



Universidade Federal  
do Rio de Janeiro  

---

Escola Politécnica

## SATYRUS III: COMPILADOR PARA COMPUTADOR QUÂNTICO

Pedro Maciel Xavier

Projeto de Graduação apresentado ao Curso de Engenharia de Computação e Informação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientadores: Priscila Machado Vieira Lima  
Felipe Maia Galvão França

Rio de Janeiro  
Dezembro de 2021

# SATYRUS III: COMPILADOR PARA COMPUTADOR QUÂNTICO

Pedro Maciel Xavier

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE COMPUTAÇÃO E INFORMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE COMPUTAÇÃO.

Examinado por:

---

Prof. Nome do Primeiro Examinador Sobrenome, D.Sc.

---

Prof. Nome do Segundo Examinador Sobrenome, Ph.D.

---

Prof. Nome do Terceiro Examinador Sobrenome, D.Sc.

---

Prof. Nome do Quarto Examinador Sobrenome, Ph.D.

---

Prof. Nome do Quinto Examinador Sobrenome, Ph.D.

RIO DE JANEIRO, RJ – BRASIL

DEZEMBRO DE 2021

Maciel Xavier, Pedro

Satyrus III: Compilador para Computador Quântico/Pedro Maciel Xavier. – Rio de Janeiro: UFRJ/ Escola Politécnica, 2021.

X, 10 p.: il.; 29, 7cm.

Orientadores: Priscila Machado Vieira Lima

Felipe Maia Galvão França

Projeto de Graduação – UFRJ/ Escola Politécnica/ Curso de Engenharia de Computação e Informação, 2021.

Referências Bibliográficas: p. 9 – 9.

1. Compiladores. 2. Otimização. 3. Computação Quântica. 4. Lógica. I. Machado Vieira Lima, Priscila *et al.* II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia de Computação e Informação. III. Título.

*A alguém cujo valor é digno  
desta dedicatória.*

# Agradecimentos

Gostaria de agradecer a todos.

Resumo do Projeto de Graduação apresentado à Escola Politécnica/ UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Computação.

## SATYRUS III: COMPILADOR PARA COMPUTADOR QUÂNTICO

Pedro Maciel Xavier

Dezembro/2021

Orientadores: Priscila Machado Vieira Lima

Felipe Maia Galvão França

Curso: Engenharia de Computação e Informação

Apresenta-se, nesta tese, ...

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Engineer.

## SATYRUS III: COMPILER FOR QUANTUM COMPUTER

Pedro Maciel Xavier

December/2021

Advisors: Priscila Machado Vieira Lima

Felipe Maia Galvão França

Course: Computer Engineering

In this work, we present ...

# Sumário

<b>Lista de Figuras</b>	<b>ix</b>
<b>Lista de Tabelas</b>	<b>x</b>
<b>1 Introdução</b>	<b>1</b>
<b>2 Revisão Bibliográfica</b>	<b>2</b>
2.1 Compiladores . . . . .	2
2.2 Lógica Proposicional . . . . .	2
2.3 Otimização Pseudo-Booleana . . . . .	2
<b>3 Metodologia</b>	<b>4</b>
3.1 Mapeamento . . . . .	4
3.2 Tipos de Restrições . . . . .	4
3.2.1 Restrições de Otimalidade . . . . .	4
3.2.2 Restrições de Integridade . . . . .	4
3.3 Cálculo das Penalidades . . . . .	4
3.4 SATish . . . . .	6
3.5 WTA . . . . .	6
<b>4 Resultados e Discussões</b>	<b>7</b>
<b>5 Conclusões</b>	<b>8</b>
<b>Referências Bibliográficas</b>	<b>9</b>
<b>A Algumas Demonstrações</b>	<b>10</b>



# Lista de Figuras

2.1	A estrutura básica de um compilador. . . . .	2
2.2	A estrutura básica de um compilador. . . . .	2

# Lista de Tabelas

# Capítulo 1

## Introdução

# Capítulo 2

## Revisão Bibliográfica

### 2.1 Compiladores

O que é um compilador?

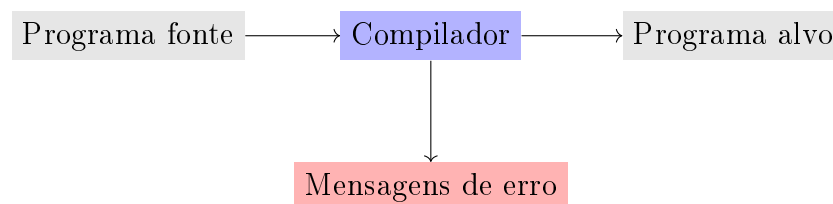


Figura 2.1: A estrutura básica de um compilador.

E o compilador do Satyrus?

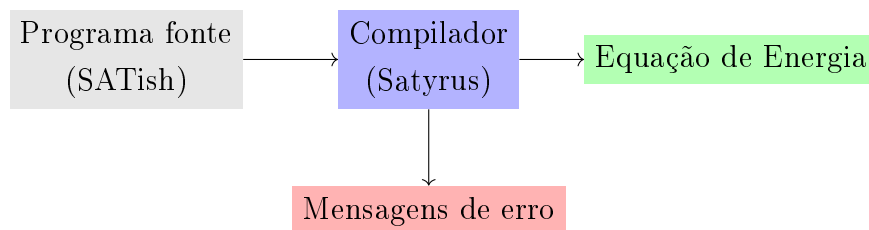


Figura 2.2: A estrutura básica de um compilador.

### 2.2 Lógica Proposicional

### 2.3 Otimização Pseudo-Booleana

Toda função pseudo-booleana pode ser representada de maneira única por um polinômio multi-linear da forma[1]

$$\varphi(x_1, \dots, x_n) = \sum_{\omega \in \Omega} c_\omega \prod_{i \in \omega} x_i$$

# Capítulo 3

## Metodologia

### 3.1 Mapeamento

### 3.2 Tipos de Restrições

O produto da compilação realizada pelo Satyrus é uma equação de energia da forma

$$E(\mathbf{x}) = E_0(\mathbf{x}) + \sum_i \lambda_i E_i(\mathbf{x}) \quad (3.1)$$

Onde  $E_0(\mathbf{x})$  representa o custo imposto pelas restrições de otimalidade e  $E_i(\mathbf{x})$  aquele oriundo das restrições de integridade. Os pesos  $\lambda_i$  dizem respeito às penalidades aplicadas sobre cada nível  $i$  da hierarquia de restrições. Trataremos com cuidado das suas especificidades em breve.

A geração de  $E(\mathbf{x})$  supõe resolver um problema de programação inteira onde as entradas de  $\mathbf{x}$  assumem valores binários e deseja-se minimizar o valor de  $E(\mathbf{x})$ .

$$\begin{aligned} \min E(\mathbf{x}) \\ \text{t.q. } \mathbf{x} \in \mathbb{B}^n \end{aligned} \quad (3.2)$$

#### 3.2.1 Restrições de Otimalidade

#### 3.2.2 Restrições de Integridade

### 3.3 Cálculo das Penalidades

Cada penalidade  $\lambda_i$  é construída de maneira que a violação de qualquer cláusula pertencente ao  $i$ -ésimo nível induza à energia total um valor maior do que aquele gerado pela violação de todas as cláusulas presentes nos níveis inferiores. De maneira um pouco mais abstrata, é preciso que a penalidade  $\lambda_i$ , somada ao custo da

configuração de menor valor dos níveis anteriores, seja maior do que a mais alta energia que estes possam proporcionar.

Tendo isso em mente, definimos

$$E(\mathbf{x})|_i = E_0(\mathbf{x}) + \sum_{j < i} \lambda_j E_j(\mathbf{x})$$

como sendo a equação de energia correspondente aos níveis de penalidade inferiores a  $i$ . Seu custo é idêntico àquele obtido por  $E(\mathbf{x})$  em estados que não violam restrições do nível  $i$  em diante.

Assim, dizemos ser necessário que

$$\lambda_i > \max E(\mathbf{x})|_i - \min E(\mathbf{x})|_i \quad (3.3)$$

Aqui lembramos que cada  $E_j : \mathbb{B}^n \rightarrow \mathbb{N}$  com  $j > 0$  conta o número  $n_j$  de cláusulas violadas no nível  $j$ . Portanto, para as funções relativas às restrições de integridade, temos  $\max E_j(\mathbf{x}) = n_j$  e  $\min E_j(\mathbf{x}) = 0$ . Isso nos permite reescrever (3.3) como

$$\lambda_i > \mu_0 + \sum_{j < i} \lambda_j n_j \quad (3.4)$$

onde  $\mu_0 = \max E_0(\mathbf{x}) - \min E_0(\mathbf{x})$ .

Escolhendo  $\varepsilon > 0$  arbitrariamente pequeno, tomamos (3.4) por

$$\lambda_i = \mu_0 + \sum_{j < i} \lambda_j n_j + \varepsilon \quad (3.5)$$

Tomando a diferença entre as penalidades de níveis consecutivos é possível obter uma expressão para calcular cada  $\lambda_i$  eficientemente através de um processo iterativo. A partir de (3.5) fazemos

$$\lambda_{i+1} - \lambda_i = \left( \mu_0 + \sum_{j \leq i} \lambda_j n_j + \varepsilon \right) - \left( \mu_0 + \sum_{j < i} \lambda_j n_j + \varepsilon \right) = \lambda_i n_i$$

ou seja,

$$\begin{cases} \lambda_{i+1} = \lambda_i (n_i + 1) & (\text{passo indutivo}) \\ \lambda_1 = \mu_0 + \varepsilon & (\text{caso base}) \end{cases} \quad (3.6)$$

Ainda nos resta calcular  $\mu_0$ . Conhecer máximo e mínimo de  $E_0(\mathbf{x})$ , a princípio, é simplesmente o caso irrestrito do próprio problema que desejamos resolver. Sabendo, contudo, que  $E_i(\mathbf{x})$  é da forma

$$E_i(\mathbf{x}) = \sum_{\omega \in \Omega_i} c_\omega \prod_{k \in \omega} x_k \quad (3.7)$$

podemos obter cotas inferiores e superiores através de seus coeficientes  $c_\omega$ . Para a cota inferior, somaremos todos os valores negativos, uma vez que é sempre possível escolher  $\mathbf{x} = 0$  para obter tal limite. Para a cota superior, consideramos os valores estritamente positivos. Em ambos os casos, não poderíamos esquecer de adicionar eventuais termos constantes, isto é,  $c_\omega$  quando  $\omega = \emptyset$ . Como tomaremos imediatamente a diferença entre as cotas, isso não será relevante. Concluindo,

$$\begin{aligned}\mu_0 &= \sum_{\omega \in \Omega_0} \frac{c_\omega + |c_\omega|}{2} - \sum_{\omega \in \Omega_0} \frac{c_\omega - |c_\omega|}{2} \\ &= \sum_{\omega \in \Omega_0} |c_\omega|\end{aligned}\tag{3.8}$$

### 3.4 SATish

### 3.5 WTA



# Capítulo 4

## Resultados e Discussões

Os resultados...

# Capítulo 5

## Conclusões

As conclusões...

# Referências Bibliográficas

- [1] BOROS, E., HAMMER, P. L., 2002, “Pseudo-Boolean optimization”, *Discrete Applied Mathematics*, v. 123, n. 1, pp. 155–225. ISSN: 0166-218X. doi: [https://doi.org/10.1016/S0166-218X\(01\)00341-9](https://doi.org/10.1016/S0166-218X(01)00341-9). Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0166218X01003419>>.

# Apêndice A

## Algumas Demonstrações