan untuk mengembangkan sebuah aplikasi baik itu perangkat lunak atau perangkat keras. Pada masa spesifikasi, metode formal dapat digunakan untuk memberikan gambaran dari sistem yang akan dikembangkan sedetail yang diinginkan. Metode formal dapat digunakan untuk memandu kegiatan pengembangan selanjutnya dan dapat digunakan untuk memverifikasi bahwa pra-syarat untuk sistem yang dikembangkan telah lengkap dan dapat dilanjutkan ke tingkat selanjutnya.

2.4 Pengujian Kotak Hitam

Pengujian kotak hitam adalah teknik pengujian yang tidak memerlukan pengetahuan kode dari aplikasi[5]. Penguji biasanya melakukan pengujian kotak hitam melalui antarmuka pengguna dari aplikasi dengan diberikan *test case* masukan dan menganalisa keluaran dari aplikasi.

2.5 Logika Proposisi

Logika proposisi adalah bahasa formal untuk memodelkan situasi yang kita sehingga kita dapat memberi alasan tentang hal itu secara formal[8]. Logika proposisi terdiri dari sebuah nilai kebenaran dari proposisinya.

Operator yang digunakan pada logika proposisi adalah sebagai berikut

- Konjungsi(dan) pada operator ini kedua proposisi yang dihubungkan harus bernilai benar agar formula bernilai *true* . Operator ini dilambangkan dengan $\wedge$.
- Disjungsi(atau) pada operator ini salah satu proposisi yang dihubungkan harus bernilai benar agar formula bernilai *true*. Operator ini dilambangkan dengan$\vee$
- Negasi(tidak) pada operator ini suatu proposisi akan memiliki nilai sebaliknya dari sebuah proposisi. Operator ini dilambangkan dengan $\neg$
- Implikasi(jika-maka) pada operator ini digunakan untuk merepresantisakan kata "jika p maka q"sebuah formula akan bernilai *false* jika nilai p benar dan nilai q salah. Operator ini dilambangkan dengan $\rightarrow$.
- Biimplikasi(jika dan hanya jika)pada operator ini suatu formula hanya akan bernilai *true* jika kedua proposisinya memiliki nilai kebenaran yang sama. Operator ini dilambangkan dengan $\leftrightarrow$

2.6 *Satisfiability Problem*(SAT)

Masalah SAT (SAT *problem*) adalah salah satu masalah penting dalam logika komputasional[8]. Dalam pengerjaannya SAT berfokus pada membuktikan sebuah klausa *satisfiable* sehingga SAT berfokus pada menemukan sebuah model yang membuat klausa tersebut *satisfiable*. Jika tidak ditemukan sebuah model yang membuat klausa tersebut *satisfiable* maka klausa tersebut *unsatisfiable*. Masalah SAT merupakan masalah NP-complete, hingga saat ini tidak terdapat algoritma yang efisien untuk memecahkan masalah tersebut.

2.7 *Waterfall Design*

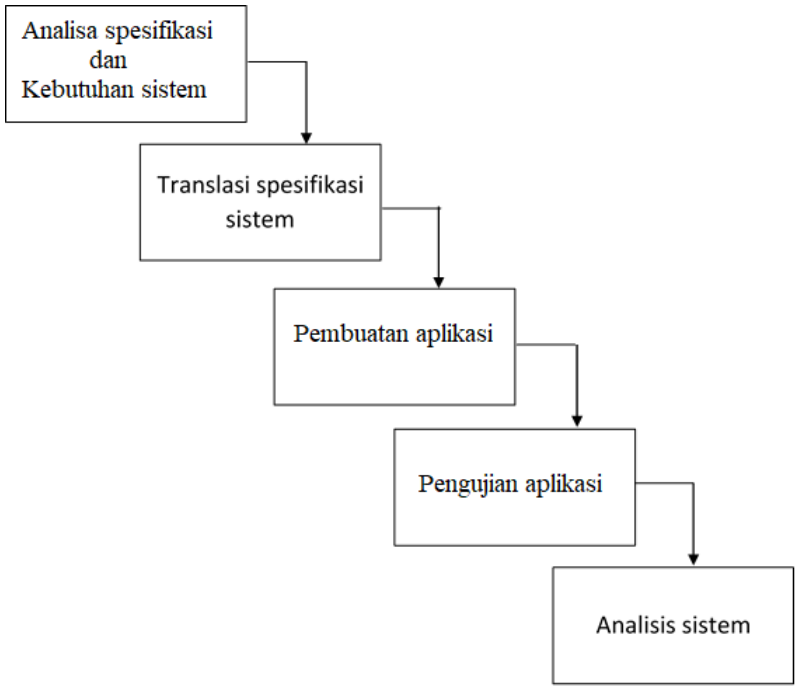
Waterfall atau air terjun adalah model yang diperkenalkan oleh Dr. Winston W. Royce pada tahun 1970. Model ini menuntun pengembang untuk melakukan pengembangan perangkat lunak yang akan dibangun secara sistematis dan mempunyai keterkaitan antara prosesnya. dalam arti tahapan proses berikutnya tidak akan dapat di wujudkan jika tahapan sebelumnya tidak terselasaikan.

Ada dua langkah penting yang harus ada dalam pembuatan perangkat lunak. pertama ialah analisis lalu di ikuti oleh *coding*, jika di bagi akan menjadi beberapa tahap yang spesifik yaitu *System Requirements* $\rightarrow$ *Software Requirements* $\rightarrow$ *Analysis* $\rightarrow$ *Program Design* $\rightarrow$ *Coding* $\rightarrow$ *Testing* [9].

Bab III

Metodologi

Pada bab ini penulis menjelaskan mengenai metodologi yang digunakan selama pengerjaan tugas akhir.



Gambar 3.1: Metodologi.

3.1 Analisa spesifikasi dan Kebutuhan sistem

Menganalisa spesifikasi-spesifikasi pada teka-teki sudoku, lalu membuat kebutuhan dari aplikasi sistem yang akan mengeluarkan *use case* aplikasi.

3.2 Translasi spesifikasi sistem

Spesifikasi yang sudah di dapat diubah dalam bentuk logika proposisi lalu dijadikan dalam bentuk CNF.

3.3 Pembuatan aplikasi

Spesifikasi yang sudah dalam bentuk CNF akan digunakan pada *SAT Solver*. Lalu *use case* yang didapat akan dibuat fungsionalitasnya. Pembuatan aplikasi menggunakan *waterfall design* tahapan-nya berupa:

3.3.1 *Analysis*

Dalam *analysis*, penulis menjabarkan mulai dari hal mendasar yaitu kendala global dalam pengembangan perangkat lunak yang sudah ada. Adapun masalahnya adalah minimnya aplikasi pemecah sudoku yang interaktif.

3.3.2 *Design*

Pada tahap ini penulis akan menjabarkan algoritma yang penulis buat dalam bentuk deskriptif dan *use case* dari program yang akan di implementasikan.

3.3.3 *Coding*

Lalu dalam tahap *coding* penulis akan menggunakan bahasa pemograman python dengan SAT *solver*[10] sebagai dasarnya. Setelah itu akan ditambahkan GUI dan *use case* yang telah di buat.

3.4 Pengujian aplikasi

Pada tahap ini aplikasi akan diujikan kebeberapa penguji dengan diberikan lembat *test case* yang meliputi beberapa aspek kualitas pada aplikasi. Teknik yang digunakan pada pengujian adalah pengujian kotak hitam[5].

3.5 Analisis sistem

Menganalisa kelayakan kualitas dari aplikasi berdasarkan hasil dari pengujian.

Daftar Pustaka

[1] "Ed pegg jr.'s math games: Sudoku variations," http://www.mathpuzzle.com/MAA/41-Sudoku

[2] G. Kwon and H. Jain, "Optimized CNF encoding for sudoku puzzles," in *Proc. 13th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR2006)*, 2006, pp. 1–5.

[3] I. Lynce and J. Ouaknine, "Sudoku as a SAT Problem." in *ISAIM*, 2006.

[4] "Understanding SAT by Implementing a Simple SAT Solver in Python," http://sahandsaba.com/understanding-sat-by-implementing-a-simple-sat-solver-in-python.html.

[5] B. Beizer, *Black-box testing: techniques for functional testing of software and systems.* John Wiley & Sons, Inc., 1995.

[6] S. Jones, P. Roach, and S. Perkins, "Properties of sudoku puzzles," in *Proceedings of the 9th international conference on Software Engineering.* IEEE Computer Society Press, 11 2007, pp. 7–11.

[7] "The answer men," http://content.time.com/time/magazine/article/0,9171,2137423,00.htr

[8] M. Huth and M. Ryan, *Logic in Computer Science: Modelling and reasoning about systems.* Cambridge University Press, 2004.

[9] W. W. Royce, "Managing the development of large software systems: concepts and techniques," in *Proceedings of the 9th international conference on Software Engineering.* IEEE Computer Society Press, 1987, pp. 328–338.

[10] "A simple SAT solver implemented in Python (with some non-SAT abilities)." https://github.com/stephenroller/satsolver.

Lampiran

Pada bab ini akan berisi lampiran dari pengerjaan tugas akhir, contohnya skrip NuSMV.