

BRKGA - Biased Random-Key Genetic Algorithm

Implementação

A implementação do BRKGA foi realizada de forma direta e simples, seguindo os princípios básicos do algoritmo:

Codificação

- **Representação:** Vetor de valores reais no intervalo $[0,1]$ (random keys)
- **Decodificação:** Pacotes com valor acima do threshold (0.5) são incluídos na solução

```
function encode(problem::ProblemContext, _::PackagesStrategy)
    return [rand() for _ in 1:problem.package_count]
end

function decode(problem, individual, strategy::PackagesStrategy)
    return [
        idx for (idx, gene) in enumerate(individual)
        if gene > strategy.used_package_threshold
    ]
end
```

Parâmetros Utilizados

- **População:** 30 indivíduos
- **Elitismo:** 20% (6 indivíduos elite)
- **Mutação:** 14% (4 indivíduos mutantes)
- **Crossover:** 66% (20 indivíduos gerados por cruzamento)
- **Iterações:** 6.000
- **Threshold de decodificação:** 0.5

Operadores

Crossover Biased: Combina um indivíduo elite com um aleatório da população, priorizando genes do elite:

```
elite_part = elite_candidate[1][1:seed]
random_part = candidate[1][seed:length(candidate)]
offspring = cat(elite_part, random_part, dims = 1)
```

Mutação: Geração de novos indivíduos completamente aleatórios para manter diversidade.

Resultados Experimentais

Instância Testada

- **Arquivo:** prob-software-85-100-812-12180.txt
- **Características:** 85 pacotes, 100 dependências, 812 relações, tamanho total 12.180

Desempenho Observado (30 execuções)

Métrica	Valor
Melhor solução	8.689
Média das melhores	7.647
Pior solução	7.289
Tempo médio	~4.0 segundos
Tempo mínimo	3.74 segundos
Tempo máximo	4.47 segundos

Distribuição de Qualidade

Das 30 execuções: - **15 execuções** (50%) obtiveram soluções ≥ 7.600 - **8 execuções** (27%) alcançaram qualidade ≥ 8.000 - **Melhor resultado individual:** 8.689 em 4.04 segundos

Análise da População Final

Uma característica importante observada é a **alta diversidade** mantida na população final:

Métrica da População	Valor Típico
Média da população	~2.000-2.400
Máximo da população	7.400-8.900
Diferença (gap)	~5.000-6.500

Interpretação: A diferença acentuada entre a qualidade média (~2.000) e o melhor indivíduo (>7.000) da população final indica que:

1. **Diversidade preservada:** A estratégia de mutação (14%) consegue manter indivíduos variados, evitando convergência prematura
2. **Elite efetiva:** Os melhores indivíduos (elite de 20%) concentram a qualidade, enquanto o restante explora outras regiões do espaço de busca
3. **Exploração ativa:** A população não convergiu totalmente, o que é positivo - há ainda potencial exploratório caso se estendesse as iterações

Esta distribuição é **esperada e desejável** em BRKGA: a elite carrega soluções de alta qualidade que são propagadas via crossover biased, enquanto mutantes e indivíduos de crossover mantêm a capacidade de exploração. O gap de ~5.500 pontos entre média e máximo demonstra que o algoritmo **não estagnou**, mantendo um equilíbrio adequado entre intensificação (elite) e diversificação (mutantes + crossover).

Comparação com Abordagens Anteriores

Abordagem	Ganho Típico	Tempo
Heurísticas Construtivas	≤ 4067	$< 0.1s$
Busca Local (Best Improvement)	≤ 4579	$< 1s$
BRKGA	> 7000	~4s

Como esperado, o BRKGA é o primeiro algoritmo que melhor consegue encontrar soluções significativamente melhores que a construção gulosa, chegando a mais que dobrar esse resultado em diversas populações

Análise e Oportunidades de Melhoria

Pontos Positivos

1. **Implementação simples:** Código direto e fácil de entender (~70 linhas)
2. **Resultados consistentes:** Baixa variação entre execuções (desvio ~300 pontos)
3. **Qualidade excelente:** Soluções de 8.000+ pontos em poucos segundos
4. **Tempo aceitável:** 4 segundos é viável para aplicação prática

Margem para Otimização

A implementação atual é **deliberadamente básica** e há várias oportunidades de ganho:

1. **Decodificação estática:** Threshold fixo (0.5) - poderia ser adaptativo ou parametrizado
2. **Crossover simples:** Ponto de corte aleatório único - operadores mais sofisticados (2-pontos, uniforme) podem melhorar
3. **Sem busca local:** Aplicar busca local nas soluções elite pode gerar saltos de qualidade
4. **Sem reinicialização:** Não há mecanismo anti-estagnação
5. **Seleção de elite fixa:** Poderia usar torneio ou ranking para aumentar pressão seletiva
6. **Critério de parada fixo:** 6.000 iterações independente de convergência

Potencial de Ganho

Considerando que: - Heurísticas construtivas aleatórias/gulosas produzem soluções base - Busca local simples agrega modestos 290-512 pontos - **BRKGA básico já atinge 7.600-8.600 pontos**

Estimamos que refinamentos na implementação (hibridização com busca local, operadores avançados, ajuste fino de parâmetros) podem: - **Melhorar tempo de convergência** (atingir boas soluções em menos iterações) - **Melhorar valores máximos e médios encontrados em um mesmo intervalo de tempo**

Conclusão

A implementação básica do BRKGA demonstrou **excelente custo-benefício**: com código simples e tempo de execução aceitável (4s), obtivemos soluções de altíssima qualidade (8.000+ pontos)

O algoritmo se mostra promissor e há clara margem para melhorias futuras através de refinamentos algorítmicos, validando a escolha do BRKGA como metaheurística para este problema de otimização.