Trabalho de Algoritmos Genéticos
Parte 2

Fábio Beranizo Fontes Lopes
Escola de Artes, Ciências e Humanidades (EACH)
Universidade de São Paulo (USP)
Email: f.lopes@usp.br

Este relatório apresenta a discussão da parte 2 do trabalho de algoritmos genéticos (GA) para a disciplina de Inteligência Computacional. Foi avaliada a utilização de GA para roteamento de veículos. O foco se manteve em analisar o comportamento do algoritmo mediante a utilização de diferentes técnicas, operadores e parametrizações. O trabalho foi desenvolvido utilizando a linguagem de programação Python, especificamente sua distribuição Anaconda.

I. Problema de Roteamento de Veículos

O problema de roteamento de veículos (VRP) é o nome dado a uma classe de problemas em que rotas de veículos são otimizadas para servir um grupo de clientes. Determinar a solução ótima é um problema NP-difícil, assim é limitado o tamanho das instâncias que tem solução ótima rapidamente calculada. Isto tudo faz deste um problema interessante para ser solucionado por heurísticas, como o algoritmo genético utilizado neste trabalho.

Existem muitas variações para o VRP:

- CVRP (Capacitated VRP): os veículos possuem capacidade limitada
- VRPTW (VRP with time windows): cada cliente tem que ser atendido dentro de uma janela de tempo
- MDVRP (Mutliple Depot VRP): múltiplos depósitos são usados para atender os clientes
- VRPPD (VRP with Pick-up and Delivering): clientes podem devolver produtos para o depósito
- SDVRP (Split Delivery VRP): mais de um veículo pode atender um cliente
- PVRP (Periodic VRP): as entregas podem ser feitas em alguns dias
- OVRP (Open VRP): os veículos não precisam voltar ao depósito

Neste trabalho a variação de VRP avaliada foi o CVRP. Neste problema, uma quantidade de veículos com capacidade uniforme deve servir demandas de clientes com custo mínimo de deslocamento. Uma descrição mais formal deste problema se encontra a seguir:

Objetivo: Minimizar a frota de veículos e a soma das distâncias de viagem. A demanda total para cada rota não pode exceder a capacidade do veículo que serve essa rota.

Factibilidade: A solução é factível se a demanda total atribuída a cada rota não excede a capacidade do veículo que serve à rota.

Formulação:

- Seja $V = \{v_0, v_1, \ldots, v_n\}$ um conjunto de vértices tal que:
  - Um depósito está localizado em v_0
  - $V' = V \backslash \{v_0\}$ é um conjunto de n clientes
- $A_i j = (v_i, v_j)/v_i, v_j \in V$ é um conjunto de arestas.
- C é uma matriz de distâncias não negativas c_{ij} entre clientes v_i e v_j
- d é um vetor de demandas de clientes
- R_i é a rota para o veículo i
- m é o número de veículos. Cada rota é atribuída a cada veículo
- Q é a capacidade de um veículo

Uma solução factível é composta de:

- uma partição $R_1, \ldots, R_m$ de V
- uma permutação σ_i de $R_i \bigcup 0$ especificando a ordem dos clientes na rota i

O custo de uma dada rota ($R_i = \{v_0, v_1, ..., v_{m+1}\}$), onde $v_i \in V$ e $v_0 = v_{m+1} = 0$, é dado por: $F(R_i) = \sum_{i=0}^{m} c_{i,i+1} + \sum_{i=1}^{m} \delta_i$.

A demanda total de todos os clientes servidos em uma rota R_i não excede a capacidade do veículo Q: $\sum_{i=1}^{m} d_i \leq Q$.

II. Algoritmo Genético

Algoritmos Genéticos (GA) são heurísticas inspiradas na seleção natural biológica. GA gera soluções aproximadas para um dado problema, buscando melhorar os resultados a cada geração. Utilizando soluções antigas (pais) e certos operadores, são criadas novas soluções (filhos).

A. Operadores Genéticos

Indivíduos selecionados de uma população passam por um dos 3 tipos de operadores genéticos: reprodução, crossover e mutação. Neste trabalho foram analisados tipos de crossover e de mutação clássicos, além de outros

orientados ao VRP.

1) Seleção de Indivíduos: Somente um operador de seleção de indivíduos foi analisado neste trabalho, o método da roleta. Ele operador foi escolhido por ser de fácil implementação e por apresentar bons resultados segundo a literatura. No entanto, durante o desenvolvimento foi possível perceber um problema: este operador aumenta consideravelmente o tempo de execução do algoritmo para grandes populações, uma vez que todos os indivíduos participam do sorteio.
O método da roleta leva em consideração o fitness de um indivíduo em relação ao fitness total da população. Nesse processo cada indivíduo possui uma probabilidade de ser selecionado (uma fatia da roleta) proporcional ao seu fitness. Embora não garanta que as melhores soluções sobrevivam, este método dá maiores probabilidades de sobrevida às soluções com bom fitness.

2) Reprodução: O operador de reprodução consiste em manter vivos alguns indivíduos baseados em seu fitness. Quanto melhor o fitness, maior a chance de sobrevivência.

3) Crossover: O operador crossover permite a criação de novos indivíduos a partir de organismos pais. Os descendentes possuem genótipo de ambos pais, embora geralmente sejam diferentes. O crossover é uma forma de ampliar o espaço de busca por soluções, porém mantendo-se material genético de boa qualidade de gerações anteriores.

4) Mutação: O operador de mutação permite inserir diversidade genética entre gerações de indivíduos trocando o valor de genes no cromossomo. Com maior variedade genética o espaço de busca é ampliado, possibilitando sair de situações de mínimo local.

B. Cromossomo

O modelo de genótipo escolhido foi uma vetor que guarda a sequencia de clientes servidas por um veículo, e separada por um caracter "X" que representa o início de um outro veículo.

A escolha deste modelo de cromossomo foi devida a facilidade de manipulação do mesmo. Dessa forma, foi simples de executar os operadores de crossover e mutação bem como no cálculo de fitness.

No entanto, esse modelo de cromossomo facilitou a geração de solução infactíveis, como será explicado na próxima seção.

C. Problemas com operadores

Durante o desenvolvimento da solução, foi detectado que a aplicação dos operadores de mutação e de crossover poderia criar soluções infactíveis para o problema. Para um cromossomo cujos genes representam um caminho, a aplicação de crossover leva fatalmente à geração de caminhos incompletos. A figura 1 mostra um exemplo de crossover que gera solução infactível.

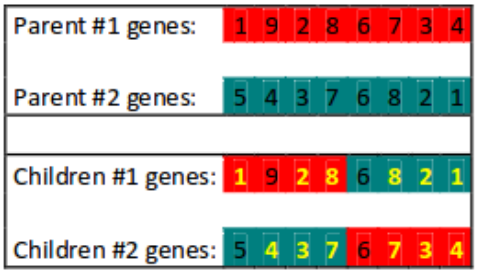

Fig. 1: Crossover que gera solução infactível.

Para contornar este problema uma primeira abordagem tomada foi repetir a realização do crossover por quantas vezes quanto necessário até que seja gerado um crossover válido. No entanto, após a realização de testes esta possibilidade foi descartada, uma vez que demorava muito até que surgisse uma solução válida. Além disso, muito recurso computacional era disperdiçado, fazendo com que essa solução fosse inviável. Assim, foi necessário buscar alternativas na literatura para resolver o problema.

D. Simple Random Mutation

A primeira solução escolhida observando-se a literatura foi o Simple Random Mutation (SRM).
O operador inicia selecionando aleatoriamente um cliente. O cliente é então removido de sua rota para evitar ficar duplicado na solução. A seguir é executada a função *BestInsertion* que retorna o índice que indica o melhor lugar para inserir o cliente removido. O cliente é então inserido na posição retornada por best-insertion.

BestInsertion estima o custo para inserir uma subrota k1..kn entre dois clientes m e m+1 como:

$$c = dist(m; m+1) - dist(m, k1) - dist(kn, m+1)$$

Os cliente m e m+1 que maximizam c são então escolhido para inserção da subrota.

Essa técnica não verifica se a solução continua factível, pois não verifica se a nova rota excede as capacidades do veículo.

```
Mutation(offspring, pSame)
NR <- number of routes in offspring
jMut <- random(0,NR)
RL <- number of customers in route jMut
kMut <- random(0,RL)
custo <- offspring[jMut][kMut]

delete cust from jMut
prob <- random(1,100)
if prob < pSame
  bestInd <- [jMut,BestInsertion(cust,jMut)]
else do j <- random(0,NR)
      while j = jMut
```

```
        bestInd <- [j, BestInsertion(cust,j)]
insertCust(cust, bestInd)
return offspring
```

E. Simple Random Crossover

A técnica de Simple Random Crossover (SRC) é análoga a SRM. Dados 2 cromossomos P1 e P2, é feita a seleção de uma subrota de P2. Em seguida, os elementos da subrota são removidos de P1. A função BestInsertion é executada então em P1 e, na posição retornada é feita a inserção da subrota.

```
SRCrossover(P1,P2)
copy individual P1 into offspring
randomly select a subroute from P2
delete the members of the subroute from the offspring
bestInd <- BestInsertion(offspring, subroute)
insertSubRoute(offspring, bestInd, subroute)
return offspring
```

III. Função Fitness

Para avaliar a seleção natural, cada indivíduo é avaliado em termos de seu valor de fitness. Essa medida, mede a qualidade das soluções e permite sua comparação.

Na parte 1 do trabalho o tipo do problema não exigiu muito trabalho de acerto da função de fitness. No entanto, nessa parte foi necessário contornar várias dificuldades, por exemplo: avaliar soluções infactíveis.

A forma adotada neste trabalho para lidar com soluções infactíveis foi somar um valor de penalidade ao fitness da solução.

Se a demanda de um veículo exceder sua capacidade, será aplicada penalidade igual à soma de das distâncias de todas rotas possíveis.

IV. Operador de Reparação

O operador de reparação (RO) é utilizado para diminuir as rotas existentes. Um cliente é selecionado da rota que mais viola a restrição de capacidade. Em seguida o mesmo é transferido para o final da rota com menor soma de demanda.

```
Repair(offspring)
jMax <- the index of the route in the offspring with largest total demand
maxDem <- the total demand of route jMax
jMin <- the index of the route in the offspring with smallest total demand
if maxDem > K
  randomly choose a customer from route jMax
  delete the customer from route jMin
  insert the customer at the end of route jMin
return offspring
```

V. Instâncias de Teste

As instâncias de teste utilizadas foram obtidas a partir do site http://www.bernabe.dorronsoro.es/vrp/. Neste site estão presentes variadas instâncias para o problema do CVRP, bem como seus valores ótimos/melhores conhecidos.

TABLE I: Valores ótimos para as instâncias avaliadas

Instância de Teste	Valor ótimo
A-n32-k5	784
B-n31-k5	672
P-n16-k8	435
A-n65-k9	1174

Para a realização dos testes foram selecionadas uma instância de dimensão pequena cada grupo A, B e P de Augerat et al.. Além disso, foi utilizada uma instância de dimensão alta do grupo A.

A Tabela I mostra os valores ótimos para as instâncias de teste avaliadas.

VI. Inicialização de População

Para a geração de uma população inicial válida, foi tomada a seguinte abordagem:

1) Gerar uma ordenação aleatória dos clientes
2) Iniciando-se do primeiro cliente ordenado, alocar demanda de cada cliente a um veículo
3) Se uma demanda exceder a capacidade restante do veículo, alocar outro veículo

Essa abordagem sempre cria soluções factíveis, embora não garanta bons valores de fitness.

VII. Tuning de Parâmetros

Existem alguns parâmetros que são utilizados no processo de resolução do VRP através algoritmos genéticos. Para obter bons resultados é necessário ajustá-los. Idealmente, cada parâmetro deveriam ser testados exaustivamente, num universo contínuo de valores. No entanto, este processo consome demasiado tempo, sendo necessário restringir o universo de busca.

No desenvolvimento deste trabalho, a abordagem tomada neste processo foi a execução de um *Grid Search* de parâmetros. Os seguintes parâmetros foram buscados:

- Tamanho da População
- Taxa de Reprodução
- Taxa de Crossover
- Taxa de Mutação

?? mostra as combinações de parâmetros testadas durante o grid search. Os valores de população foram escolhidos em escala logarítmica para que tivessem grandezas bem diferentes, e assim apresentassem diferença significativa de resultados. As taxas, por sua vez foram testadas em combinações que não permitiam probabilidade zero, para que todos operadores fossem testados. Ao mesmo tempo, foi possível testar taxas bem distintas (ex: 0.75, 0.125, 0.125) permitindo observar o comportamento com operadores dominantes.

A. Resultados do Tuning

VIII. Discussão

Analisando as técnicas avaliadas nas seções anteriores, é possível avaliar que nenhuma delas levou a excelentes

TABLE II: Variação dos Parâmetros no Grid Search

Parâmetro	Valores
Tamanho da População	8, 27, 97, 337, 1176, 4096
Taxas de Reprodução, Crossover e Mutação	Combinações de 0.125, 0.25, 0.375, 0.5, 0.625, 0.75

resultados. Nenhuma das técnicas conseguiu alcançar os valores ótimos das instâncias, mesmo daquelas de dimensão pequena. Aquela que obteve melhores resultados foi a Simple Random Mutation/Crossover com correção. Tal técnica é a mais elaborada dentre as que foram desenvolvidas, no entanto ainda é uma heurística simples, dependente da aleatoriedade.

Uma possível explicação para o desempenho ruim de SRC/SRM é é que são geradas muitas soluções infactíveis. Em um cenário que indivíduos pais são boas soluções, muito provavelmente cada veículo da rota está cheio. A aplicação de *BestInsertion* muito provavelmente levará à solução infactível.

Quanto aos operadores clássicos de crossover e mutação, eles claramente não são puramente aplicáveis ao problema. É difícil criar novos indivíduos válidos a partir do crossover de um ponto, fazendo do operador de mutação (permutação) a unica opção aplicável. No entanto, observou-se que isso causa muita aleatoriedade nas soluções e contribui para que ela não converja facilmente para o ótimo.

IX. Introduction

This demo file is intended to serve as a "starter file" for IEEE conference papers produced under LaTeX using IEEEtran.cls version 1.8b and later. I wish you the best of success.

mds
August 26, 2015

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

X. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.