Uma Heurística Híbrida para o Problema de Roteamento de Viaturas Policiais em Grandes Centros Urbanos*

Raphael Leardini[1], Eduardo Canellas[1], Bruno Sá[1], Wagner Santos[1,2], Yuri Frota[1], Daniel de Oliveira[1], Isabel Rosseti[1]

[1]Instituto de Computação, Universidade Federal Fluminense (IC/UFF), Brasil

{raphaelleardini, bcunha, eduardocanellas, wagnergs}@id.uff.br

{yuri, danielcmo, rosseti}@ic.uff.br

[2]Polícia Militar do Estado do Rio de Janeiro (PMERJ), Brasil

Resumo. *Neste artigo uma heurística híbrida, baseada na metaheurística Iterated Local Search, com busca local Variable Neighborhood Descent, é proposta para resolver, de maneira aproximada, um problema de otimização relacionado ao Roteamento de Viaturas Policiais em Grandes Centros Urbanos (RVP-Urb), onde o principal objetivo é diminuir o risco de áreas com alta taxa de criminalidade, reduzindo a violência nas cidades.*

Abstract. *In this paper, a hybrid heuristic, based on the Iterated Local Search metaheuristic, with local search based on Variable Neighborhood Descent, is proposed to solve an optimization problem related to Routing Police Vehicles in Large Urban Centers (RVP-Urb), where the main goal is to reduce the risk of areas with the high crime rate.*

1. Introdução

A crescente movimentação da população em busca de oportunidades de emprego, alavancada pela crise econômica nos últimos anos, têm gerado um aumento significativo dos índices criminais nos grandes centros urbanos. De acordo com o Instituto de Segurança Pública do Estado do Rio de Janeiro (ISP-RJ), os crimes de rua (*e.g.*, furto de celular, roubo e furto de veículos, *etc*.) têm seguido uma trajetória inconstante, mas ascendente, no período entre janeiro de 2010 e dezembro de 2017[1]. Devido a esse aumento da criminalidade, as autoridades da segurança pública, em seus diversos níveis governamentais, têm trabalhado para desenvolver ferramentas para criação de políticas públicas eficazes com o objetivo de reduzir a violência, usando ações que tentam se antecipar à ocorrência de alguma atividade criminal. Esse tipo de policiamento é chamado de *Policiamento Preditivo* [Alikhademi et al. 2022]. A ideia do Policiamento Preditivo é tornar a atuação da força policial clara para a população, seja pela presença de oficiais em pontos estratégicos da cidade ou por meio de patrulhas policiais. No contexto do policiamento preditivo, uma das tarefas mais complexas a serem realizadas é o Roteamento de Viaturas Policiais em Grandes Centros Urbanos (*RVP-Urb*), conforme discutido por [Saint-Guillain et al. 2021]. A complexidade do problema de *RVP-Urb* é que comumente os recursos existentes são escassos, *i.e.*, a quantidade de viaturas e policiais disponíveis é muito menor do que a necessária para cobrir uma determinada região. Assim, os

*Os autores gostariam de agradecer a FAPERJ, CAPES e CNPq por financiarem o artigo.

[1]https://www.ispgeo.rj.gov.br

comandantes de batalhão (que são regularmente responsáveis pela definição de rotas de patrulhamento) devem definir quais rotas específicas cada veículo será responsável. Esse tipo de definição de rotas acaba priorizando os chamados *Hot Spots* [Reis et al. 2006], *i.e.*, áreas da cidade onde o índice de criminalidade é mais elevado.

Dessa forma, para resolver o problema *RVP-Urb*, precisamos maximizar a cobertura de áreas com maior incidência de crimes em uma região (representada por um grafo) nas rotas geradas. A geração de tais rotas deve considerar uma série de restrições informadas pelo oficial de polícia: (i) o número de unidades policiais disponíveis (*e.g.*, viaturas da polícia, policiais, postos policiais, *etc*.), (ii) as regiões da cidade a serem consideradas (*i.e.*, zonas), (iii) as distâncias máximas da rota gerada para cada tipo de unidade, e (iv) a topologia da cidade representada como um grafo. Uma rota de patrulhamento, no contexto do *RVP-Urb*, é sempre representada por um trajeto cíclico pelas ruas da cidade, e pode ser dos seguintes tipos: (i) Tipo 1: rotas que podem cobrir mais de uma zona, mas possuem limite de distância máximo (*i.e.*, Inter-zonas). Cada Inter-zona pode possuir um conjunto de pontos de interesse (*e.g.*, escolas, bancos) que devem obrigatoriamente ser cobertos pela rota; (ii) Tipo 2: rotas que devem se localizar dentro de uma zona somente e também possuem limite de distância máximo (*i.e.*, Intra-zona). É importante ressaltar que essa organização da cidade em zonas e os tipos de rotas são utilizados pela Polícia Militar em diversos estados do Brasil. Além dos tipos de rotas citados anteriormente, as unidades fixas ainda devem ser posicionadas (*e.g.*, policiais que se encontram em estações de policiamento). O problema de roteamento de veículos (mesmo sem considerarmos as características do problema *RVP-Urb*) é um conhecido problema NP-Completo [Lenstra and Rinnooy Kan 1981]. Tal problema, mesmo em pequena/média escala não pode ser solucionado com modelos exatos, o que faz com que soluções heurísticas se façam necessárias. Esse artigo apresenta uma heurística híbrida, denominada *ILS*$_{Urb}$, para o problema *RVP-Urb*. Avaliamos a heurística desenvolvida com um cenário real da cidade de São Paulo (os dados da Secretaria de Segurança do Rio de Janeiro não são públicos na granularidade necessária para a abordagem proposta) e os resultados reforçam a importância de soluções automatizadas para o problema *RVP-Urb*.

2. Referencial Teórico

O grafo de crimes desempenha um papel fundamental na solução do problema *RVP-Urb*. Com base nas informações fornecidas por este grafo, a heurística proposta será capaz de definir soluções de boa qualidade em relação ao ótimo. Assim, nesta seção, fornecemos mais detalhes sobre o grafo de crimes. Seja $G^c = (V, E, Q)$ um grafo, denotado como grafo de crimes, com conjunto de vértices V de ordem $n = |V|$ representando esquinas ou divisões de ruas e conjunto de arestas $E = E^1 \cup E^2$ de tamanho $m = |E|$ representando (segmentos de) ruas, onde E^1 e E^2 representam o conjunto de vias de mão única e dupla, respectivamente. Podemos também definir $Q = \{Q_1, Q_2, ..., Q_q\}$ como a partição de V em componentes disjuntos (zonas), *i.e.*, $Q_1 \cup Q_2 \cup .. \cup Q_q = V$ e $Q_u \cap Q_o = \emptyset$, para cada $u, o = 1...q$ com $u \neq o$. Também definimos como $Q[i]$ o índice da componente do vértice $i \in V$: *i.e.*, $i \in Q_{Q[i]}$. Além disso, para cada aresta $\{i, j\} \in E$, definimos f_{ij} e l_{ij} como o fator de crime e o comprimento da aresta, respectivamente. Vale ressaltar que o fator crime f_{ij} pode ser calculado com base no número de ocorrências c_x^{ij} de cada tipo de crime $x = 1...d$ na aresta $ij \in E$ (assumindo que temos d tipos de crimes) e o peso associado a cada tipo α_x (o peso define a importância de um tipo específico de crime na análise) no conjunto de dados, *i.e.*, $f_{ij} = \sum_{x=1}^{d} c_x^{ij} \alpha_x$. Na Figura 1(c), apresentamos um exemplo de um grafo de crime com nove vértices e três zonas (Q_0, Q_1 e Q_2) onde as ruas de mão única são representadas como linhas simples, as ruas de mão dupla

são representados como linhas tracejadas, e os rótulos das arestas (f, l) representam o fator de crime e o comprimento da aresta, respectivamente.

3. Formulação Matemática do Problema *RVP-Urb*

A partir do grafo de criminalidade $G^c = (V, E, Q)$, definido anteriormente na Seção 2 e fornecido como entrada, somos capazes de construir um grafo direcionado $\overrightarrow{G} = (V, A, Q)$ onde, para cada aresta $\{i, j\} \in E^2$, dois arcos (i, j) e (j, i) são criados em A. Similarmente, para cada aresta $\{i, j\} \in E^1$ um arco único (i, j) é adicionado em A representando a sua direção. A *vizinhança de saída* de $i \in V$, denotada por $N^+(i)$, pode ser definida como o conjunto de todos os vértices j alcançados por i em $\overrightarrow{G}$; isto é, $N^+(i) = \{j|(i, j) \in A\}$. Além disso, a *vizinhança de entrada* de $i \in V$, denotada por $N^-(i)$, pode ser definida como o conjunto de todos os vértices j que chegam em i em $\overrightarrow{G}$; isso é, $N^-(i) = \{j|(j, i) \in A\}$.

Também podemos definir $N^+_Q(i) = \{j \in N^+(i) : Q[i] = Q[j]\}$ como a *vizinhança de saída de mesma zona* de um vértice $i \in V$ (*i.e.*, os vértices de saída de i que estão na mesma zona) e $N^-_Q(i) = \{j \in N^-(i) : Q[i] = Q[j]\}$ como a *vizinhança de entrada de mesma zona* de vértice $i \in V$ (*i.e.*, os vértices na vizinhança de entrada de i que estão na mesma zona). Dado um subconjunto de vértices $S \subseteq V$, denotamos como $A[S]$ o subconjunto de arcos incluídos em $\overrightarrow{G} = (V, A, Q)$ induzidos por S. Portanto, podemos definir $A_Q = A[Q_1] \cup A[Q_2] \cup ... \cup A[Q_q]$ como o conjunto de arcos de *intra-zona*, *i.e.*, arcos com as duas extremidades na mesma zona (componente).

Uma vez definidos os arcos e a estrutura de $\overrightarrow{G}$, somos capazes de definir R como sendo a família de todas as rotas viáveis de $\overrightarrow{G}$, onde uma rota do grafo $C \in R$ é definida pelo seu conjunto de arcos, *i.e.*, $C \subseteq A$. Similarmente, é definido $R_u \subseteq A[Q_u]$ como a família de todas as rotas de intra-zona viáveis de zona $u = 1, ..., q$. Na Figura 1(a) é ilustrado o grafo direcional gerado a partir da Figura 1(c), enquanto que a Figura 1(b) exemplifica uma solução do problema *RVP-Urb*, com uma rota inter-zona ❶ → ⓪ → ❻ → ❷ → ❶ (em azul) passando através das zonas Q_0, Q_1 e Q_2, uma rota de intra-zona ❸ → ❷ → ❹ → ❸ (em vermelho) na zona Q_1, e uma unidade fixa no vértice 8, na zona Q_2.

Na Figura 1(c) também podemos ver as arestas cobertas no grafo de criminalidade pela solução apresentada na Figura 1(b). A rota inter-zona, a rota intra-zona e a unidade fixa cobrem 9, 5 e 4 *Unidades de Crime* (UC) no grafo, respectivamente. Uma UC é equivalente à um valor unitário do fator de crime. Se uma rota contem três arcos no grafo com fatores de crime de valor 1, 2 e 3, a mesma cobre 6 UCs. A solução apresentada na Figura 1(c) apresenta uma cobertura de crime de 18 UCs. É importante ressaltar que todas as arestas que possuem uma unidade fixa em alguma de suas extremidades estão cobertas. Da mesma forma, uma rota não precisa usar todos os arcos de uma aresta de mão-dupla para cobri-la. Dessa forma, o problema *RVP-Urb* pode ser formulado como um problema de programação linear inteira, conforme detalhado a seguir. A Tabela 1 resume os dados utilizados na formulação matemática proposta e as variáveis de decisão do modelo.

A função objetivo (1) maximiza o fator de crime das arestas cobertas pelas unidades móveis e fixas.

$$\max \sum_{\{i,j\} \in E} f_{ij} \cdot c_{ij} \tag{1}$$

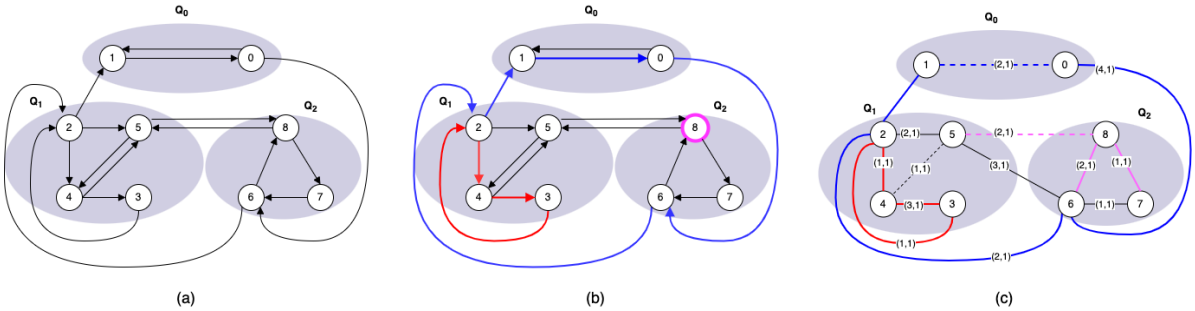

Figura 1. (a) Grafo direcionado (b) Exemplo de solução (c) Arestas cobertas no grafo

Tabela 1. Lista de Dados, Conjuntos e Variáveis.

Dados e Conjuntos	**Descrição**
P	Conjunto de unidades fixas (*i.e.*, guardas).
K, K_u	Conjunto de unidades móveis (veículos, motos, *etc*) usados nas rotas inter-zona (K) e nas rotas de intra-zona (K_u) na zona $u = 1, ..., q$. Além disso, é definido que cada unidade móvel $k \in K$ tem um conjunto de arestas prioritárias $E_k \subseteq E$ que precisam ser cobertas (*e.g.* escolas, shoppings, etc), *i.e.* a rota precisa conter estas arestas.
$\overline{K} = K \cup K_u$	Conjunto de todas as unidades móveis.
L_{max}, L^u_{max}	Comprimento máximo das rotas do grafo (L_{max}) e as rotas de intra-zona (L^u_{max}) na zona $u = 1, ..., q$.
f_{ij}	fator de crime da aresta $\{i, j\} \in E$
l_{ij}	comprimento da aresta $\{i, j\} \in E$
Variáveis	**Descrição**
x_{ijk}	Variável binária que indica se o arco $(i, j) \in A$ é atravessado por uma unidade móvel $k \in K$ ou não, em uma rota inter-zona.
x^u_{ijk}	Variável binária que indica se o arco $(i, j) \in A[Q_u]$ é atravessado por uma unidade móvel $k \in K_u$ ou não, em uma rota de intra-zona na zona $u = 1, ..., q$.
y_{ik}	Variável binária que indica se o vértice $i \in V$ é visitado por uma unidade móvel $k \in K$ ou não, em uma rota inter-zona.
y^u_{ik}	Variável binária que indica se o vértice $i \in Q_u$ é visitado por uma unidade móvel $k \in K_u$ ou não, em uma rota de intra-zona na zona $u = 1, ..., q$.
w_i	Variável binária que indica se uma unidade fixa está alocada para guardar o vértice $i \in V$ ou não.
c_{ij}	Váriavel binária que indica se a aresta $\{i, j\} \in E$ está coberta (protegida) por pelo menos uma unidade (móvel ou fixa).

As restrições (2-6) são usadas para construir as rotas inter-zona das unidades móveis K. As restrições (2-3) garantem que um vértice pertencente a uma rota inter-zona precisa ter exatamente dois arcos adjacentes (um de entrada e um de saída), enquanto as restrições (4) previnem que duas rotas do grafo diferentes usem a mesma unidade. Além disso, as restrições (5-6) denotam que toda aresta prioritária de uma rota inter-zona precisa ser coberta pelo seu percurso.

$$\sum_{j \in N^+(i)} x_{ijk} = y_{ik}, \qquad \forall i \in V, \forall k \in K \tag{2}$$

$$\sum_{j \in N^-(i)} x_{jik} = y_{ik}, \qquad \forall i \in V, \forall k \in K \tag{3}$$

$$\sum_{(i,j) \in C} x_{ijk} + x_{pqk} \leq |C|, \qquad \forall C \in R, \forall (p, q) \in A \backslash C, \forall k \in K \tag{4}$$

$$x_{ijk} = 1, \qquad \forall k \in K, \forall \{i, j\} \in (E_k \cap E^1) \tag{5}$$

$$x_{ijk} + x_{jik} = 1, \qquad \forall k \in K, \forall \{i, j\} \in (E_k \cap E^2) \tag{6}$$

As restrições (7-9) tem o mesmo objetivo o das (2-4), porém para o caso da rota de intra-zona.

$$\sum_{j \in N_Q^+(i)} x_{ijk}^u = y_{ik}^u, \qquad \forall u = 1...q, \forall i \in Q_u, \forall k \in K_u \tag{7}$$

$$\sum_{j \in N_Q^-(i)} x_{jik}^u = y_{ik}^u, \qquad \forall u = 1...q, \forall i \in Q_u, \forall k \in K_u \tag{8}$$

$$\sum_{(i,j) \in C} x_{ijk}^u + x_{pqk}^u \leq |C|, \qquad \forall u = 1...q, \forall C \in R_u, \forall (p,q) \in A[Q_u] \backslash C, \forall k \in K_u \tag{9}$$

As restrições (10) e (11) impõem um limite de distância para cada rota inter-zona e intra-zona, respectivamente, enquanto as restrições (12) limitam o número de unidades fixas no grafo.

$$\sum_{(i,j) \in A} l_{ij} \cdot x_{ijk} \leq L_{max}, \qquad \forall k \in K \tag{10}$$

$$\sum_{(i,j) \in A[Q_u]} l_{ij} \cdot x_{ijk}^u \leq L_{max}^u, \qquad \forall u = 1...q, \forall k \in K_u \tag{11}$$

$$\sum_{i \in V} w_i \leq |P| \tag{12}$$

As desigualdades (13-16) ligam os recursos móveis e fixos com as variáveis c para estabelecer quais arestas se encontram cobertas por rotas ou unidades fixas. Uma aresta em que cada extremidade pertence a uma zona diferente é considerada coberta se os seu(s) arco(s) correspondente(s) pertence(m) a uma rota ou pelo menos uma das extremidades possui uma unidade fixa (13-14). Similarmente, uma aresta de zona (*i.e.*, uma aresta com as duas extremidades na mesma zona) é considerada coberta e os seu(s) arco(s) correspondente(s) pertence(m) a uma rota inter-zona ou intra-zona, ou pelo menos uma das suas extremidades possui uma unidade fixa (15-16).

$$c_{ij} \leq \sum_{k \in K} x_{ijk} + w_i + w_j \quad \forall \{i,j\} \in E^1, \text{ tal que } Q[i] \neq Q[j] \tag{13}$$

$$c_{ij} \leq \sum_{k \in K} (x_{ijk} + x_{jik}) + w_i + w_j \quad \forall \{i,j\} \in E^2, \text{ tal que } Q[i] \neq Q[j] \tag{14}$$

$$c_{ij} \leq \sum_{k \in K} x_{ijk} + \sum_{k \in K_u} x_{ijk}^{Q[i]} + w_i + w_j \quad \forall \{i,j\} \in E^1, \text{ tal que } Q[i] = Q[j] \tag{15}$$

$$c_{ij} \leq \sum_{k \in K} (x_{ijk} + x_{jik}) + \sum_{k \in K_u} (x_{ijk}^{Q[i]} + x_{jik}^{Q[i]}) + w_i + w_j \quad \forall \{i,j\} \in E^2, \text{ tal que } Q[i] = Q[j] \tag{16}$$

Finalmente, as restrições (17) e (18) não são necessárias para o modelo, mas a sua inclusão ajuda a eliminar soluções simétricas. Essas restrições utilizam ordenação lexicográfica para atribuir um número único a cada possível conjunto de vértices em uma rota e ordenar essa rota de acordo com o número atribuído. Para garantir a unicidade, é usada a base de 2 como coeficiente, conforme provado por [Jans 2009]

$$\sum_{i \in V} 2^{(n-i)} y_{ik} \leq \sum_{i \in V} 2^{(n-i)} y_{i(k+1)},$$
$$\forall k = 1...|K| - 1 \tag{17}$$
$$\sum_{i \in Q_u} 2^{(n-i)} y^u_{ik} \leq \sum_{i \in Q_u} 2^{(n-i)} y^u_{i(k+1)},$$
$$\forall u = 1...q, \forall k = 1...|K_u| - 1 \tag{18}$$

4. A Heurística ILS_{Urb}

Nessa seção apresentamos a heurística híbrida proposta, baseada na combinação da metaheurística *Iterated Local Search* (ILS) [Lourenço et al. 2001] com a busca local *Variable Neighborhood Descent* (VND) [Duarte et al. 2018].

4.1. Metaheurísticas Utilizadas

A metaheurística ILS corresponde ao uso iterativo de uma heurística de Busca Local (BL) sobre versões modificadas de soluções prévias encontradas. A principal ideia é que, a cada iteração, ao invés de se gerar uma solução aleatória nova como ponto de partida, seja aplicada uma BL sobre uma solução anterior com modificações em algumas de suas partes, por meio da aplicação de um mecanismo de perturbação [Lourenço et al. 2001]. A busca local VND consiste na aplicação de uma BL exaustiva sobre a solução, repetidamente, ascendendo a um máximo local (uma solução com o valor da função objetivo maior que todas soluções vizinhas próximas). Além disso, o mecanismo de perturbação da ILS tenta escapar de ótimos locais em que a solução pode se encontrar. A cada iteração, uma nova solução é gerada pela perturbação e usada pela heurística BL. A heurística BL é então aplicada repetidamente até que não haja mais melhorias, *i.e.*, até que seja alcançado um ótimo local (OL).

4.2. Heurística Construtiva

A heurística construtiva proposta constrói uma solução para o problema inserindo iterativamente arestas (estipulando rotas para viaturas) e vértices (posicionando policiais) até alcançar uma solução viável, respeitando as restrições do problema. Denotaremos como ROUTES as rotas estabelecidas para as unidades móveis da polícia ($\overline{K}$) e como STATIC os posicionamentos estabelecidos para as unidade fixas (P). Assim, uma solução viável é composta por todas as rotas (inter-zonas e intra-zonas) geradas no conjunto ROUTES, e os vértices escolhidos do conjunto STATIC. É importante ressaltar que para uma rota gerada ser considerada válida, ela precisa respeitar todas as restrições do seu respectivo conjunto associado (inter-zonas ou intra-zonas). Dessa forma, particionamos o conjunto de rotas ROUTES nos seguintes subconjuntos usados na heurística, a partir do menos restrito para o mais restrito: (i) $\text{INTER}_{\emptyset}$: o conjunto de rotas inter-zona $k \in K$ onde o conjunto de arestas prioritárias é vazio (*i.e.*, $E_k = \emptyset$). Ele possui apenas a restrição de distância máxima do percurso L_{max}, (ii) INTRA: o conjunto de rotas intra-zona $k \in K_u$. Além da restrição de distância, *i.e.* L^u_{max}, os percursos formados só podem passar pelos vértices da sua zona específica $u = 1...q$, e (iii) INTER_{E_k}: o conjunto de rotas inter-zona $k \in K$ onde o conjunto de arestas prioritárias não é vazio (*i.e.*, $E_k \neq \emptyset$). Note que se qualquer aresta prioritária nas rotas de INTER_{E_k} não for coberta, então a solução inteira é considerada inviável para aquela instância do *RVP-Urb*.

O método construtivo é ilustrado no Algoritmo 1. Inicialmente, são criados conjuntos vazios para todos os conjuntos de ROUTES ($\text{INTER}_{\emptyset}$, INTRA e INTER_{E_k}) e STATIC. Para

cada conjunto de ROUTES, a partir do mais restrito para o menos restrito, é chamado o seu método de inserção correspondente, encontrando uma rota para cada unidade móvel disponível daquele tipo de conjunto. Os conjuntos $\text{INTER}_{\emptyset}$ e INTRA não precisam usar todos os recursos disponíveis, de forma que se nenhuma rota for encontrada para aquele conjunto, o algoritmo segue para o método de inserção do próximo conjunto. A solução inicial gerada é formada pelo conjunto de todas as rotas de todos os conjuntos classificados como ROUTES e o conjunto de vértices de STATIC.

Algorithm 1 Contruction(G)

1: $\mathit{INTER}_{E_k} \leftarrow \{\}$, $\mathit{INTRA} \leftarrow \{\}$, $\mathit{INTER}_{\emptyset} \leftarrow \{\}$, $\mathit{STATIC} \leftarrow \{\}$
2: **for** each $k \in K$ such $E_k \neq \emptyset$ **do**
3: *route* $\leftarrow$ FindINTER$_{E_k}$Route(G, E_k)
4: **if** *route* is feasible **then**
5: $\mathit{INTER}_{E_k} \leftarrow \mathit{route} \cup \mathit{INTER}_{E_k}$
6: **for** each u in $1, ..., q$ **do**
7: **for** each $k \in K_u$ **do**
8: *route* $\leftarrow$ FindINTRARoute(G, u)
9: **if** *route* is feasible and has at least one uncovered edge **then**
10: $\mathit{INTRA} \leftarrow \mathit{route} \cup \mathit{INTRA}$
11: **for** each $k \in K$ *such* $E_k = \emptyset$ **do**
12: *route* $\leftarrow$ FindINTER$_{\emptyset}$Route(G)
13: **if** *route* is feasible and has at least one uncovered edge **then**
14: $\mathit{INTER}_{\emptyset} \leftarrow \mathit{route} \cup \mathit{INTER}_{\emptyset}$
15: $\mathit{STATIC} \leftarrow$ FindSTATICVertices(G, P)
16: $s \leftarrow \mathit{INTER}_{E_k} \cup \mathit{INTRA} \cup \mathit{INTER}_{\emptyset} \cup \mathit{STATIC}$
17: **return** s

Para as rotas $k \in K$ do conjunto INTER_{E_k}, foi implementado o método FindINTER$_{E_k}$Route. Iniciando com uma aresta prioritária aleatória de E_k, é invocado um algoritmo de *backtracking*, de forma a maximizar as chances de se encontrar uma rota viável que contenha as demais arestas prioritárias de E_k. Neste procedimento, para cada aresta prioritária $\{i, j\} \in E_k$ que ainda precisa ser coberta pela rota, tenta-se encontrar o menor caminho, do último vértice adicionado a rota até a aresta prioritária (*i.e.* até i ou j), usando o algoritmo de Dijkstra. Se o caminho existir, então a aresta prioritária e o caminho encontrado são inseridos na rota, e uma nova chamada ao procedimento recursivo é realizada. Se não existir um caminho, ou o caminho encontrado ultrapassar L_{max}, o procedimento desfaz a última inserção, não cobrindo mais o último caminho adicionado à rota. Dessa forma, o procedimento tenta uma combinação nova com outra aresta. Esse método continua executando até que todas as arestas prioritárias sejam adicionadas à rota, e a rota possa alcançar o vértice de início. Este método é custoso em termos de tempo de processamento, gastando mais da metade do tempo de execução da heurística ILS_{Urb} para instâncias com rotas $k \in K$ onde $|E_k| \geq 10$. Por essa razão, escolhemos estratégias focadas na fase de busca local, como o ILS, ao invés de abordagens focadas na fase construtiva, como o GRASP. Já para as rotas dos conjuntos $\text{INTER}_{\emptyset}$ e INTRA, desenvolvemos métodos de inserção com início aleatório e uma memória para caminhos inválidos previamente encontrados durante a fase construtiva. Os dois métodos começam selecionando aleatoriamente um vértice inicial do grafo como o ponto inicial de uma nova rota. A partir do último vértice inserido, é escolhido um novo vértice vizinho como próximo ponto de visita. O vértice é escolhido baseado no maior fator de crime dentre os vizinhos atuais, caso o caminho resultante não esteja presente na memória de caminhos inválidos. Esse processo é repetido até que o vértice inicial seja encontrado novamente, e o ciclo seja fechado. Se o caminho, em qualquer ponto da iteração, ultrapassar L_{max} ou não possuir mais vértices vizinhos válidos, esse caminho é então adicionado a memória de

caminhos inválidos. Se esta memória estiver cheia, ela e a rota em construção são descartadas, e um novo vértice inicial é escolhido. Os procedimentos são executados até que uma rota válida seja encontrada ou o número de tentativas exceda um *threshold*. A rota válida que for encontrada é coberta e adicionada ao seu conjunto correspondente. A principal diferença entre os dois métodos se encontra na presença de uma restrição de zona (para INTRA) e a distância máxima que as rotas podem possuir. Finalmente, o último procedimento de inserção, usado para o conjunto das STATICs percorre o grafo completo, selecionando os vértices que podem cobrir o maior fator de crime, inserindo-os no conjunto STATIC.

4.3. VND como Busca Local

Na heurística de BL, foi implementado o Algoritmo 2. Ele iterativamente tenta realizar movimentos para melhorar a cobertura total de crime da solução. Em qualquer ponto da execução, se um dos conjuntos de ROUTES e STATIC encontrar uma melhoria, então o processo inteiro é reiniciado para o primeiro movimento. As iterações ocorrem até que nenhuma nova melhoria seja alcançada. Cada conjunto de ROUTES possui dois tipos de movimentos de primeira-melhoria (*i.e.*, o método realiza o primeiro movimento encontrado que melhore o custo da solução): (i) SP-1: para cada rota do conjunto atual ($\text{INTER}_{\emptyset}$, INTER_{E_k} e INTRA) é feita uma tentativa de melhoria selecionando uma aresta aleatória a ser retirada da rota (exceto para o caso da rota ser do conjunto INTER_{E_k}, *i.e.*, a aresta selecionada não pode ser uma aresta prioritária), considerando uma extremidade da aresta como vértice origem e a outra como vértice de destino. A seguir é chamado um método para encontrar um caminho existente entre esses dois pontos, fazendo uma escolha aleatória entre o método de Dijkstra ou o método de busca em largura. Se um caminho alternativo existir, e possuir um fator de crime maior que o da aresta selecionada para exclusão, então a aresta original é descoberta e removida da rota, e o caminho novo é coberto e adicionado à rota. Este algoritmo de melhoria executa até que todas as arestas tenham sido selecionadas e nenhuma melhoria tenha sido encontrada; e (ii) SP-2: a única diferença para o movimento SP-1 é que, ao invés de uma aresta, ele tenta substituir duas arestas em sequência na rota. Da mesma forma, o sub-caminho formado por essas duas arestas não pode possuir nenhuma aresta prioritária, se a rota pertencer ao conjunto INTER_{E_k}. Já para o conjunto STATIC só é realizado um movimento, denominado de MOV_v, de melhor-melhoria (ou seja, o método realiza o movimento que possuir o melhor acréscimo ao valor da função objetivo). Este movimento de melhoria consiste apenas em substituir os vértices atuais do conjunto STATIC por vértices com que consigam cobrir arestas com fatores de crime maiores não cobertas ainda, caso eles existam.

4.4. Algoritmo de Perturbação

O algoritmo de perturbação implementado consiste de uma série de movimentos aleatórios, sem nenhuma preocupação em melhorar o custo da solução. O objetivo deste método é remover a solução de um OL, provendo um novo ponto de partida para a BL. O objetivo é que a BL não consiga (ou seja muito difícil) retornar para o mesmo OL. São definidos os números de rotas (de qualquer um dos conjuntos) que serão modificadas e o número de modificações que serão realizadas para cada rota. Ambos os valores são baseados na iteração atual do ILS. Para o número fixado de rotas, é selecionado um conjunto aleatório e uma rota aleatória deste conjunto. A seguir, uma decisão aleatória é tomada: (i) destruir a rota e construir uma nova, ou (ii) realizar as modificações aleatórias na rota. Se a segunda opção for a escolhida, um método baseado nos movimentos SP-1 e SP-2 é invocado. A principal diferença deste algoritmo para os movimentos SP-1 e SP-2 originais é que ele não realiza somente movimentos de melhoria. No final da perturbação, os vértices do conjunto STATIC são realocados.

Algorithm 2 LocalSearchVND(G, s)

improve ← True; $INTER_{E_k}$, *INTRA*, $INTER_{\emptyset}$, *STATIC* ← s
while *improve* is True **do**
  improve ← False
  $INTER_{E_k}$* ← SP-1(*G, INTER*$_{E_k}$) ▷ f( ) represents the total criminal factor
  if f($INTER_{E_k}$*) > f($INTER_{E_k}$) **then**
    improve ← True; $INTER_{E_k}$ ← $INTER_{E_k}$*; Continue
  *INTRA** ← SP-1(*G, INTRA*)
  if f(*INTRA**) > f(*INTRA*) **then**
    improve ← True; *INTRA* ← *INTRA**; Continue
  $INTER_{\emptyset}$* ← SP-1(*G,* $INTER_{\emptyset}$)
  if f($INTER_{\emptyset}$*) > f($INTER_{\emptyset}$) **then**
    improve ← True; $INTER_{\emptyset}$ ← $INTER_{\emptyset}$*; Continue
  *STATIC** ← MOV$_v$(*G, STATIC*)
  if f(*STATIC**) > f(*STATIC*) **then**
    improve ← True; *STATIC* ← *STATIC**; Continue
  $INTER_{E_k}$* ← SP-2(*G,* $INTER_{E_k}$)
  if f($INTER_{E_k}$*) > f($INTER_{E_k}$) **then**
    improve ← True; $INTER_{E_k}$ ← $INTER_{E_k}$*; Continue
  *INTRA** ← SP-2(*G, INTRA*)
  if f(*INTRA**) > f(*INTRA*) **then**
    improve ← True; *INTRA* ← *INTRA**; Continue
  $INTER_{\emptyset}$* ← SP-2(*G,* $INTER_{\emptyset}$)
  if f($INTER_{\emptyset}$*) > f($INTER_{\emptyset}$) **then**
    improve ← True; $INTER_{\emptyset}$ ← $INTER_{\emptyset}$*; Continue
*s** ← $INTER_{E_k}$ ∪ *INTRA* ∪ $INTER_{\emptyset}$ ∪ *STATIC*
return *s**

4.5. Heurística ILS-VND para o *RVP-Urb*

O Algoritmo 3 apresenta a heurística ILS_{Urb}, que considera a utilização de ILS e VND. A solução inicial é gerada, e então é aplicada a BL, baseada em VND, para atingir o primeiro OL. A seguir, em cada iteração, a perturbação é aplicada na nova solução alcançada, seguida de uma nova aplicação da BL. O nível da perturbação cresce progressivamente até que um novo OL seja encontrado. Quando um novo OL é obtido, a pertubação é reiniciada para a versão mais branda. O algoritmo encerra sua execução quando a perturbação atinge o seu máximo sem que nenhuma melhoria para o OL ocorra.

Algorithm 3 ILS_{Urb}(*G*, *maxIterILS*)

s ← Contruction(*G*); *s* ← LocalSearchVND(*G, s*);
*s** ← *s*; *iterILS* ← *0*
while $iterILS < maxIterILS$ **do**
  s ← Perturb(*G, s, iterILS*); *s* ← LocalSearchVND(*G, s*);
  iterILS ← *iterILS* + *1*;
  if f(*s**) < f(*s*) **then**
    *s** ← *s*; *iterILS* ← *0*
return *s**

5. Avaliação Experimental

Nessa seção são apresentados dois experimentos computacionais realizados e os resultados obtidos usando a abordagem proposta na Seção 4. Os algoritmos que compõem a heurística foram implementados em C++ (8.1.0), e executados em uma máquina CPU Intel(R) Core(TM) i5-7200U CPU 2.50 GHz e 8 GB de RAM. Os dados usados para os experimentos são uma integração dos dados do *OpenStreetMap* (informação das vias) e do portal de dados abertos da Secretaria de Segurança Pública de São Paulo (informação sobre a taxa de criminalidade)[2].

[2] `http://www.ssp.sp.gov.br/transparenciassp/Consulta.aspx`

Esses dados são carregados em um banco de dados PostgreSQL. Mais informações sobre o *dataset* utilizado podem ser encontradas em [Sá et al. 2021]. O primeiro experimento considerou instâncias de tamanho menor (pequenas áreas dentro de bairros, *e.g.*, quatro quarteirões em torno do MASP) para que fosse possível comparar o resultado exato com o resultado obtido pela heurística proposta. A Tabela 2 apresenta o resultado ótimo encontrado pelo modelo e o resultado da heurística (melhor solução e média dentre 10 execuções), tanto em termos do resultado da função objetivo, quanto em relação ao tempo necessário para obtenção das rotas. A partir dos dados apresentados na Tabela 2, é possível notar que a heurística demonstra bons resultados, encontrando 7 das 12 soluções ótimas geradas. No geral, a qualidade das soluções é alta uma vez que o *gap* médio (a diferença percentual da média das soluções encontradas pela heurística entre as soluções ótimas do modelo) é sempre inferior a 1%. Em relação ao tempo de execução, para a maioria das instâncias o tempo da heurística se manteve próximo ao do modelo, com exceção das instâncias "mini1-64"e "mini2-75", nas quais o modelo apresentou um tempo de execução superior a 5 segundos, enquanto a heurística se manteve abaixo de 0,3 segundos. Além disso, o modelo foi capaz de encontrar soluções ótimas em instâncias com até 75 vértices (o número de vértices de cada instância está presente em seu nome).

Tabela 2. Comparação entre os resultados exatos e os resultados da heurística

Instância	Ótimo	Melhor$_{ILS_{Urb}}$	Gap Melhor	Média$_{ILS_{Urb}}$	Gap Médio	T$_{Modelo}$(s)	T$_{ILS_{Urb}}$(s)
cjn1-18	126,74	123,85	-0,023	123,85	-0,023	0,037	0,042
cjn1-24	169,47	**169,47**	0,000	165,70	-0,022	0,081	0,059
cjn1-33	270,96	269,96	-0,004	264,72	-0,023	0,190	0,271
cjn1-40	335,52	278,43	-0,170	275,94	-0,178	0,243	0,124
cjn1-46	398,25	385,20	-0,033	382,23	-0,040	0,740	0,296
cjn2-18	120,77	**120,77**	0,000	120,77	0,000	0,008	0,033
cjn2-25	211,84	**211,84**	0,000	208,64	-0,015	0,146	0,085
cjn2-35	305,36	**305,36**	0,000	304,16	-0,004	0,057	0,155
cjn2-50	408,41	**408,41**	0,000	399,41	-0,022	0,284	0,219
mini1-64	390,97	357,23	-0,087	357,23	-0,087	13,390	0,260
mini2-75	399,94	**399,94**	0,000	388,97	-0,027	5,680	0,193
mini3-20	108,48	**108,48**	0,000	100,71	-0,072	0,029	0,048

Já o segundo experimento considerou instâncias de maior escala que não possuíam resultado ótimo conhecido, *i.e.*, instâncias em que o modelo não é capaz de gerar resultados viáveis em tempo hábil. As instâncias utilizadas nesse experimento englobam bairros inteiros, *e.g.*, Brás, Campo Belo, Morumbi, Itaim Paulista, *etc*. A Tabela 3 apresenta o valor da função objetivo e o tempo de execução da heurística para cada instância. Pode-se observar na Tabela 3 que o tempo de execução cresce devido ao aumento do tamanho das instâncias (o número de vértices), *e.g.*, a instância "bosque-da-saude-405"(a menor instância da Tabela 3) apresenta o menor tempo de execução (de aproximadamente 0.087 segundos), enquanto a instância "sacoma-3745"(a maior instância da Tabela 3) possui o maior tempo de execução (de aproximadamente 104 segundos). Vale ressaltar que os resultados apresentados na Tabela 3 não possuem comparativo com o modelo, apenas as soluções obtidas pela heurística. Isto acontece pelo fato de não ser possível computar as suas soluções ótimas (por limitação tecnológica das máquinas utilizadas para os testes), e devido ao tamanho das instâncias e da complexidade do problema *RVP-Urb*. Os arquivos produzidos nos experimentos podem ser obtidos em `https://osf.io/y3j27/`.

6. Trabalhos Relacionados

Alguns trabalhos na literatura tratam do problema de roteamento de viaturas policiais. [Chawathe 2007] modela uma rede de ruas usando uma representação baseada em grafos,

Tabela 3. Resultados para instâncias de bairros

Instância	Melhor$_{ILS_{Urb}}$	Média$_{ILS_{Urb}}$	T$_{ILS_{Urb}}$(s)	Instância	Melhor$_{ILS_{Urb}}$	Média$_{ILS_{Urb}}$	T$_{ILS_{Urb}}$(s)
bosque-da-saude-405	645,40	631,57	0,087	favela-dos-anjos-922	2010,82	1922,07	2,703
fazenda-itaim-864	886,14	886,14	0,361	jardim-japao-879	2529,98	2485,23	2,215
jardim-primavera-682	1106,28	1074,27	0,424	mirandopolis-509	598,16	586,74	0,142
planalto-paulista-567	994,23	962,60	0,612	vila-guarani-625	1189,16	1159,60	0,324
vila-renato-645	663,66	662,65	0,263	alto-de-santana-1152	1253,94	1232,54	1,002
belenzinho-1548	2201,91	2067,07	4,209	bom-retiro-1542	1498,33	1388,70	4,668
bras-1674	2241,53	2078,45	9,189	cidade-ipava-1608	2937,02	2764,01	6,495
jardim-aricanduva-1430	3440,90	3246,16	3,065	jardim-maristela-1372	3668,07	3549,50	7,558
pompeia-1617	2865,16	2699,09	9,611	rio-pequeno-1779	3421,86	3301,80	9,403
vila-amelia-1112	1740,10	1606,32	3,334	vila-antonio-1912	5538,27	5249,20	12,244
vila-leonor-1130	1861,08	1756,68	1,230	vila-leopoldina-1258	2743,53	2596,59	2,499
vila-progresso-1988	4961,00	4841,09	23,221	americanopolis-2897	8336,42	8119,13	38,019
baixa-grande-2223	5223,87	5075,37	15,109	campo-belo-2515	8114,80	7774,61	38,897
campo-limpo-2488	7352,29	6996,18	26,582	interlagos-2318	4220,88	4070,60	23,487
jardim-botucatu-2244	5002,92	4809,81	20,668	jardim-campinas-2485	5320,40	5087,91	24,564
jardim-paulista-2408	1747,67	1663,55	9,142	jardim-tupi-2745	3927,32	3747,76	17,033
mooca-2572	7137,80	6599,82	36,846	morumbi-2942	4627,24	4279,62	29,515
parque-cisper-2006	2537,41	2438,27	5,601	sossego-2463	4238,18	4060,44	25,312
sumare-2250	3377,24	3141,53	17,192	cidade-tiradentes-3562	6456,89	6263,75	34,814
itaim-paulista-3209	9545,04	8833,24	64,788	jardim-tres-marias-3282	12805,70	12079,60	61,397
limoeiro-3472	9639,66	9176,26	60,360	sacoma-3745	12745,90	12264,10	103,881
vila-formosa-3460	7029,78	6581,77	60,500	vila-matilde-3454	9708,57	9318,87	75,710
vila-santa-teresa-3222	11127,80	10558,70	74,798	vila-solange-3372	11319,30	10952,60	54,207
aeroporto1-1301	6193,76	5958,76	30,183	perdizes1-1793	5712,98	5554,95	33,755

chamada de *street-graph*. No grafo proposto, os vértices representam interseções e as arestas representam segmentos de ruas. Vale ressaltar que no grafo proposto, os pesos das arestas representam o benefício de patrulhar este segmento, e a heurística desenvolvida pelos autores considera tal benefício. Embora o conceito de representar o benefício de patrulhamento de uma área específica do grafo seja interessante, essa métrica é de difícil definição por parte das autoridades policiais. [Chen et al. 2015] propõem uma abordagem para definir rotas de patrulhamento usando a técnica de otimização de colônia de formigas. Embora também modelem as ruas da cidade como um grafo, eles consideram apenas um subconjunto de tipos de crimes (furto de celular e carro), já que o foco é o patrulhamento realizado a pé. A abordagem proposta por [Melo et al. 2005] utiliza o conceito de agente da sociedade para simular o ambiente e propor uma rota de patrulhamento. Diferentemente das abordagens anteriores, os autores modelam o ambiente como um grupo de agentes independentes que representam criminosos, policiais e lojas. As rotas são definidas como um conjunto de pontos que um policial deve percorrer durante um determinado período. Outras abordagens focam no problema de roteamento dinâmico de veículos e consideram mudanças no ambiente para ajustar a rota produzida em tempo de execução. [Bertsimas et al. 1991] propõem uma abordagem que implementa o problema do caixeiro viajante dinâmico para o cenário de roteamento policial. Por sua vez, [Gendreau et al. 2006] analisa o problema de realocação de cobertura máxima esperada para serviços de emergência (que também considera viaturas policiais) e também [Shen et al. 2018] focam em estratégias de otimização para os veículos de emergência, especialmente veículos policiais e ambulâncias.

7. Conclusão e Trabalhos Futuros

Neste artigo apresentamos uma heurística híbrida para resolução, em tempo polinomial, do problema *RVP-Urb* de roteamento de viaturas policiais em grandes centros urbanos. O objetivo da heurística proposta é maximizar a cobertura de crimes utilizando os recursos disponíveis pela polícia de maneira eficiente. Foram realizados experimentos com diferentes

heurísticas construtivas e de busca local, diferentes versões de perturbação e de aleatoriedade de movimentos, consumindo dados reais da Polícia Militar do Estado de São Paulo. É interessante ressaltar que a heurística proposta conseguiu atingir resultados satisfatórios (dentro dos objetivos propostos) quando comparados aos resultados ótimos obtidos com o modelo, onde a heurística, na maioria das instâncias, chegou ao melhor resultado possível, e forneceu soluções aproximadas para instâncias as quais o modelo não conseguiu executar (seja pelo tempo de execução longo, ou pela impossibilidade na arquitetura, limite de memória, *etc*). Como trabalhos futuros, é planejada a integração da heurística proposta com técnicas de mineração de dados, na tentativa de encontrar padrões entre subtrechos de rotas geradas. Além disso, a implementação de outras heurísticas, baseadas em metaheurísticas já conhecidas, com foco em busca local, como *Simulated Annealing* e a Busca Tabu, para realização de comparações, a fim de verificar a eficiência do ILS_{Urb}.

Referências

Alikhademi, K., Drobina, E., Prioleau, D., Richardson, B., Purves, D., and Gilbert, J. E. (2022). A review of predictive policing from the perspective of fairness. *Artif. Intell. Law*, 30:1–17.

Bertsimas, D., van Ryzin, G., and Management, S. (1991). A stochastic and dynamic vehicle routing problem in the euclidean plane. *Operations Research*, 39:601–615.

Chawathe, S. S. (2007). Organizing hot-spot police patrol routes. In *2007 IEEE Intelligence and Security Informatics*, pages 79–86.

Chen, H., Cheng, T., and Wise, S. (2015). Designing daily patrol routes for policing based on ant colony algorithm. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-4/W2:103–109.

Duarte, A., Sánchez-Oro, J., Mladenović, N., and Todosijević, R. (2018). Variable neighborhood descent. In Martí, R., Pardalos, P. M., and Resende, M. G. C., editors, *Handbook of Heuristics*, pages 341–367. Springer.

Gendreau, M., Laporte, G., and Semet, F. (2006). The maximal expected coverage relocation problem for emergency vehicles. *J. Oper. Res. Soc.*, 57:22–28.

Jans, R. (2009). Solving lot-sizing problems on parallel identical machines using symmetry-breaking constraints. *INFORMS J. on Computing*, 21:123–136.

Lenstra, J. K. and Rinnooy Kan, A. H. G. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11:221–227.

Lourenço, H., Martin, O., and Stützle, T. (2001). A beginner's introduction to iterated local search. In *4th Metaheuristics International Conference*, pages 1–11, Porto, Portugal.

Melo, A., Belchior, M., and Furtado, V. (2005). Analyzing police patrol routes by simulating the physical reorganization of agents. In Sichman, J. S. and Antunes, L., editors, *MABS 2005*, pages 99–114.

Reis, D., Melo, A., Coelho, A. L. V., and Furtado, V. (2006). Towards optimal police patrol routes with genetic algorithms. In *IEEE ISI*, pages 485–491.

Saint-Guillain, M., Paquay, C., and Limbourg, S. (2021). Time-dependent stochastic vehicle routing problem with random requests: Application to online police patrol management in brussels. *Eur. J. Oper. Res.*, 292:869–885.

Shen, Y., Lee, J., Jeong, H., Jeong, J., Lee, E., and Du, D. H. C. (2018). SAINT+: Self-adaptive interactive navigation tool+ for emergency service delivery optimization. *IEEE Transactions on Intelligent Transportation Systems*, 19:1038–1053.

Sá, B., Muller, G., Banni, M., Santos, W., Lage, M., Rosseti, I., Frota, Y., and de Oliveira, D. (2021). Polroute-DS: um dataset de dados criminais para geração de rotas de patrulhamento policial. In *Anais do III Dataset Showcase Workshop*, pages 117–127, Porto Alegre, RS, Brasil.