



Optimization of Public Transport Networks by Considering Alternative Positions for Network Stations

Ingo P. P. Montarroyos , and Danilo R. B. Araújo 

Abstract—Nowadays, the planning of metropolitan areas considers improving the quality of life of inhabitants and urban mobility is one of the main concerns. Studies point out that investments in public transportation and other modes are aimed at the overall improvement of mobility. However, there is a gap in proper tools for optimizing public transport networks. In fact, network optimization is an NP-Hard problem and there are usually many conflicting objectives that need to be optimized simultaneously. This paper proposes the use of many-objective evolutionary algorithms to address the problem of public transport networks optimization, focusing on metropolitan bus lines. The proposal consists in optimizing the position of bus stops and consequently obtaining new routes that pass through these stops in order to minimize the average travel time, the time spent between origin / destination and the variance of distance between the stops. To evaluate our proposal, a simulator was used to simulate the behavior of different passenger profiles in an urban area and the results were compared between the lines obtained by the optimization process and existing bus lines in the city of São Paulo. According to our results, optimized bus routes have mean travel time 22% less than the existing route and the time spent between origin/destination has decreased up to 18%.

Link to graphical and video abstracts, and to code: <https://latamt.ieeer9.org/index.php/transactions/article/view/8626>

Index Terms—Public Transport, Public Transport Network, Bus, Optimization, Evolutionary computation.

I. INTRODUÇÃO

O crescimento saudável de grandes centros metropolitanos está amplamente conectado ao fornecimento de serviços de transportes adequados. Uma crescente população urbana requer acesso à educação, comércio, emprego e lazer. A provisão de infraestruturas de transporte, como linhas de metrô e linhas de ônibus, é importante para o desenvolvimento dos centros urbanos, uma vez que permite aos cidadãos que não possuem veículos próprios a locomoção entre pontos de interesse em um menor espaço de tempo. Por outro lado, de acordo com [1], o conceito de sustentabilidade veio a ser recentemente incorporado, mesmo que em graus variantes, no que se refere a planejamento e gerenciamento urbano em diferentes regiões do Brasil. Levando em consideração a crescente preocupação com a sustentabilidade no planejamento urbano, é preciso investir em tecnologias que auxiliem e incentivem ainda mais

a utilização e implementação de meios sustentáveis. Estudos recentes mostram que a qualidade de serviço influencia a satisfação de viagem sentida e a fidelização de usuários de transporte público [2] [3].

Portanto, o planejamento de uma infraestrutura de transporte público de qualidade possibilitaria que mais cidadãos se sintam motivados a escolher viagens por meios públicos, reduzindo assim a dependência de veículos pessoais, que consequentemente também proporcionaria cidades mais habitáveis, saudáveis e neutras em carbono, como foi argumentado em [4].

Em adição, de acordo com [5], a qualidade de serviço pode influenciar a acessibilidade percebida em transportes públicos, uma vez que transportes públicos em sua essência proporcionam uma menor facilidade de uso que opções de transporte porta a porta, sendo assim importante enfatizar o foco nos atributos relacionados à facilidade percebida pelas pessoas de viver a vida que desejam com a ajuda do sistema de transporte público.

No que se refere a qualidade de infraestrutura, o planejamento vai além da escolha de materiais e tecnologias para a sua construção, o projeto eficiente da topologia física de uma rede de transporte público também é essencial para o bom funcionamento e qualidade dela. Sob o ponto de vista do usuário de uma linha de ônibus, diversas preocupações estão presentes, como a proximidade entre as paradas e a sua localização de partida e chegada, o tempo médio de viagem, e assim por diante.

Pode-se dizer que existe uma lacuna em termos de ferramentas adequadas para otimizar as redes de transporte público. A otimização de redes é um problema NP-Difícil e normalmente existem muitos objetivos conflitantes que precisam ser otimizados simultaneamente.

Ultimamente Algoritmos Evolucionários Multiobjetivo (AEMOs) têm sido amplamente usados para encontrar soluções quase ótimas para o projeto de topologias físicas [6] [7]. Banerjee e Kumar [6] propuseram um AEMO para lidar com o projeto da topologia física de redes ópticas, aplicando a proposta deles em três cenários de redes com 10, 21 e 36 terminais. Em [6], o desempenho do AEMO foi comparado com uma abordagem exaustiva. Essa comparação foi feita apenas com a rede de 10 terminais por causa da inviabilidade de se usar a abordagem exaustiva em redes maiores. Além disso, abordagens de otimização multiobjetivo que consideram danos nas camadas físicas apresentam um grande custo computacional para avaliação de uma possível solução, visto que a avaliação de desempenho geralmente é realizada por meio

I. P. P. Montarroyos, and D. R. B. Araújo are with Federal Rural University of Pernambuco, Recife, Brazil (e-mails: ingo.past@gmail.com, and danilo.araujo@ufrpe.br).

de simulações de Monte Carlo. Em 2023, Candeias *et al.* [7] propuseram uma nova metodologia de lidar com o projeto de redes ópticas considerando a posição dos nós terminais como variáveis de decisão. O método consistiu na utilização de um algoritmo de clusterização para determinar a localização inicial dos nós terminais e de um algoritmo evolucionário para muitos objetivos com busca local para modificar as localizações dos terminais durante o processo de otimização. Neste caso o algoritmo evolucionário escolhido foi o *Non-dominated Sorting Genetic Algorithm III* (NSGAIII).

A proposta de Candeias *et al.* [7] superou abordagens tradicionais e inspirou este trabalho para aplicar uma metodologia de otimização de topologias semelhante no contexto de redes de transporte público, uma vez que não foram encontradas pesquisas anteriormente publicadas com uma proposta semelhante.

Por tanto, o objetivo deste trabalho é otimizar linhas de ônibus com a finalidade de melhorar a experiência do usuário sob diversos aspectos. Nessa linha, o foco deste artigo é otimizar a posição das paradas e consequente obtenção de novas rotas que passam pelas novas paradas com o objetivo de minimizar o tempo médio de viagem, o tempo de deslocamento da origem/destino até as paradas e a variância entre as paradas. Para avaliação da proposta, é utilizado um simulador próprio que simula o comportamento de diferentes perfis de passageiros em uma área urbana e os resultados foram comparados entre linhas obtidas pelo processo de otimização e linhas de ônibus existentes na cidade de São Paulo. De acordo com nossos resultados, as rotas de ônibus otimizadas têm tempo médio de viagem 22% menor que a rota existente e o tempo gasto entre origem/destino diminuiu até 18%.

O restante deste artigo está organizado da seguinte forma. A Seção II fornece uma revisão sobre Otimização Multiobjetivo. A Seção III apresenta as ferramentas utilizadas na pesquisa. A Seção IV descreve o problema em detalhes. A Seção V apresenta e discute os resultados obtidos nos experimentos iniciais. A Seção VI apresenta as conclusões e sugestões para trabalhos futuros.

II. OTIMIZAÇÃO MULTIOBJETIVO

AEMOs são meta-heurísticas capazes de fornecer boas soluções para problemas de otimização com vários objetivos conflitantes e apresentam baixo custo computacional em comparação com abordagens exaustivas. Diversos AEMOs estão disponíveis na literatura [8] e neste trabalho é utilizado o algoritmo baseado em pontos de referência *Non-dominated Sorting Genetic Algorithm III* (NSGA-III), que é considerado por muitos como estado da arte na área, especialmente para problemas com 3 ou mais objetivos conflitantes.

A. NSGA-III

Em 2002, Deb *et al.* [9] propuseram o *Non-dominated Sorting Genetic Algorithm II* (NSGA-II). Desde 2002, vários estudos focados em problemas no mundo real demonstraram que o NSGA-II é uma meta heurística apropriada para a solução de problemas de otimização com muitos objetivos, especialmente se dois objetivos conflitantes são considerados

[10]. Para tratar com problemas com mais de três objetivos conflitantes, Deb *et al.* propuseram em 2014 uma versão melhorada do NSGA-II, denominada NSGA-III [11]. O NSGA-III funciona da seguinte maneira. Inicialmente, um conjunto de pontos de referência são gerados utilizando um método sistemático descrito em [12]. Nesse estágio, o NSGA-III inicializa a população P_t com N indivíduos. Em seguida, o NSGA-III realiza os seguintes passos a cada geração t . Primeiramente, o algoritmo cria uma população descendente Q_t a partir da população P_t utilizando algumas operações como seleção aleatória, cruzamento e mutação. Em seguida, um mecanismo de seleção é utilizado na população combinada $R_t = P_t \cup Q_t$ para eliminar soluções extras. Assim, R_t é ordenada pelo método de *fast-non-dominated sorting* e categorizada em diferentes frentes de Pareto. Então, a nova população P_{t+1} é preenchida por cada nível em sequência até o nível F_t que não pode ser incluído integralmente sem que o limite de tamanho para a nova população seja ultrapassado. Essa população parcial é denominada S_t . Quando a população P_{t+1} está incompleta, o algoritmo seleciona soluções da população F_1 utilizando uma nova técnica que associa cada solução na população temporária S_t com um ponto de referência que possui a menor distância perpendicular em relação a solução. Depois disso, o NSGA-III aplica uma seleção incremental de indivíduos da população F_t . Esse procedimento é executado da seguinte maneira. Primeiramente, seja p_j definido como a contagem de indivíduos em S_t para o j -ésimo ponto de referência, e μ_j como o número de indivíduos na população F_t já associados com o ponto de referência j . Assim, o mecanismo de preservação de grupos começa procurando um conjunto de pontos de referência J_{min} com o mínimo definido por p_j e um desses pontos j é aleatoriamente selecionado para o conjunto. Se $p_j = 0$, isso significa que não existe um indivíduo em S_t associado com o ponto de referência \bar{j} . Desta forma, dois casos podem acontecer: a) se $\mu_j \neq 0$, então o indivíduo com a menor distância perpendicular em relação a linha de referência é adicionado a P_{t+1} e 1 é acrescido ao valor do termo p_j ; e b) Se $\mu_j = 0$, então o ponto de referência atual é eliminado. Se $p_j \geq 0$ então existem um ou mais indivíduos em S_t associados com o ponto de referência e, nesse caso, um indivíduo é aleatoriamente escolhido de F_t que esteja associado com o ponto de referência \bar{j} e é adicionado a P_{t+1} . Além disso, o termo p_j também tem seu valor incrementado por 1. Para mais detalhes sobre o algoritmo veja o artigo original em [11].

B. Hipervolume

No processo de otimização multiobjetivo existem dois objetivos distintos: descobrir soluções mais próximas possível de soluções ótimas de Pareto, e achar soluções as mais diversas possível na frente não dominada. Desta forma, para comparar e avaliar o desempenho da otimização é preciso utilizar métricas de desempenho que avaliem a convergência até a frente de Pareto ótima e a diversidade da frente de Pareto obtida. O Hipervolume [13] é uma métrica que proporciona uma medida qualitativa tanto de convergência quanto de diversidade. Se soluções são considerados pontos em um espaço objetivo, o hipervolume é o espaço n -dimensional que é contido por uma

população de soluções. Devido a propriedades do indicador [14], uma população com um maior hipervolume é mais provável de apresentar um conjunto com melhor compromisso dentre os diversos objetivos.

III. FERRAMENTAS UTILIZADAS PARA A REALIZAÇÃO DA PESQUISA

A. Base de Dados

Para validar a proposta deste trabalho foram utilizadas rotas de ônibus que estão atualmente implementadas, comparando-as com as novas soluções obtidas pelo processo de otimização. Portanto, um dos passos iniciais na execução da pesquisa consistiu em buscar dados referentes às infraestruturas de transportes públicos que estão atualmente em uso. Considerando a disponibilidade e correteza de dados abertos de diversas regiões metropolitanas, optou-se pela base de dados SPTrans referente à região metropolitana de São Paulo, que disponibiliza os seus dados em formato GTFS [15].

B. APIs de Roteamento

No processo de otimização de uma linha de transporte público é preciso avaliar a qualidade das soluções candidatas (*fitness*). Nesta proposta o processo de avaliação consiste em avaliar a variação da distância percorrida pelos veículos da linha entre os terminais de parada, a média de tempo de caminhada dos passageiros até as paradas (da origem até a parada de início e da parada final até o destino efetivo), e o tempo médio que estes passageiros gastaram percorrendo a linha dentro do ônibus. Para realizar os cálculos relacionados aos objetivos de otimização foi preciso a utilização de uma API que possui a funcionalidade de cálculo de rotas. Uma pesquisa foi executada para tentar encontrar a API que melhor atendesse às necessidades da pesquisa, resultando na consideração de três principais APIs: Google Maps API [16], Bing Maps API [17] e OSRM API [18]. A Tabela I apresenta os principais aspectos considerados para cada API.

A primeira API considerada para o simulador foi a Google Maps API, uma vez que esta é desenvolvida pela Google, possui uma documentação de fácil leitura e é a API mais popular no mercado. A API apresentou todas as ferramentas e funcionalidades requeridas pela aplicação, sendo capaz de calcular rotas que passam por pontos específicos, chamados de *waypoints*, ao mesmo tempo que fornece a distância percorrida e o tempo gasto entre estes *waypoints*. Entretanto, a Google Maps API apresentou aspectos negativos que desencorajaram o uso nessa pesquisa. Uma chave gratuita da Google Maps API possui um limite de 25000 requisições diárias com apenas 25 *waypoints* cada. Um itinerário de ônibus na cidade de São Paulo possui em torno 60 terminais de parada ao longo de sua rota, resultando na necessidade de 3 requisições para calcular totalmente uma rota. Para apresentar resultados significantes, a aplicação precisaria calcular mais de 40000 rotas. Com o limite de 25000 requisições diárias somado à necessidade de gastar 3 dessas requisições para calcular uma única rota, não seria possível obter bons resultados em um espaço de tempo razoável. Para contornar estes limites seria preciso entrar pessoalmente em contato com a Google para elaborar

um plano de contrato e obter uma chave *premium*, resultando em custo financeiro adicional.

A segunda API considerada foi a Bing Maps API, desenvolvida pela Microsoft. Tanto a Google Maps API quanto a Bing Maps API apresentam cálculos de rotas com resultados semelhantes. Diferente da API da Google, a Bing Maps API possui o mapeamento de São Paulo desatualizado, com vias recentemente construídas ainda não mapeadas e com fluxos de direção desatualizados. Este ponto negativo, entretanto, não traria problemas para aplicação, uma vez que o processo de otimização poderia ser executado com a utilização de qualquer mapeamento. Assim como a Google Maps API, a Bing Maps API também possui limitações em sua chave gratuita, possibilitando apenas 25 *waypoints* por requisição e disponibilizando apenas 125000 requisições acumulativas por ano. Além da chave gratuita básica, a Bing Maps API também disponibiliza uma chave gratuita educacional para projetos não lucrativos. Esta chave mantém o limite de 25 *waypoints* por requisição e apresenta uma melhoria em relação ao limite de requisições, possibilitando a solicitação de 50000 requisições diárias. 50000 requisições diárias é uma melhoria em relação ao limite de 25000 requisições diárias da Google Maps API, porém ainda não é um número adequado para obter-se bons resultados em um espaço de tempo razoável, levando-se em conta o limite de 25 *waypoints* por requisição, que resultam na necessidade de 3 requisições para calcular uma rota de 60 terminais por completo. Assim como a API da Google, os limites da Bing Maps API também podem ser contornados ao elaborar-se um contrato com a Microsoft para a aquisição de uma chave *enterprise*, também resultando em custo financeiro adicional.

A terceira API considerada, *Open Source Routing Machine - OSRM*, representa uma solução *open source* que não apresenta muitas limitações ao usuário. O projeto OSRM visa implementar um motor de roteamento de alto desempenho de forma gratuita. Este motor utiliza dados abertos do OpenStreetMap, outro projeto que visa disponibilizar um mapeamento global gratuito. Para utilizar o sistema de roteamento é preciso antes implementá-lo em uma máquina que servirá como um servidor de API no estilo REST. Esta máquina recebe requisições no formato HTTP contendo os *waypoints* da rota, e retorna o resultado do roteamento no formato JSON. A OSRM API não possui limite de *waypoints* por rota e, já que é implementada em um servidor próprio, não possui limite de requisições diárias. Outra vantagem desta API é que o seu servidor pode ser implementado na mesma rede que a máquina executando a aplicação da pesquisa, possibilitando assim um maior desempenho no recebimento de requisições e no envio de respostas. Entretanto, a API apresenta duas desvantagens em relação as outras APIs consideradas: a necessidade de ter uma máquina extra para implementar o servidor da API, e a menor eficiência no sistema de cálculo de rotas.

Considerando as vantagens e desvantagens de cada API, optou-se pela OSRM API para compor o motor de avaliação do processo de otimização desta pesquisa.

TABELA I
APIs DE ROTEAMENTO

	Google Maps API - Chave gratuita	Bing Maps API - Chave educacional	OSRM - Servidor próprio
Requisições gratuitas	25000 diárias	50000 diárias	Ilimitado
Waypoints	25 waypoints por requisição	25 waypoints por requisição	Ilimitado

C. Ambiente

No desenvolvimento da aplicação utilizou-se o JMetal [19], *framework* baseada na linguagem de programação Java e que implementa vários algoritmos de otimização, entre eles o NSGA-III, e que também possui um conjunto bem conhecido de indicadores de qualidade para problemas multiobjetivo. Outras bibliotecas de programação para Java também foram usadas no desenvolvimento da aplicação, tais como: Apache HttpComponents para lidar com requisições HTTP; e Org.Json para manusear as respostas no formato JSON vindas da API de mapeamento.

IV. DESCRIÇÃO DO PROBLEMA

O problema de pesquisa tratado neste artigo pode ser formalmente formulado da seguinte forma. Considerando as geolocalizações das estações/paradas em uma linha de transporte público como variáveis de decisão, o objetivo é encontrar novas posições para as paradas (e novas rotas) considerando os seguintes objetivos: minimizar a média de tempo que os passageiros gastam andando de suas localizações até as paradas; minimizar a média de tempo que os passageiros permanecem no ônibus (tempo de viagem); e minimizar a variância da distância entre os terminais de parada na linha (uniformidade no atendimento à população).

As geolocalizações dos terminais são definidas por $T = (lat_i, lon_i)$, em que $(i \in 1, 2, \dots, N)$. lat_i e lon_i são coordenadas em graus da N-ésima estação/parada e N é o número total de paradas.

A seguinte técnica foi utilizada para simular viagens realizadas por usuários que fazem uso da linha. Dadas as geolocalizações dos terminais de parada em um determinado itinerário de uma linha de transporte público, definem-se limites considerando as paradas com maior latitude, menor latitude, maior longitude e menor longitude. Os valores da maior latitude e maior longitude são incrementados por um número m de metros, e por sua vez os valores da menor latitude e menor longitude são decrementados por este mesmo valor, formando um quadrilátero que é considerado o limite das redondezas da linha de transporte sendo avaliada. No mapeamento utilizado o número de quilômetros por grau de coordenada equivale a mais ou menos 111km. Para este problema foi considerado o valor 111,32km. Um quilometro em graus de coordenada equivale a $1/111,32 = 0,0089$. Por sua vez, um metro em graus de coordenada equivale a $0,0089/1000 = 0,000089$. Desta forma, para incrementar ou decrementar os valores de latitude ou longitude em metros calcula-se $c = m * 0,000089$ e adiciona-se ou subtrai-se o valor de c ao valor da coordenada em questão. As coordenadas $[Lat_{max} + c; Lat_{min} - c; Lon_{max} + c; Lon_{min} - c]$ formam o quadrilátero mencionado anteriormente. Dois pontos são aleatoriamente gerados dentro do quadrilátero, um considerado

a origem e o outro o destino de um passageiro no simulador. Para simular uma viagem da origem até o destino os seguintes passos são seguidos. Utiliza-se a API de roteamento para calcular o tempo t_o em segundos que uma pessoa leva para andar do ponto de origem até o terminal de parada mais próximo a este ponto. Em seguida a API é novamente utilizada para determinar o tempo t_v em segundos gasto percorrendo a linha dentro de um veículo de transporte até o terminal mais próximo do ponto de destino. E por último a API é mais uma vez utilizada para calcular o tempo t_d em segundos que uma pessoa levaria para andar do terminal de descida até o ponto de destino. Seja w a soma de t_o e t_d em uma única viagem, uma quantidade x de simulações são efetuadas e a média do tempo de caminhada é obtida utilizando a Eq. 1.

$$\frac{\sum_{i=1}^x w_i}{x} \tag{1}$$

Por sua vez os valores de t_v obtidos com as x simulações são utilizadas para calcular a média de tempo de viagem dentro da linha, que pode ser definida pela Eq. 2.

$$\frac{\sum_{i=1}^x t_{vi}}{x} \tag{2}$$

Para determinar a variância da distância entre os terminais de parada, as distâncias entre os terminais da linha, medidas em metros, são obtidas através da API de roteamento e em seguida o cálculo de variância é efetuado. Portanto, nesta pesquisa foi considerado um simulador de eventos discretos, no qual um evento consiste na subida/descida de um passageiro e este processo segue uma distribuição de probabilidade uniforme. Desta forma, a avaliação de aptidão de novas soluções candidatas para a linha de ônibus consiste em executar este simulador na tentativa de emular o comportamento real de uma população de passageiros que potencialmente usaram a linha considerada. Em suma, o problema pode ser resumido conforme equações abaixo.

Descrição do problema: Dadas as variáveis de decisão representadas pela Eq. 3

$$\vec{X} = [T_1, T_2, \dots, T_N], \tag{3}$$

o objetivo é

$$\text{minimizar} \begin{cases} f_1(\vec{X}) = MTC \\ f_2(\vec{X}) = MTV \\ f_3(\vec{X}) = V \end{cases} . \tag{4}$$

Na Eq. 4 MTC é a média do tempo de caminhada, em segundos, MTV é a média do tempo de viagem dentro da linha, em segundos, e V é a variância da distância entre paradas, em metros.

A. Arranjo Experimental

Esta subseção descreve detalhes sobre os experimentos considerados inicialmente para validar a proposta. O itinerário escolhido para ser otimizado foi o da linha número "423032", "Sao Bernardo do Campo (Jardim Las Palmas) / São Paulo (Terminal Sacoma)", com 69 terminais de parada. Após receber o itinerário selecionado, a aplicação cria uma população inicial P_1 para o algoritmo NSGA-III de tamanho 92, sendo este o tamanho definido para todas as populações geradas pelo NSGA-III. Cada indivíduo da população P_1 possui um cromossomo com $69 \times 2 = 138$ posições (genes). O primeiro indivíduo da população, denominado de p_1 , tem as suas variáveis com valores equivalentes aos valores das latitudes e longitudes dos terminais do itinerário original escolhido (latitudes e longitudes da linha em operação). Todos os outros indivíduos da população P_1 são criados da seguinte maneira. Considerando que p_1 é o indivíduo a ser criado, 800 é o valor de m , e m_g é a conversão de m para graus de coordenada, o valor de cada variável em p_i será igual ao valor da variável em p_1 com posição equivalente em seu vetor de características somado a um valor gerado aleatoriamente entre 0 e m_g . Depois da criação da população inicial, o NSGA-III é executado da forma tradicional, mas operadores especiais foram desenvolvidos na pesquisa. O método de mutação foi um destes operadores. A mutação funciona da seguinte maneira. Considerando um determinado conjunto de duas variáveis que representa a geolocalização de um terminal de parada, é identificada a distância d entre este terminal e outro terminal mais próximo no cromossomo considerado, e em seguida um valor aleatório entre $-d$ e d é gerado e somado a cada variável do conjunto de duas variáveis. A probabilidade de mutação é definida como $1,0/(|\vec{X}|)$. No caso da linha 423032, a probabilidade de mutação é $1,0/138 = 0,007$, ou 0,7%. O Torneio Binário foi escolhido empiricamente como método de seleção pois obteve sucesso em problemas semelhantes anteriormente [7]. O operador de cruzamento adotado seleciona dois indivíduos para gerar dois descendentes cujas variáveis receberão valores do pai 1 ou do pai 2. Cada conjunto de duas variáveis do descendente 1 tem 25% de chance para receber valor do pai 2 e 75% de chance para receber valor do pai 1. Por sua vez cada conjunto de duas variáveis do descendente 2 tem 75% de chance para receber valor do pai 2 e 25% de chance para receber valor do pai 1. Finalmente, foi determinado que 30 simulações de viagens serão executadas para avaliar cada indivíduo. Foi definido como critério de parada uma quantidade máxima de 400 gerações.

V. RESULTADOS

Utilizando um laptop com processador Intel i5-3337U e 6GB de memória RAM DDR3 como servidor do serviço de roteamento, um computador pessoal com processador Intel Core i5-4590 e 8GB de memória RAM DDR3 1600MHz para rodar o algoritmo de otimização, e com ambas as máquinas presentes na mesma rede local, cada execução do algoritmo com 400 gerações levou em média 9 horas para ser concluída.

O gráfico da Fig. 1 apresenta a média do hipervolume considerando 30 execuções independentes do algoritmo. Com

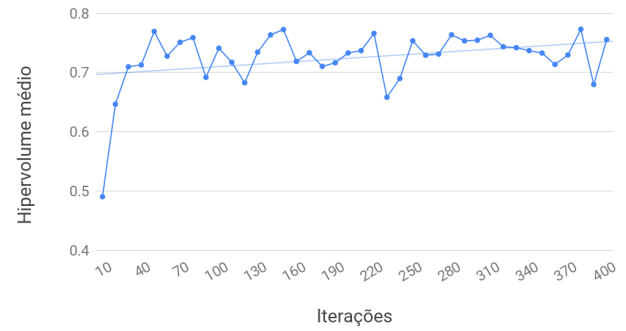


Fig. 1. Hipervolume vs. iterações.

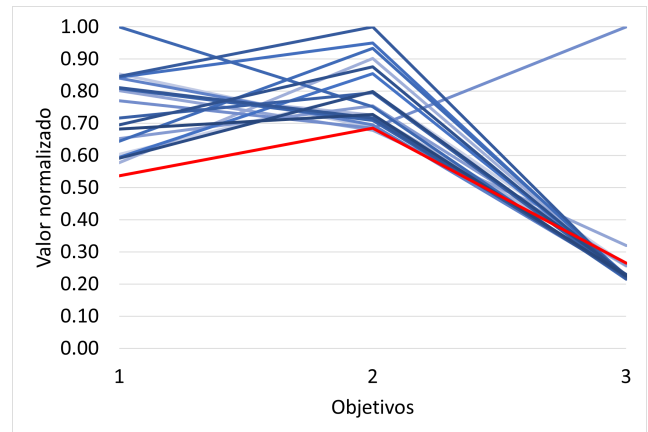


Fig. 2. Exemplo de frente de Pareto em gráfico de coordenadas paralelas.

base nas execuções, foi possível perceber que as primeiras iterações do algoritmo levam o hipervolume para um patamar de 0,45 e obtém-se um pico pouco acima de 0,9, que é a região com soluções de maior qualidade.

A Fig. 2 representa um exemplo de frente de Pareto em gráfico de coordenadas paralelas. Este gráfico foi obtido em uma das execuções do algoritmo que forneceu um hipervolume de 0,96. Na Fig. 2, o objetivo 1 é referente ao tempo médio de viagem; o objetivo 2 é referente ao tempo médio que os indivíduos gastam caminhando até as paradas; o objetivo 3 é referente à variância da distância entre as paradas.

A Fig. 3 represente a linha de ônibus original da linha "423032" a Fig. 4 representa a solução em vermelho destacada na Fig. 2, sendo uma nova solução otimizada obtida pela proposta deste artigo. Embora as linhas sejam semelhantes quando é considerada uma inspeção visual no mapa, os valores numéricos dos objetivos considerados evidenciam a vantagem em usar o algoritmo proposto neste trabalho. A solução original da linha "423032" obteve a seguinte avaliação: 872,3 segundos como o tempo médio de viagem pela linha; 573 segundos como tempo o médio de caminhada até o destino; 140574 metros quadrados de variância na distância entre os terminais da solução. A solução destacada em vermelho no gráfico da Fig. 2 obteve a seguinte avaliação: 676,5 segundos como tempo médio de viagem dentro da linha; 469 segundos como tempo médio de caminhada até o destino; e 19156,5

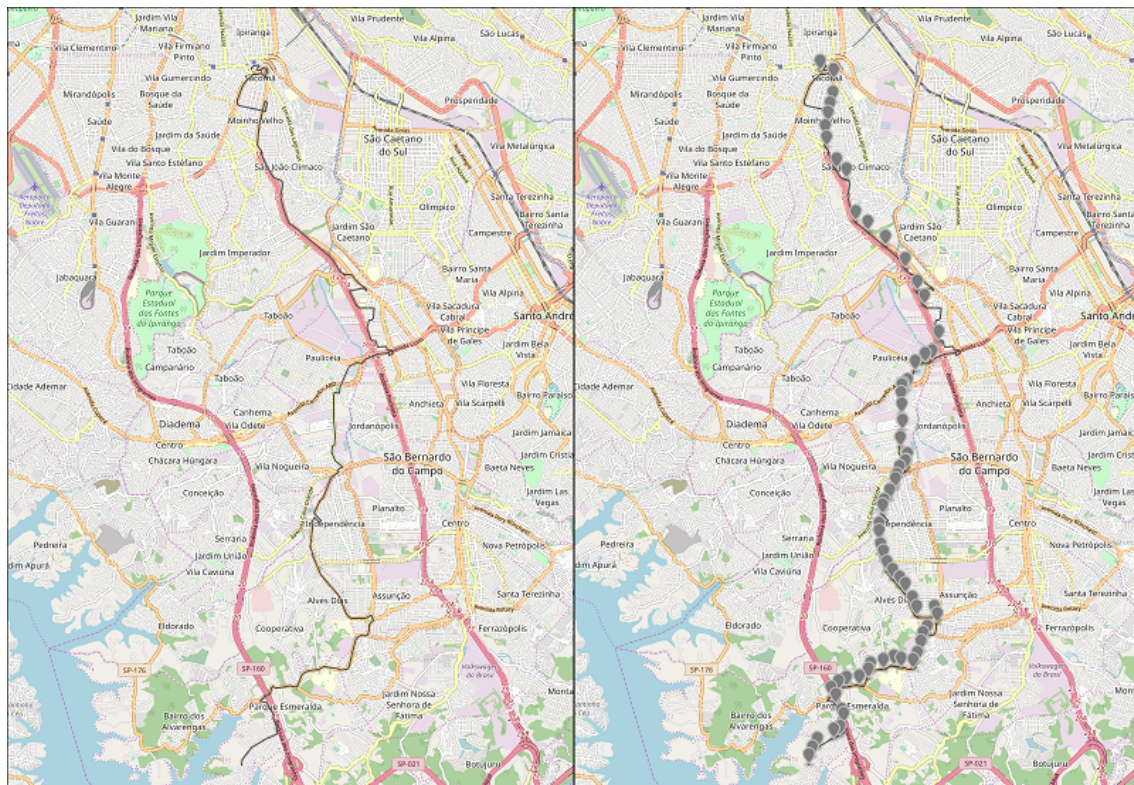


Fig. 3. Solução original.

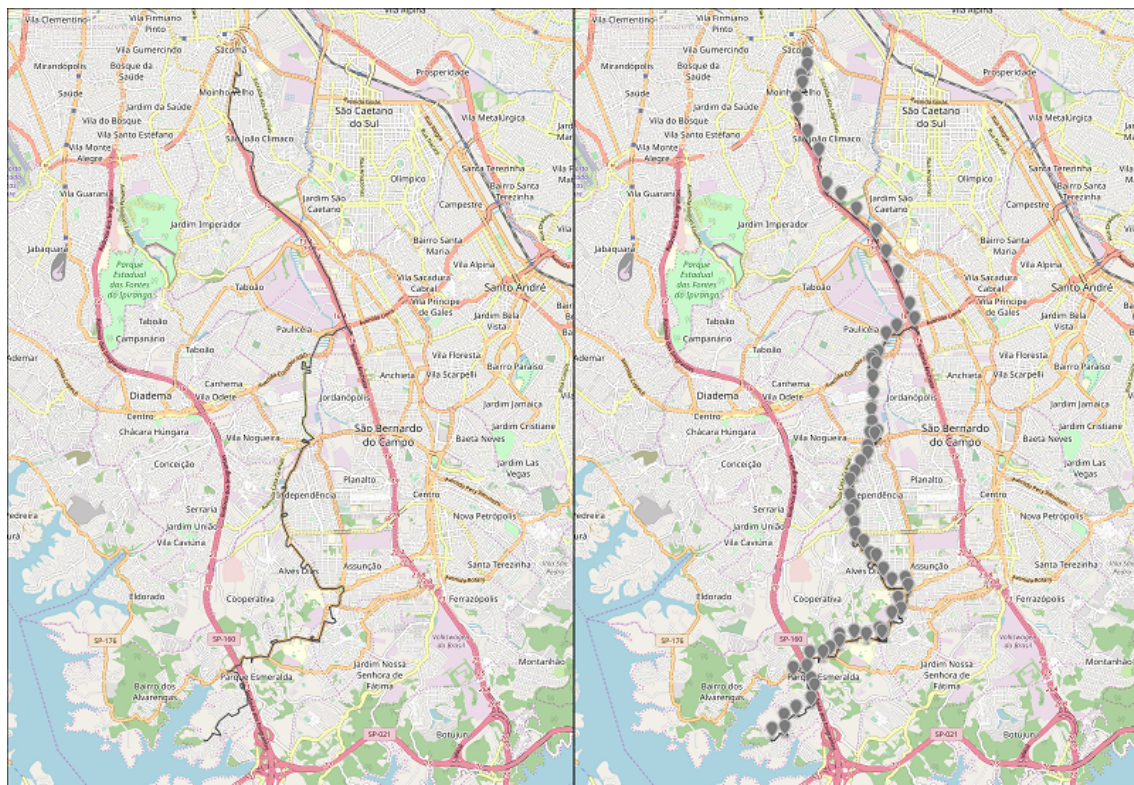


Fig. 4. Solução obtida pelo algoritmo.

metros quadrados de variância na distância entre os terminais de parada na solução. Além de menor variância das distâncias entre paradas (melhoria de atendimento à população), obtém-

se uma redução de 22,44 % no tempo médio de viagem e uma redução de 18,15% no tempo de caminhada.

VI. CONCLUSÕES

Este artigo propõe uma abordagem multiobjetivo para tratar o problema de otimização de linhas de transporte público considerando a otimização de diversos objetivos importantes para a população atendida, como tempo de caminhada até as paradas, tempo de viagem e variância da posição das paradas. A abordagem considera a determinação da posição das paradas e a definição das rotas, usando o algoritmo NSGA-III como linha mestra para otimização multiobjetivo. Para validação da proposta foi considerada uma importante linha de ônibus da cidade de São Paulo e uma das soluções obtidas pela proposta do artigo obteve melhoria significativa em todos os objetivos considerados, chegando a uma redução acima de 22% no tempo de viagem.

Uma das limitações desta abordagem foi a incapacidade de considerar fatores como a qualidade do asfaltamento das ruas, a densidade demográfica de diversas regiões e a intensidade de movimentação de veículos nas vias, desta forma pode ser considerado melhorar o simulador de eventos discretos em trabalhos futuros para considerar esses fatores.

Referente à outras sugestões para trabalhos futuros, devem ser consideradas diversas outras linhas de transporte público para demonstração da capacidade de generalização da proposta, incluindo outros modais (como metrô). Com relação aos aspectos técnicos do algoritmo proposto, poderão ser feitos estudos sobre o uso de algoritmos de clusterização para proposição de operadores meméticos para busca local da posição das paradas, e estudos com outras meta-heurísticas e ajustes paramétricos durante a execução do algoritmo.

AGRADECIMENTOS

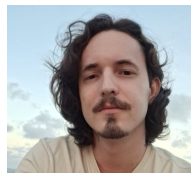
Os autores agradecem o apoio financeiro fornecido pela UFRPE e CNPq.

REFERENCES

- [1] A. N. R. da Silva, M. da Silva Costa, and M. H. Macedo, "Multiple views of sustainable urban mobility: The case of Brazil," *Transport Policy*, vol. 15, no. 6, pp. 350–360, 2008. [Online]. Available: <https://doi.org/10.1016/j.tranpol.2008.12.003>
- [2] L. Olsson, M. Friman, K. Lättman, and S. Fujii, "Travel and life satisfaction - from gen z to the silent generation," *Journal of Transport & Health*, vol. 18, p. 100894, 2020. [Online]. Available: <https://doi.org/10.1016/j.jth.2020.100894>
- [3] J. Minser and V. Webb, "Quantifying the benefits: Application of customer loyalty modeling in public transportation context," *Transportation Research Record*, vol. 2144, no. 1, pp. 111–120, 2010. [Online]. Available: <https://doi.org/10.3141/2144-13>
- [4] M. J. Nieuwenhuisen, "Urban and transport planning pathways to carbon neutral, liveable and healthy cities; a review of the current evidence," *Environment International*, vol. 140, p. 105661, 2020. [Online]. Available: <https://doi.org/10.1016/j.envint.2020.105661>
- [5] M. Friman, K. Lättman, and L. E. Olsson, "Public transport quality, safety, and perceived accessibility," *Sustainability*, vol. 12, no. 9, 2020. [Online]. Available: <https://doi.org/10.3390/su12093563>
- [6] N. Banerjee and R. Kumar, "Multiobjective network design for realistic traffic models," in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 1904–1911. [Online]. Available: <https://doi.org/10.1145/1276958.1277341>
- [7] J. Candeias, D. R. de Araújo, P. Miranda, and C. J. Bastos-Filho, "Memetic evolutionary algorithms to design optical networks with a local search that improves diversity," *Expert Systems with Applications*, vol. 232, p. 120805, 2023. [Online]. Available: <https://doi.org/10.1016/j.eswa.2023.120805>

- [8] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 48, no. 1, sep 2015. [Online]. Available: <https://doi.org/10.1145/2792984>
- [9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002. [Online]. Available: <https://doi.org/10.1109/4235.996017>
- [10] C. A. Coello Coello, C. Dhaenens, and L. Jourdan, *Multi-Objective Combinatorial Optimization: Problematic and Context*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–21. [Online]. Available: https://doi.org/10.1007/978-3-642-11218-8_1
- [11] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014. [Online]. Available: <https://doi.org/10.1109/TEVC.2013.2281535>
- [12] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998. [Online]. Available: <https://doi.org/10.1137/S1052623496307510>
- [13] L. Bradstreet, "The hypervolume indicator for multi-objective optimisation: calculation and use," Ph.D. dissertation, 2011. [Online]. Available: <https://research-repository.uwa.edu.au/en/publications/the-hypervolume-indicator-for-multi-objective-optimisation-calcul>
- [14] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003. [Online]. Available: <https://doi.org/10.1109/TEVC.2003.810758>
- [15] SPTrans, "Gtfs - sp," <http://www.sptrans.com.br/desenvolvedores/perfil-desenvolvedor/>, accessed: 2023-09-26.
- [16] G. LLC, "Google maps api," <https://mapsplatform.google.com/>, accessed: 2023-09-26.
- [17] M. Corporation, "Bing maps api," <https://www.microsoft.com/en-us/maps/bing-maps/choose-your-bing-maps-api>, accessed: 2023-09-26.
- [18] P. OSRM, "Osm api," <https://github.com/Project-OSRM>, accessed: 2023-09-26.
- [19] M. V. A.J. Nebro, J.J. Durillo, "jmetal framework," <https://github.com/jMetal/jMetal>, accessed: 2023-09-26.

Ingo Porfirio Pastl Montarroyos joined the Bachelor's degree in Computer Science at UFRPE - Universidade Federal Rural de Pernambuco as a student in 2016. He currently holds the position of Solutions Specialist at Truewind.



Danilo R. B. de Araújo was born in Recife, Brazil. He received the B.Sc. degree in Computer Science from the Catholic University of Pernambuco, in Recife, in 2002, and the M.Sc. degree in Computer Engineering from University of Pernambuco in 2009. He received his PhD degree in Electronics Engineering from the Federal University of Pernambuco, Recife in 2015. In 2013 he joined the Department of Computing at Federal Rural University of Pernambuco, in Recife, where he currently is an Associate Professor. His research interests are in networking, simulation, modelling and optimization. He has coordinated several research projects financed by state and federal agencies. He also took part in research and development (R&D) projects sponsored by Itaotec-OkI and Padtec-Brazil since 2003. He has authored more than 50 papers last 10 years. Prof. Danilo Araújo is Member of SBC (Institute of Electrical and Electronics Engineers) and Member of SBC (Brazilian Society of Computing). RID: <http://www.researcherid.com/rid/1-3029-2015>. Citations: <https://scholar.google.com.br/citations?user=o6eUGfAAAAAJ&hl=pt-BR>.



ACRONYMS

AEMO Algoritmo Evolucionário Multiobjetivo