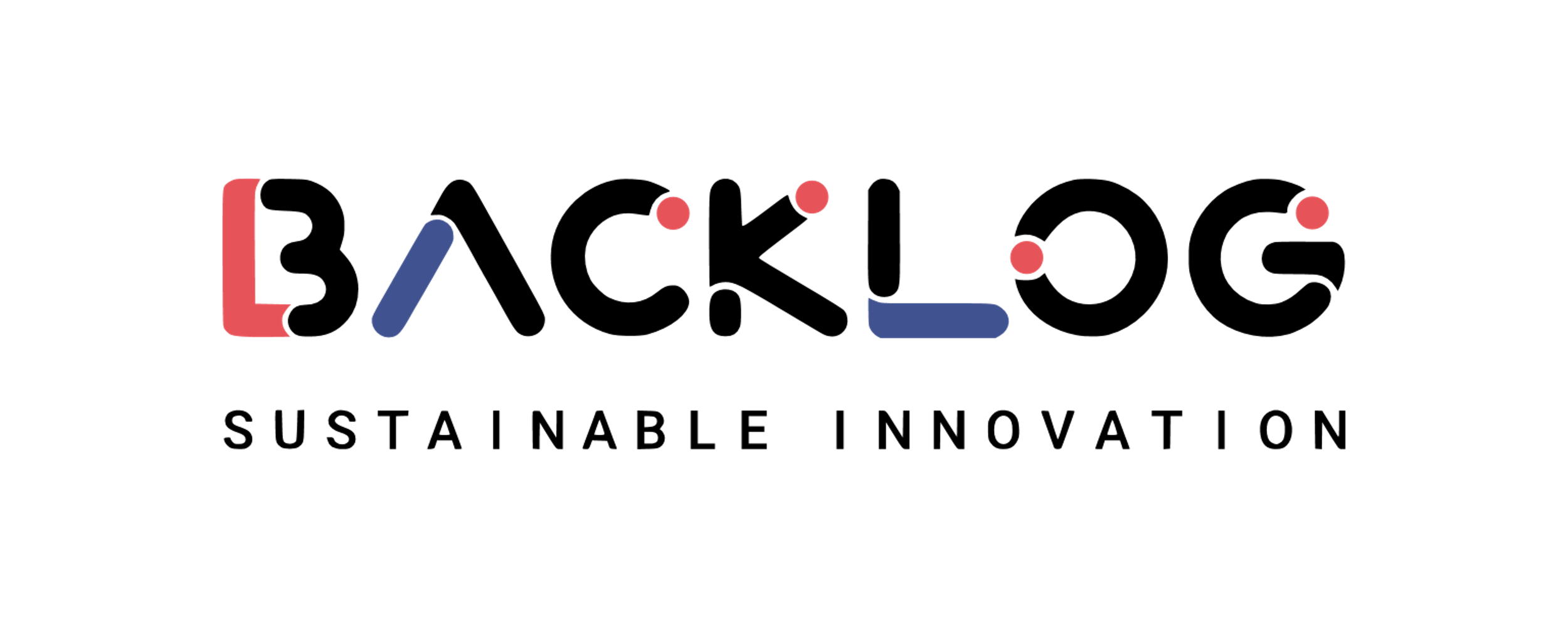
****

SUPER FORMAÇÃO INTELIGÊNCIA ARTFICIAL

Seção 2 - Algoritmos de Busca e Otimização

**Sumário**

[**1.** **Elementos de um problema de busca:** 2](#_Toc105250590)

[1.1 Busca Global vs Busca Local 2](#_Toc105250591)

[**2.** **Algoritmos de Busca** 2](#_Toc105250592)

[2.1 Hill Climbing: 2](#_Toc105250593)

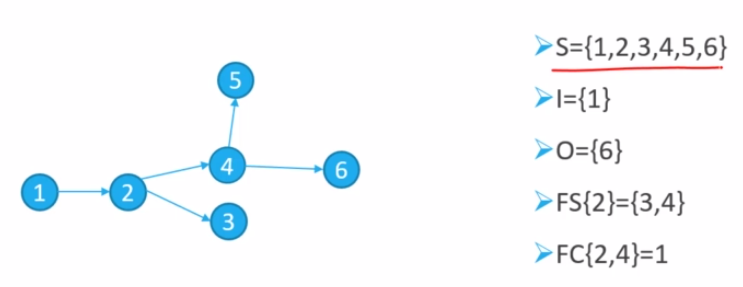
**Histórico de atualizações**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versão** | **Data** | **Descrição** | **Responsável** |
| 1.0 | 04/06/2021 | Criação do documento | Eudes |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **Elementos de um problema de busca:**

* S: conjunto finito de estados – Search Space
* I: Conjunto finito de estados iniciais
* O: Conjunto finito de objetivos
* FS: função que recebe o estado atual e retorna os estados alcançáveis
* FC: Função de custo, recebe o estado atual e um possível próximo estado, e retorno o custo.

Ex:



## Busca Global vs Busca Local

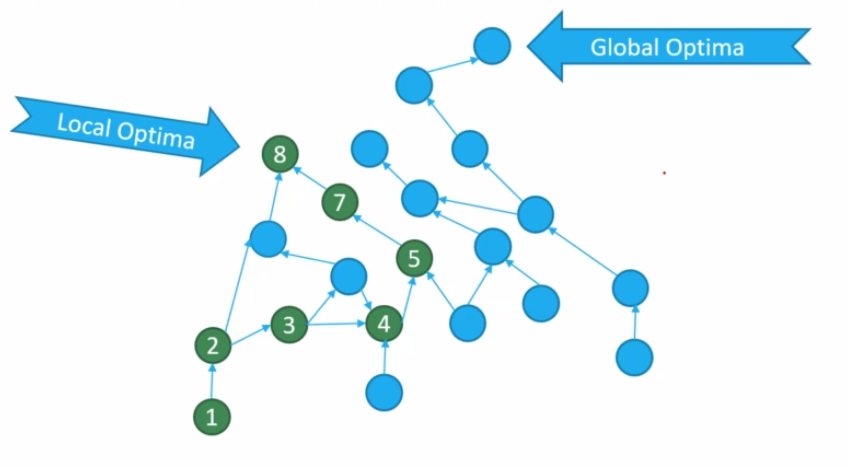
***Global:*** Busca a melhor solução global, explora todo espaço de busca. Encerra quando encontra a melhor solução global, expira um tempo determinado ou quando explora todo o espaço.

***Local:*** Busca melhor solução na região ou “vizinhança”. Encerra quando expira um tempo determinado ou quando não consegue melhorar o resultado a partir de uma função de avaliação.

# **Algoritmos de Busca**

## Hill Climbing:

* Inicia a busca em um único ponto aleatório
* Escolhe um novo ponto na vizinhança
* Se o novo ponto é uma solução melhor, passa a ser a melhor solução
* Se não, escolhe um novo ponto na vizinhança
* Até não ter mais como subir ou acabar o tempo.
* Tem grande probabilidade de ficar preso no local optima



* Reinicia o algoritmo com esperança de ter uma solução melhor.

## Características Hill Climbing:

* Segue apenas um sentido, explorando a vizinhança
* Não garante obter o global optima
* Existem muitas variações, principalmente incluindo elementos não determinísticos no algoritmo.

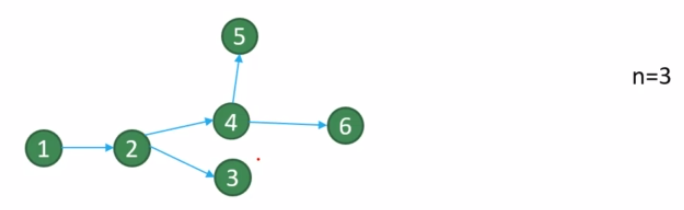
## Algoritmos Brute Force/ Blind Search

* Exploram todo o espaço de busca
* Garantem encontrar global optima

## Breadth First

* Capaz de retornar de uma vizinhança em busca de uma solução melhor
* “Backtracing”
* “Quais estados pode ser alcançado a partir do 1 com apenas **n** etapas?

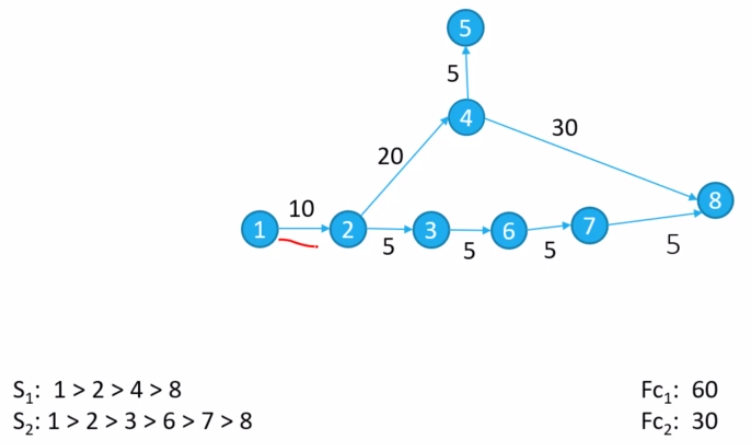
n=3



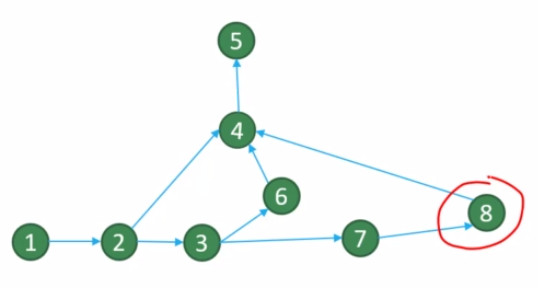
## Custos

* Leva em consideração até chegar o ponto de destino.

Ex: O S2 é melhor porque o custo é menor, porém exige mais caminho. Ao contrário do S1 o caminho é menor, mas o custo é maior.



## Depth-First Search

* Explora uma vizinhança, retornando e tentando outras ramificações

## Best-First Search

* Usa heurística para avaliar o valor de cada nó
* Sua performance depende da heurística.

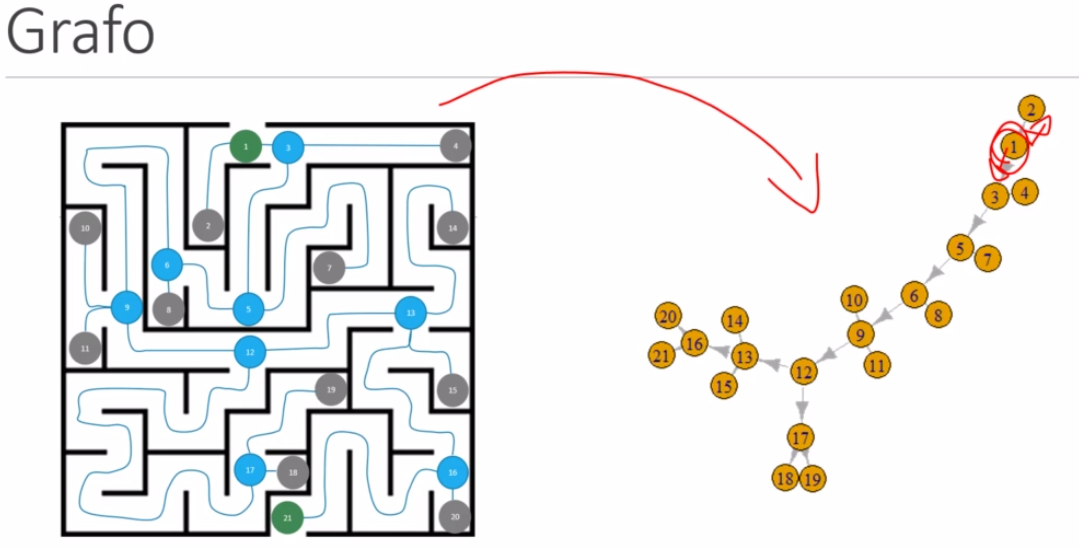
Obs: Heurística nem sempre traz uma boa solução ou uma solução rápida.

## Problemas de “Caminhos”

* Normalmente só pode ser resolvida com algoritmos baseados em Blind Search.
* Lee Algorithm, baseado em Breadth-first Search.

Transformar um labirinto em grafo

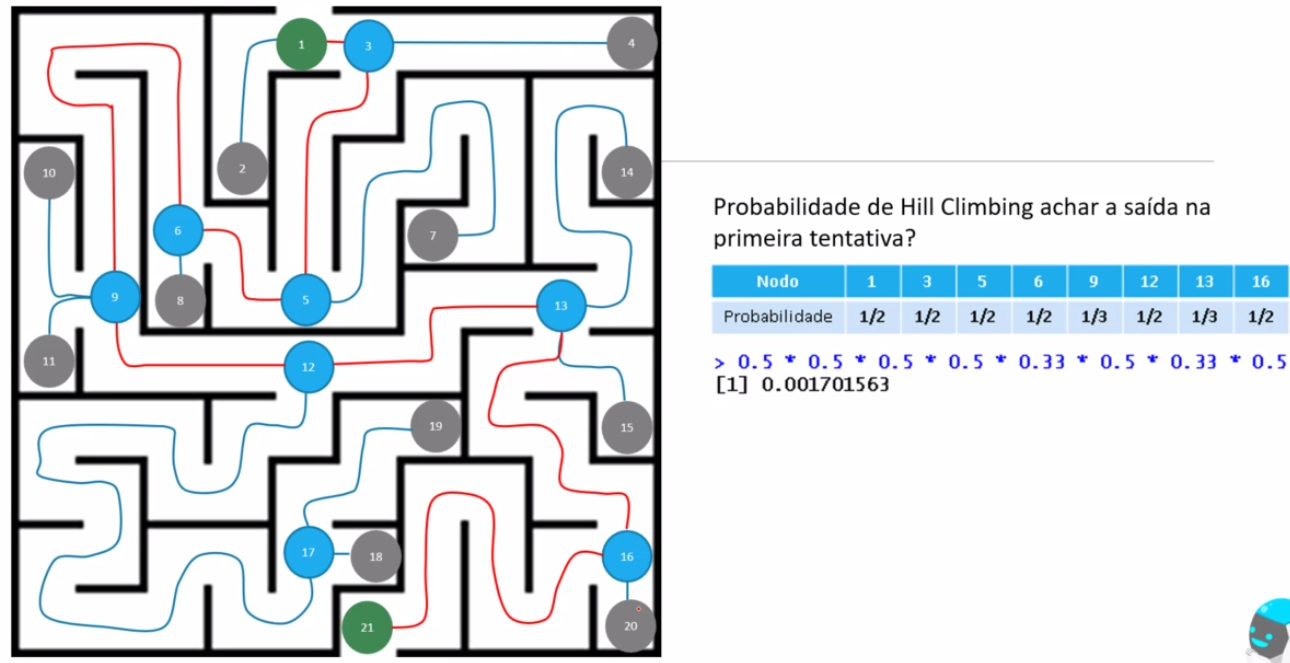
1. Nodo no início
2. Nodo no fim
3. Nodo a cada divisão no labirinto (intermediários)
4. Nodo a cada ponto sem saída (Local Optima)
5. Apenas nodo de inicio e os nodos intermediários podem ter mais de uma aresta



Total de nodos:21

Local optima: 11(sem saída)

* Hill Climbing



* Avaliar evolução através da quantidade de passos até o fim.



## Tabu Search

* Mantém uma lista de locais proibidos (Tabu) em memória
* Proibidos ou por já terem visitados ou por não otimizarem a função objetivo
* Bom para problemas combinatórios
* Critério de parada: número de iterações, tempo, iterações sem melhoria na função objetivo.

## Simulated Annealing (Arrefecimento Simulado)

* Principio básico: não busca sempre a otimização, mas também explorar

(Subir vs Explorar)

* Annealing: processo de aquecer e resfriar metal para alterar sua estrutura. Quando ocorre o resfriamento metal tende a manter a nova estrutura.
* Uma variável “temperatura” é criada:
* A temperatura é alterada dinamicamente
* Com “temperatura” alta, o algoritmo vai explorar soluções que aparentemente não otimizam a função objeto, saindo de um local optima.
* Com “temperatura” baixa, o algoritmo tende a explorar vizinhanças e aceitar o local optima, que pode estar mais próximo do global optima.

O objetivo é explorar outras soluções que no inicio podem não serem boas, mas essa busca vai diminuindo (temperatura)

## Algoritmo de Busca e Otimização Tabu Search em R

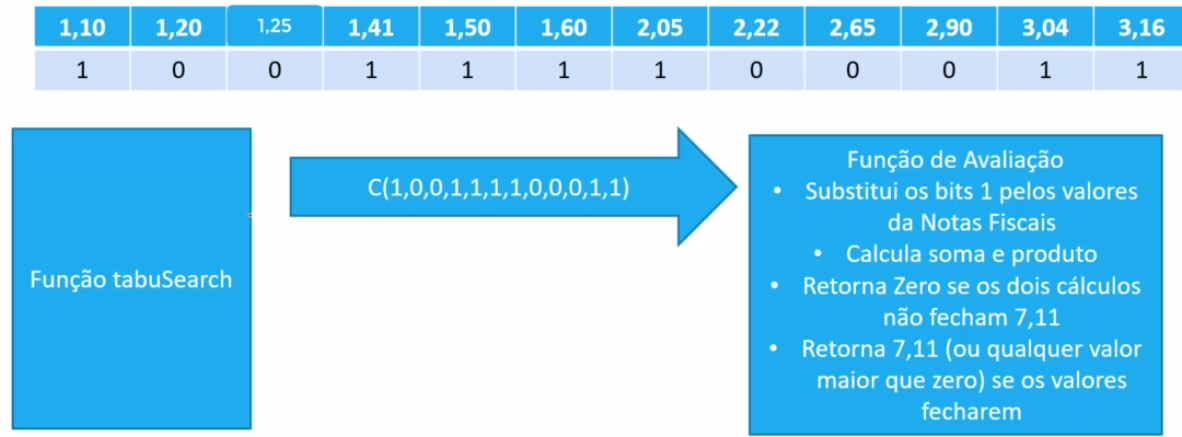
* Problema da loja 711: Quais os produtos que somados e multiplicados tem o resultado 7,11?

|  |  |  |  |
| --- | --- | --- | --- |
| Blusa | 1,10 | Jaqueta | 2,05 |
| Camisa | 1,20 | Macacão | 2,22 |
| Calça | 1,25 | Saia | 2,65 |
| Casaco | 1,41 | Saruel | 2,90 |
| Blusão | 1,50 | Short | 3,04 |
| Jeans | 1,63 | Suéter | 3,16 |

No R usar o pacote tabuSearch – Configuração não binária (Tem como objetivo de maximizar o valor de retorno da função de Avaliação)



Como funciona no exemplo:



Link código Git:

## Algoritmo de Busca e Otimização Simulated Annealing em R

Exemplo: Otimizar uma função matemática

Rosenbrock’s Function:

Minimo global: f(x,y) = 0, com x e y = 1

* O pacote no R “GenSA”
* Busco Mínimo Global em função objetiva não linear