

UNIVERSIDADE ESTADUAL DE MARINGÁ  
CENTRO DE TECNOLOGIA  
DEPARTAMENTO DE INFORMÁTICA  
TCC - TRABALHO DE CONCLUSÃO DE CURSO  
PROPOSTA DE PROJETO DE TCC 2022

**Guia de ajustes de parâmetros para algoritmos  
heurísticos.**

**Área de Concentração:** Otimização algorítmica

**Aluno:** José Rafael Silva Hermoso

**Orientador:** Prof. Dr. Ademir A. Constantino

Maringá, 27 de novembro de 2022.

UNIVERSIDADE ESTADUAL DE MARINGÁ  
CENTRO DE TECNOLOGIA  
DEPARTAMENTO DE INFORMÁTICA  
TCC - TRABALHO DE CONCLUSÃO DE CURSO  
PROPOSTA DE PROJETO DE TCC 2022

**Guia de ajustes de parâmetros para algoritmos  
heurísticos.**

**Área de Concentração:** Otimização algorítmica

**Aluno:** José Rafael Silva Hermoso

**Membros da Banca:**

**Orientador:** Prof. Dr. Ademir A. Constantino

**Convidado 1:** Marco Aurélio Lopes Barbosa

**Convidado 2:** Daniel Kikuti

Maringá, 27 de novembro de 2022.

# 1 Introdução

Quando buscamos resolver um problema, o primeiro passo a ser feito é planejar uma forma de resolver, ou seja, determinar qual método de resolução será aplicado. Na área de algoritmos determinar uma técnica de solução para a formulação do algoritmo é um passo muito importante para que a solução seja computada da melhor forma. Dentro dessas técnicas temos a força bruta, a divisão e conquista, programação dinâmica e a programação linear mista e inteira que percorrem todas as instâncias do problema, para chegar na solução ótima. Essas técnicas geram uma solução exata, ao custo de processamento, o que as tornam lentas quando aplicadas para problemas com grandes entradas. Então, para esse tipo de problema, pode ser utilizado algoritmos heurísticos, que se destacam pela eficiência, pois com uma quantidade baixa de recursos, principalmente no quesito tempo de execução, encontram soluções boas (SUCUPIRA, 2004). Essa solução encontrada pelo algoritmo heurístico não possui garantia de ser a solução ótima para o problema.

Os algoritmos heurísticos, são utilizados em várias áreas do conhecimento, com o intuito de otimizar uma função objetivo, aparecendo em problemas clássicos como o problema do caixeiro viajante e o problema da cobertura de conjuntos (CONSTANTINO et al., 2003), e nas mais diversas áreas do conhecimento, como na parte de eletrônica com a busca do ponto de máxima potência (MOÇAMBIQUE, 2012), química com o problema de estimar propriedades radioativas (SOUTO et al., 2005) e preservação ambiental como o problema de estimar fontes de poluentes (LUZ et al., 2007).

Ao utilizarmos algoritmos heurísticos, nos deparamos com uma grande quantidade de parâmetros, como por exemplo o algoritmo genético, onde temos como parâmetros ajustáveis o número de gerações, número de indivíduos na população, taxa de mutação e critério de parada (NERY, 2017), e cabe ao usuário escolher esses parâmetros. Essa escolha de parâmetros, muitas vezes é feita de maneira empírica, fazendo ajustes manuais e analisando o desempenho, o que não garante que a combinação de parâmetros escolhida é a melhor possível, influenciando no tempo de processamento e resultado final do algoritmo.

Tendo isso em mente, propõe-se neste trabalho um estudo para a criação de um guia de ajustes de parâmetros para algoritmos heurísticos, para dar suporte a futuros estudos que venham a utilizar algoritmos heurísticos. Será utilizado inicialmente o problema do caixeiro viajante e a heurística de algoritmo genético para o estudo e análise de seus parâmetros.

## 2 Fundamentação Teórica

Neste capítulo são apresentadas as definições e os conceitos gerais que serão utilizados para o desenvolvimento do trabalho. Inicialmente, define-se o que são algoritmos heurísticos, onde são utilizados, como funcionam e o que são seus parâmetros. Posteriormente define-se a ferramenta de automatização de teste de parâmetros que será utilizada neste trabalho.

### 2.1 Problemas que podem ser resolvidos com heurísticas

Os problemas onde as heurísticas se destacam são os da classe NP, que não possuem métodos eficientes de solução conhecidos (SUCUPIRA, 2004), podendo ser de decisão, busca ou otimização, que estão presentes em várias áreas da ciência, como grafos, redes, conjuntos, otimização, química e escalonamento (CORMEN et al., 2009).

### 2.2 Algoritmos Heurísticos

#### 2.2.1 Definição

Ao contrário de métodos exatos, que buscam encontrar uma solução ótima por meio da combinação de todas as soluções possíveis, as heurísticas funcionam como um procedimento algorítmico baseado em um modelo cognitivo, através de regras que são definidas baseadas na experiência do desenvolvedor. Os métodos heurísticos englobam estratégias aproximativas que tem como objetivo encontrar uma boa solução com um custo computacional razoável, levando em conta que não existe garantia de que a solução encontrada é a solução ótima (CORDENONSI, 2008).

#### 2.2.2 Aplicação

A aplicação de heurísticas é bem ampla, como já foi citado anteriormente, aparecem em diversas áreas do conhecimento e buscam gerar soluções para os problemas da classe NP. “O fato de um problema de programação discreta ser classificado como NP-Completo é aceito como forte evidência da não existência de algoritmos polinomiais para sua resolução e consequentemente como uma justificativa para a utilização de algoritmos enumerativos ou para o desenvolvimento de heurísticas”(CAMPELLO, 1994, p. 57).

### 2.2.3 Tipos de heurísticas

No contexto de algoritmos heurísticos, temos as heurísticas construtivas, que tem como foco construir uma solução do zero, como o vizinho mais próximo (COVER; HART, 1967), que busca construir uma solução verificando sua vizinhança e escolhendo o melhor elemento baseado em uma função de avaliação. Essa vizinhança é o espaço que contém os elementos que compõem a solução, no caso do PCV a vizinhança é composta pelas outras cidades que possuem ligações com a cidade que está sendo analisada, já para o problema da mochila essa vizinhança é o espaço que contém os elementos que serão colocados na mochila. Também existem as heurísticas melhorativas que partindo de uma solução já construída, realiza passos sucessivos que alteram a posição dos elementos da solução inicial com o intuito de gerar uma nova solução com qualidade superior à solução inicial (CORDENONSI, 2008). Alguns exemplos de algoritmos que realizam a alteração das posições de uma solução são o 2-opt e 3-op. Por fim, temos as meta-heurísticas, que são algoritmos heurísticos que servem de guia para outras heurísticas. Geralmente são algoritmos de busca local, que tem como objetivo explorar o espaço de soluções com o intuito de encontrar novas soluções (CORDENONSI, 2008), sendo uma opção para contornar o problema de encontrar uma solução ótima local, que aparece em problemas de otimização combinatória.

### 2.2.4 Parâmetros

As meta-heurísticas possuem um grande número de parâmetros que devem ser ajustados de forma correta para que o algoritmo produza um bom resultado (LEÓN-ALDACO; CALLEJA; ALQUICIRA, 2015). Esses parâmetros são diferentes para cada tipo de algoritmo, variando de acordo com a proposta do algoritmo, por exemplo, o algoritmo genético utiliza tamanho da população e taxa de mutação, já o algoritmo de colônia de formigas, utiliza número de formigas, taxa de evaporação e pesos do feromônio.

## 2.3 Ferramenta para configuração automática de parâmetros de algoritmos *Irace*

O pacote *Irace* implementa o método Iterated Race, para a configuração automática de parâmetros de algoritmos de otimização. O ajuste de seus parâmetros é feito encontrando as configurações mais adequadas a um conjunto de instâncias de uma otimização utilizando o teste estatístico de Friedman. Seu funcionamento se baseia em começar com uma configuração de parâmetros já estabelecida pelo usuário e a cada iteração da ferramenta selecionar a melhor configuração, aquela que gera um melhor resultado, e atualizar os parâmetros com essa configuração, para então repetir esse processo com as novas configurações (LÓPEZ-IBÁÑEZ et al., 2016b).

## 3 Motivação

Os algoritmos heurísticos e meta-heurísticos são opções fortes e muito utilizadas para aproximar soluções em problemas NP, mas a calibragem de seus parâmetros é bem complexa e importante, devido ao fato de que esses algoritmos possuem diversos parâmetros, afetando diretamente na solução e no tempo de execução do algoritmo. Tendo isso em vista, o trabalho buscará entender e desenvolver um guia para auxiliar futuros estudos, que venham a utilizar algoritmos heurísticos, à calibrar os parâmetros de seus algoritmos da melhor forma, visando obter a melhor solução possível.

## 4 Objetivos

Este trabalho tem como proposta montar um guia para auxiliar o ajustes de parâmetros para os algoritmos heurísticos, buscando entender o comportamento de cada parâmetro e como ele interfere na solução gerada pelo algoritmo. Visando chegar em um conjunto de parâmetros que proporcionem as melhores soluções.

### 4.1 Objetivos Específicos

Os objetivos específicos deste trabalho são:

1. Investigar métodos de calibragem de parâmetro;
2. Aplicar esses métodos com algoritmos heurísticos a serem selecionados;
3. Analisar como os parâmetros influenciam na solução final;
4. Desenvolver um guia para auxiliar na calibragem de parâmetros de algoritmos heurísticos.

## 5 Materiais e Métodos

Este trabalho tem como proposta a criação de um guia para o ajuste de parâmetros de algoritmos heurísticos.

Este guia será desenvolvido baseado em testes de configurações de parâmetros para algoritmos heurísticos, que possuam parâmetros ajustáveis, através da ferramenta *Irace*. Os testes serão feitos a partir de parâmetros, que inicialmente serão escolhidos de forma arbitrária e serão melhorados a partir de testes com a ferramenta. Também será feita uma análise dos parâmetros que tiverem um bom resultado nos testes com o *Irace*, comparando com outros parâmetros próximos e analisando o comportamento do tempo de execução e do valor da solução ótima.

As referências bibliográficas para este trabalho, tem como base a pesquisa de publicações periódicas, artigos e livros que abordam algoritmos heurísticos, ferramentas de análise e ajustes de parâmetros para algoritmos heurísticos e problemas de otimização. Para a o teste de parâmetros e visualização do resultado será utilizado a ferramenta *Irace*, uma plataforma para testes automáticos de parâmetros, que facilita a alteração e manutenção dos parâmetros para os algoritmos ([LÓPEZ-IBÁÑEZ et al., 2016b](#)).

Os algoritmo heurístico de testes para o ajuste de parâmetros serão implementados em C++, uma linguagem de programação rápida e compilada que possui suporte pela ferramenta de testes de parâmetros *Irace* que será utilizada. O desenvolvimento e a execução dos testes será realizado em uma máquina com configurações de memória de 8GB de RAM e processador Intel Core i7-7500U, com sistema operacional Ubuntu 64-bit (20.04), uma distribuição Linux.

## 6 Cronograma

Seguem abaixo as atividades que serão desenvolvidas para este trabalho:

1. Levantamento bibliográfico para base teórica;
2. Estudo de parâmetros para os algoritmos heurísticos selecionados para o trabalho;
3. Implementação dos algoritmos selecionados;

Tabela 1 – Cronograma.

Atividades	2022				2023				
	Set	Out	Nov	Dez	Jan	Fev	Mar	Abr	Mai
1	x	x							
2		x	x						
3			x	x					
4			x						
5					x	x			
6					x	x			
7						x			
8						x			
9							x		
10							x	x	
11							x	x	
12								x	x

Fonte: Elaborado pelo autor.

4. Escrita do primeiro relatório técnico;
5. Utilização da ferramenta *Irace* para o teste de parâmetros para os algoritmos;
6. Análise de como os parâmetros influenciam a solução e o tempo de execução;
7. Escrita do segundo relatório técnico;
8. Realizar a calibragem dos parâmetros dos algoritmos baseado na análise já feita;
9. Análise dos resultados;
10. Escrita da monografia;
11. Montar o guia para ajustes de parâmetros;
12. Preparação para apresentação da monografia

## 7 Atividades realizadas

Foram realizados os tópicos um, dois e três, sendo que no tópico três o algoritmo implementado foi o algoritmo genético para o caixeiro viajante. Este algoritmo será



utilizado inicialmente e pode ocorrer a inclusão de um novo algoritmo ao decorrer do trabalho. Também foi criado um manual de uso para a ferramenta Irace.

## 7.1 Manual da ferramenta

### 7.1.1 Contexto da ferramenta

O pacote *Irace* implementa o método *Iterated Race*, para a configuração automática de parâmetros de algoritmos de otimização. O ajuste de seus parâmetros é feito encontrando as configurações mais adequadas a um conjunto de instâncias de uma otimização utilizando o teste estatístico de Friedman. Seu funcionamento se baseia em começar com uma configuração de parâmetros já estabelecida pelo usuário e a cada iteração da ferramenta selecionar a melhor configuração, aquela que gera um melhor resultado, e atualizar os parâmetros com essa configuração, para então repetir esse processo com as novas configurações.

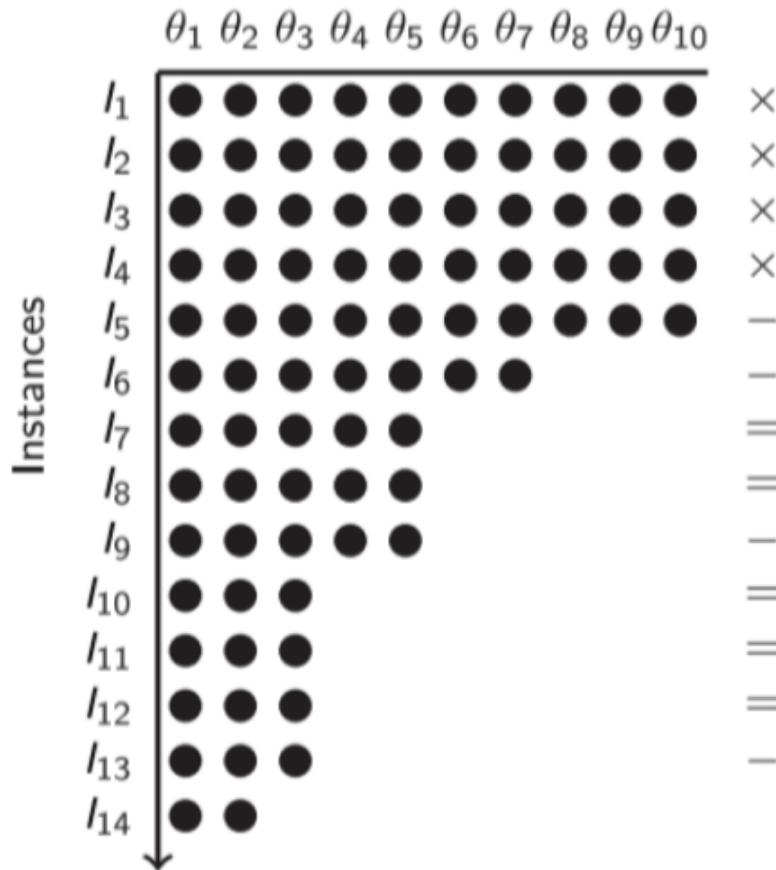


Figura 1 – Representação das iterações da ferramenta, onde cada *theta* é uma configuração e cada *I* é uma iteração do programa (LÓPEZ-IBÁÑEZ et al., 2016b) .

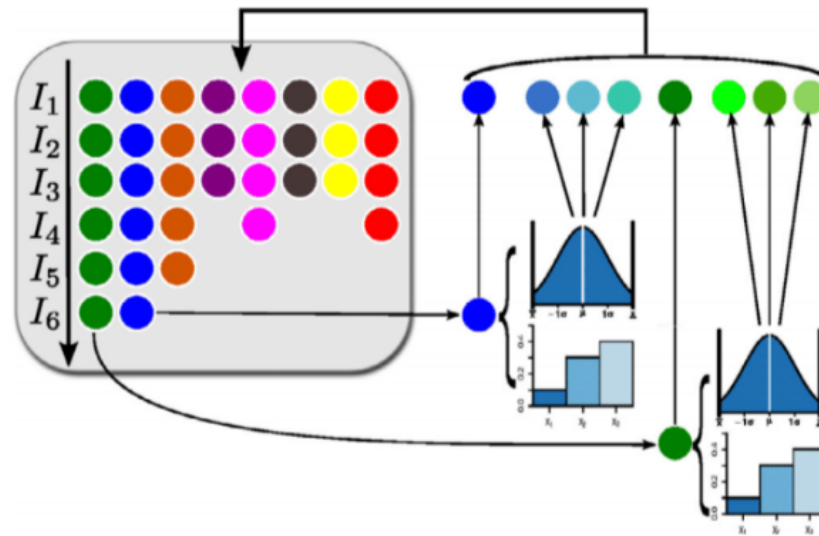


Figura 2 – Forma de seleção entre os parâmetros utilizados pela ferramenta (LÓPEZ-IBÁÑEZ et al., 2016b) .

### 7.1.2 Requisitos

Para utilizar o *Irace* é preciso que o sistema operacional utilizado possua suporte a linguagem de programação *R*, uma linguagem interpretada que serve de base para rodar a ferramenta *Irace*.

### 7.1.3 Instalação do R

Para a instalação do interpretador da linguagem, é necessário acessar este *link* <https://www.r-project.org>. Após entrar no *site*, a primeira sessão indica como iniciar com a linguagem, mostrando um novo *link*, <https://cran.r-project.org/mirrors.html>, que leva para diversos *links* de *download* da ferramenta, no caso foi utilizado o <https://cran-r.c3sl.ufpr.br/> e escolhido a versão base (LÓPEZ-IBÁÑEZ et al., 2016a). A versão utilizada é a 4.2.1. Após acessar este último *link*, será apresentado três opções de *download*, uma para sistemas baseado no *Linux*, uma para *macOS* e outra para *Windows*.

#### 7.1.3.1 Instalação no Windows

Para este sistema, um arquivo executável será baixado e a instalação será gerenciada por ele.

A instalação segue o padrão de instalação de qualquer programa, bastando ir avançando os passos, recomendo manter todas as configurações na forma padrão, como já estão selecionadas. Com tudo instalado, será necessário configurar as variáveis de ambiente,

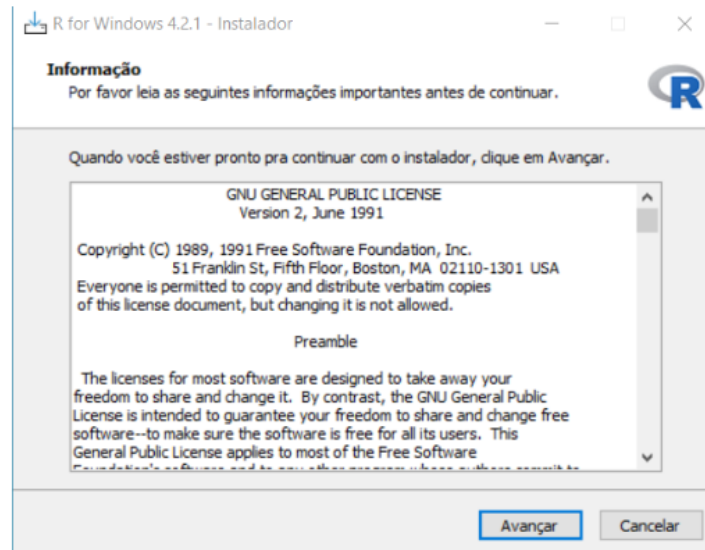


Figura 3 – Tela inicial da instalação.

do *R* e do *Irace*. Para configurar essas variáveis, digite ambiente na barra de pesquisa do *windows* e selecione a opção “Editar variáveis de ambiente do sistema”, em seguida selecione a opção variáveis de ambiente.

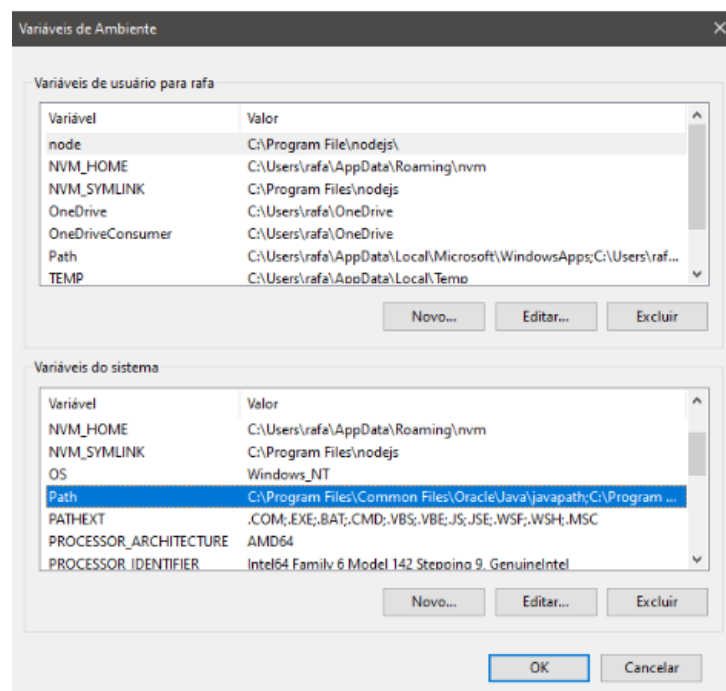


Figura 4 – Local onde se deve adicionar as variáveis de ambiente.

Clique em editar, para abrir a janela onde será adicionada às novas variáveis de ambiente. Nesse caso, as variáveis são o *R* e o *Irace*, então precisamos localizar onde esses dois elementos estão instalados no sistema e local a pasta *bin* dentro de cada diretório.

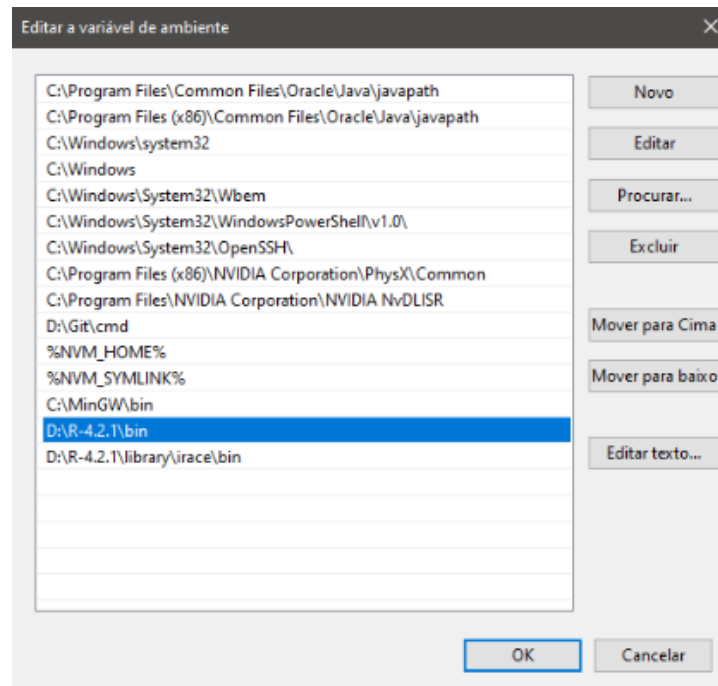


Figura 5 – Variáveis de ambiente adicionadas.

#### 7.1.3.2 Instalação no *Linux* (*Ubuntu* 20.04).

Para este sistema, a instalação será feita através do comando “*sudo apt install -no-install-recommends r-base*”. O comando aparece no *link* de *download* citado após escolher a opção do sistema. Também será necessário adicionar o *Irace* no *PATH*, variáveis de ambiente do *Linux*.

Para isso, primeiramente abra o programa do *R*, e digite “*system.file(package="irace", "bin", mustWork=TRUE)*” para localizar a instalação do *Irace* e salve esse caminho. Execute este comando “*export PATH="caminho\_copiado":\$PATH*”.

Em seguida, no terminal, digite “*nano ~/.bashrc*” e ao final do arquivo que foi aberto, adicione a seguinte linha “*export PATH="caminho\_copiado:\$PATH*” e salve o arquivo.

#### 7.1.4 Instalação do *Irace*

Com o *R* devidamente instalado, você deve abrir o seu console, executando o programa que foi instalado e executar o seguinte comando: *install.packages("irace")*. Após isso selecione o mirror e a instalação vai ser iniciada.

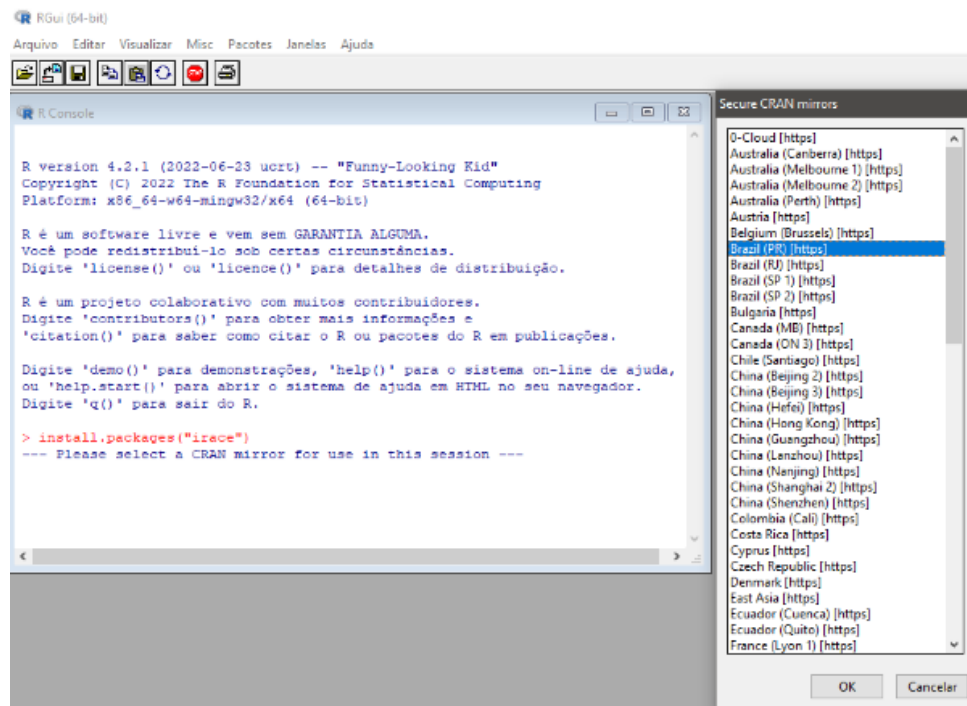


Figura 6 – Instalação do pacote *Irace*.

Ao final, uma mensagem de sucesso será apresentada no console do programa.

```
> library("irace")
> system.file(package = "irace")
[1] "D:/R-4.2.1/library/irace"
> |
```

Figura 7 – Verificando se está tudo instalado corretamente, o primeiro comando carrega a biblioteca e o segundo mostra onde ela está instalada.

Esse procedimento também funciona para o Linux, basta executar os mesmos comandos no programa do R que foi instalado.

## 8 Revisão do cronograma

Até o momento de entrega deste documento, as atividades propostas até o mês de Novembro foram realizadas, juntamente com o início da escolha dos algoritmos para a implementação, atividade descrita para Novembro e Dezembro.

## 9 Referencias

### Referências

CAMPELLO, R. E. *Algoritmos e heurísticas: desenvolvimento e avaliação de performance*. [S.l.]: EDUFF, 1994. Citado na página 4.

CONSTANTINO, A. A. et al. Aplicação de algoritmos genéticos ao problema de cobertura de conjunto. *XXXV SBPO A Pesquisa Operacional e os Recursos Renováveis*, v. 4, 2003. Citado na página 3.

CORDENONSI, A. Z. Ambientes, objetos e dialogicidade: uma estratégia de ensino superior em heurísticas e metaheurísticas. 2008. Citado 2 vezes nas páginas 4 e 5.

CORMEN, T. H. et al. Introduction to algorithms. third. *New York*, 2009. Citado na página 4.

COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE transactions on information theory*, IEEE, v. 13, n. 1, p. 21–27, 1967. Citado na página 5.

LEÓN-ALDACO, S. E. D.; CALLEJA, H.; ALQUICIRA, J. A. Metaheuristic optimization methods applied to power converters: A review. *IEEE Transactions on Power Electronics*, IEEE, v. 30, n. 12, p. 6791–6803, 2015. Citado na página 5.

LÓPEZ-IBÁÑEZ, M. et al. *The irace package: User guide*. [S.l.]: IRIDIA, Institut de Recherches Interdisciplinaires et de Développements en ..., 2016. Citado na página 10.

LÓPEZ-IBÁÑEZ, M. et al. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, Elsevier, v. 3, p. 43–58, 2016. Citado 4 vezes nas páginas 5, 7, 9 e 10.

LUZ, E. F. P. da et al. Estimating atmospheric area source strength through particle swarm optimization. 2007. Citado na página 3.

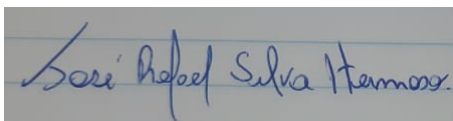
MOÇAMBIQUE, N. E. M. *Aplicação de Algoritmos de Busca do Ponto de Máxima Potência e controladores lineares e/ou Fuzzy para a regulação da tensão terminal de Painéis Fotovoltaicos*. Tese (Doutorado) — Universidade de São Paulo, 2012. Citado na página 3.

NERY, S. W. L. Análise de operadores de cruzamento genético aplicados ao problema do caixeiro viajante. 2017. Citado na página 3.

SOUTO, R. P. et al. On the use of the ant colony system for radiative properties estimation. In: *Proceedings of the 5th International Conference on Inverse Problems in Engineering: Theory and Practice*. [S.l.: s.n.], 2005. v. 11, p. 15th. Citado na página 3.

---

SUCUPIRA, I. R. Métodos heurísticos genéricos: metaheurísticas e hiper-heurísticas. *USP: São Paulo*, v. 32, 2004. Citado 2 vezes nas páginas [3](#) e [4](#).

A photograph of a handwritten signature in blue ink on lined paper. The signature reads "José Rafael Silva Hermoso".

---

José Rafael Silva Hermoso

---

Prof. Dr. Ademir A. Constantino

Maringá, 27 de novembro de 2022.