



ALGORITMO GENÉTICO

Lucas Vieira Barreto

Pedro Nasser

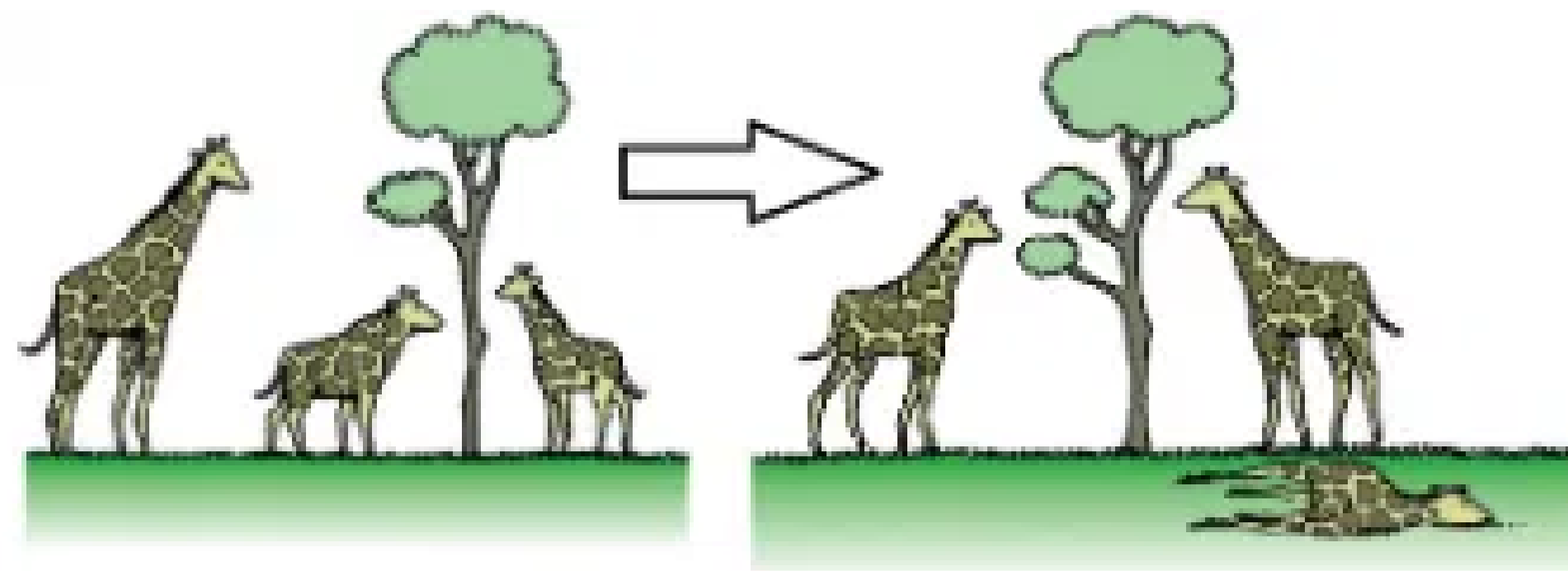
Gabriel Teles

Modelagem e Otimização de
algoritmos



COMO SURTIU?

O algoritmo surgiu com base na Teoria da Seleção Natural de Darwin, a qual afirma que os indivíduos mais bem adaptados ao ambiente em que vivem possuem maior chance de sobreviverem e se reproduzir





NÚMERO DE GERAÇÕES

FITNESS(AVALIAÇÃO)

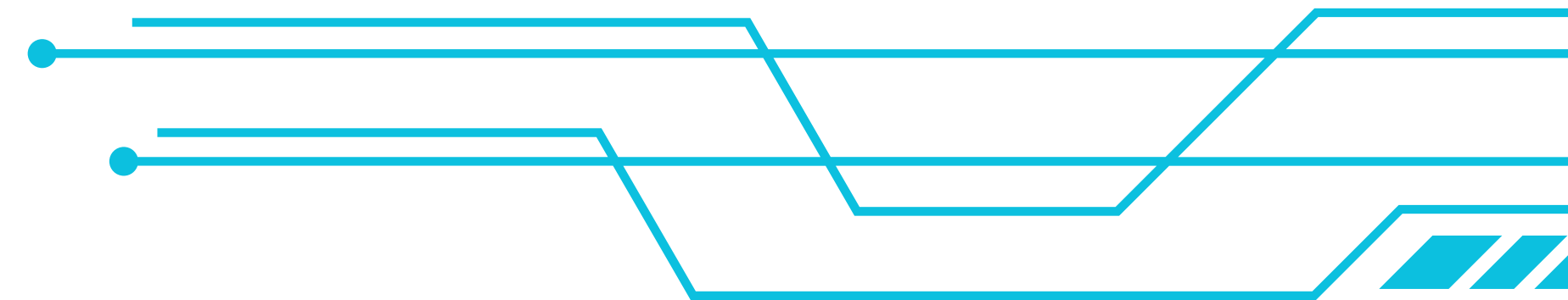
SELEÇÃO

SUBSTITUIÇÃO

CRUZAMENTO

MUTAÇÃO

NÚMERO DE INDIVÍDUOS



PARÂMETROS IMPORTANTES

NUMERO DE GERAÇÕES: Limite de gerações a serem criadas

FITNESS: Número de Colisões

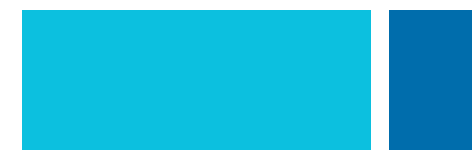
SELEÇÃO: Escolhe os indivíduos para reprodução.

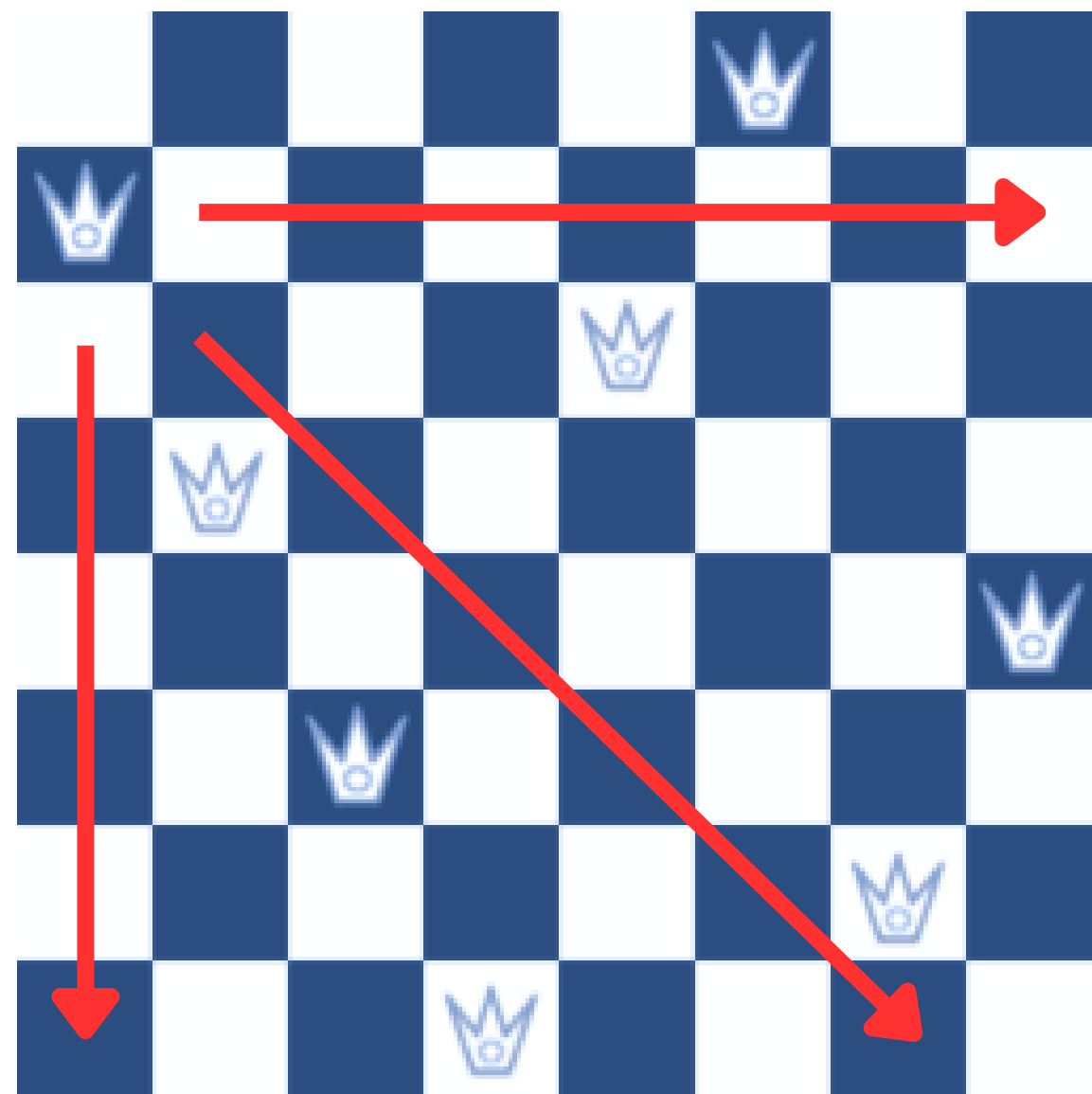
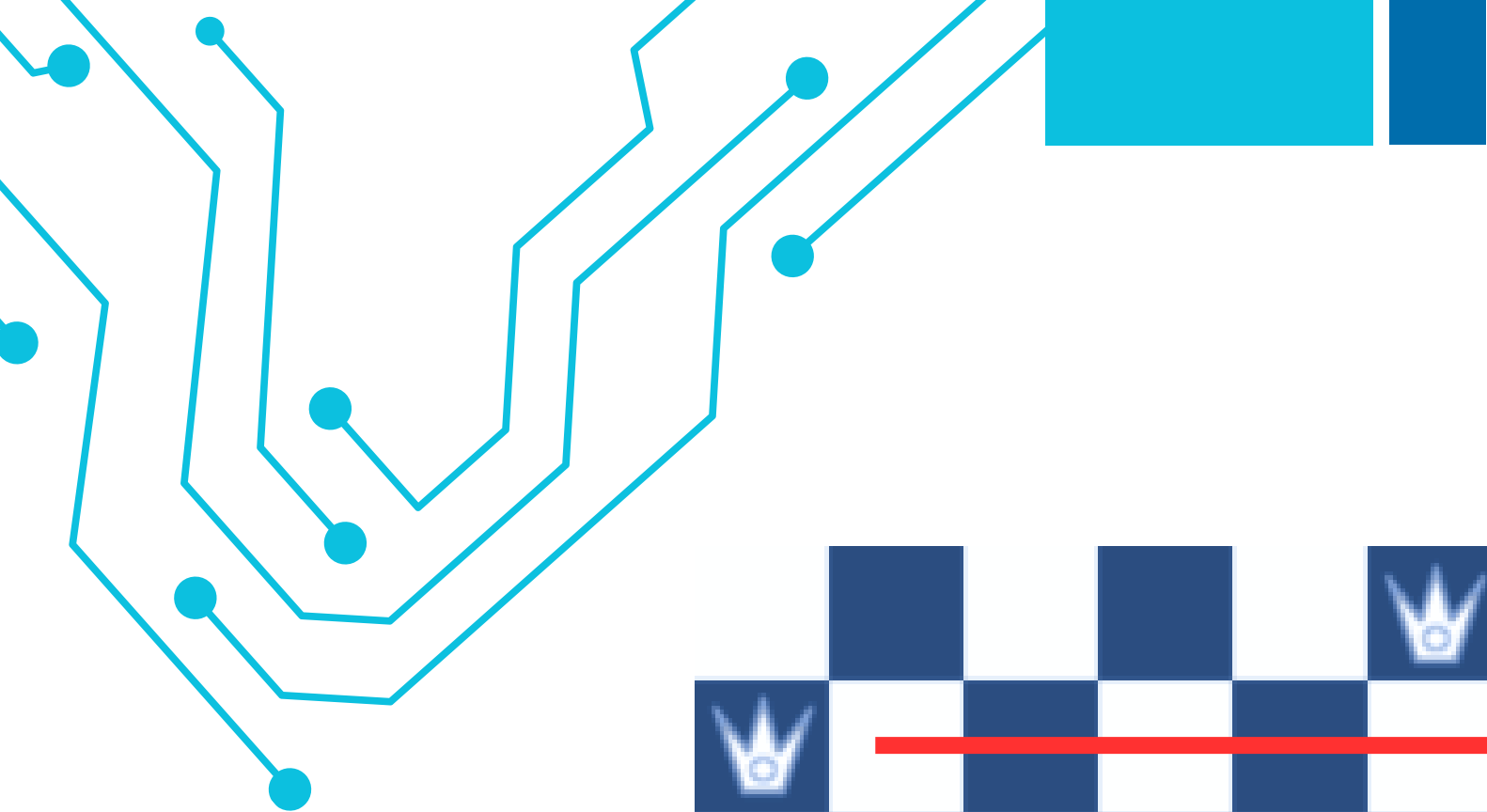
SUBSTITUIÇÃO: Atualiza a população, mantendo ou não os melhores.

CRUZAMENTO: define a frequência de reprodução entre indivíduos.

MUTAÇÃO: Introduz mudanças aleatórias para diversificar soluções.

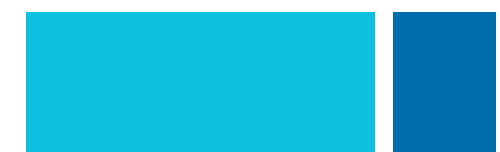
NÚMERO DE INDIVÍDUOS: Quantidade de soluções por geração; maior número traz mais diversidade.





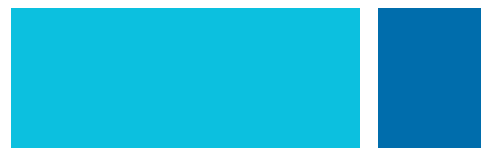
PROBLEMA N-RAINHAS

O Problema das N Rainhas consiste em colocar em um tabuleiro uma certa quantidade de rainhas e evitar que elas se cruzem nas diagonais e na horizontal.



NOSSA SOLUÇÃO

```
31 self.file = None
32 self.fingerprints = set()
33 self.logopen = True
34 self.debug = debug
35 self.logger = logging.getLogger(__name__)
36
37 if path:
38     self.file = os.path.join(path, 'log.txt')
39     self.file.write('')
40     self.fingerprints.add('')
41
42 @classmethod
43 def from_settings(cls, settings):
44     debug = settings.get('DEBUG', False)
45     return cls(log_dir=settings.LOG_DIR, debug=debug)
46
47 def request_seen(self, request):
48     fp = self.request_fingerprint(request)
49     if fp in self.fingerprints:
50         return True
51     self.fingerprints.add(fp)
52     if self.file:
53         self.file.write(fp + '\n')
54
55 def request_fingerprint(self, request):
56     return request_fingerprint(request)
```





OBRIGADO

