revista científica LINKSCIENCEPLACE interdisciplinar

Revista Científica Interdisciplinar. ISSN: 2358-8411
Nº 1, volume 1, artigo nº 1, Julho/Setembro 2014
D.O.I: 10.17115/2358-8411/v1n1a1

PROGRAMAÇÃO DISTRIBUIDA PARA OTIMIZAÇÃO DE HEURÍSTICA ILS APLICADA A PROBLEMAS DO CAIXEIRO VIAJANTE

Beatriz Damica Freitas[1]

Graduada em Sistemas de Informação

Bruno Missi Xavier[2]

Mestre em Pesquisa Operacional e Inteligência Computacional

Fabio Machado de Oliveira[3]

Mestrando em Cognição e Linguagem

Marcelo Albuquerque Schuster[4]

Mestre em Pesquisa Operacional e Inteligência Computacional

Resumo

Atualmente há uma grande procura por técnicas que agilizam o processamento de algoritmos usados na resolução de problemas cujo tempo computacional cresce exponencialmente. Este trabalho tem o objetivo de analisar os tempos computacionais de uma solução proposta para o Problema do Caixeiro Viajante utilizando o algoritmo ILS com programação distribuída. A comunicação entre as máquinas é facilitada por meio de Socket TCP, viabilizando o compartilhamento de recursos e fornecendo sincronização no envio dos dados. Os resultados foram obtidos por meio da execução do algoritmo em 1, 10, 15 e 20 máquinas

[1] Centro Universitário São Camilo, Cachoeiro de Itapemirim/ES, biadamif@gmail.com
[2] Universidade Cândido Mendes, Campos dos Goytacazes/RJ, bmissix@gmail.com
[3]Universidade Estadual do Norte Fluminense, Campos dos Goytacazes/RJ, fabiomac@gmail.com
[4] Universidade Cândido Mendes, Campos dos Goytacazes/RJ, marceloashuster@gmail.com

remotas utilizando três instâncias distintas, sendo estas providas da biblioteca TSPLIB. A contribuição para os diversos estudos relacionados com a otimização de tempos computacionais caracteriza a importância deste trabalho. Além disso, o mesmo aborda um modelo genérico e eficiente para distribuição de heurísticas baseadas em busca local.

Palavras-chave: Iterated Local Search; Programação Distribuída; Socket TCP; Problema do Caixeiro Viajante.

Abstract

Currently there is a great demand for techniques that streamline the processing algorithms used in solving problems whose computational time grows exponentially. This work aims to analyze the computational times of a proposal for the Traveling Salesman Problem using the ILS algorithm with distributed programming solution. Communication between machines is facilitated via TCP socket, enabling the sharing of resources and providing synchronization in data transmission. The results were obtained by running the algorithm on 1, 10, 15 and 20 remote machines using three separate bodies, and these provided the TSPLIB library. The contribution to the various studies related to the optimization of computational time characterizes the importance of this work. Moreover, it addresses a generic and efficient model for distribution of location-based heuristic search.

Keywords: Iterated Local Search; Distributed Programming; Socket TCP; Travelling Salesman Problem.

1. INTRODUÇÃO

Nos tempos atuais é cada vez maior a busca pela otimização de processos empresariais objetivando a redução do tempo e custo. Em algumas situações estes processos estão diretamente relacionados a problemas de Otimização Combinatória e são objetos de estudo da área do conhecimento denominada Pesquisa Operacional (PO). Os métodos de PO estão sendo muito utilizados para encontrar e promover a eficácia das resoluções de problemas do mundo real. Segundo Medeiros (2012), a PO tem beneficiado muitos setores, entre eles, o segmento de logística, para o qual proporciona aspectos importantes na entrega dos produtos aos clientes, por meio da definição dos melhores caminhos a serem seguido por veículos.

O Problema do Caixeiro Viajante (PCV) é um dos problemas de otimização combinatória e segundo Blume (2012), esta classe de problemas demonstra grandes dificuldades para encontrar soluções exatas em tempo computacional satisfatório. Cunha, Bonasser e Abrahão (2002) afirmam que o PCV foi desenvolvido por William Rowan Hamilton por meio de um jogo, cuja finalidade era traçar rotas entre os vértices de um dodecaedro, passando por todos os vértices até chegar ao ponto inicial, sem visitar os vértices mais de uma vez.

O PCV pode ser classificado como NP-Difícil, pois exige maior esforço computacional para encontrar uma solução aproximada. Segundo Constantino e Júnior (2008), métodos heurísticos são mais adequados para a resolução de problemas NP-Difíceis por obterem uma solução aceitável em tempo computacional apropriado. Diaz (1996) afirma que heurística pode ser um algoritmo capaz de alcançar soluções próximas do real para um problema específico, por meio de uma função objetivo.

Métodos heurísticos tem o objetivo de encontrar as melhores soluções em tempos computacionais acessíveis, no meio de uma coleção de elementos discretos. Um problema de otimização combinatória compreende a relação das variáveis de decisão com a uma função objetivo e um conjunto de restrições. Para Russell e Norvig (2004), uma função objetivo pode obter valores mínimos ou máximos dentro do espaço de busca delineado para o problema.

Apesar dos métodos heurísticos serem apropriados para a resolução de problemas NP - Difíceis, nem sempre o tempo computacional necessário para a resolução de um problema é satisfatório para processos que necessitem de respostas em tempo real. Desta forma, técnicas de programação distribuída auxiliam na otimização da execução de heurísticas.

Este trabalho tem o objetivo de avaliar a execução de uma heurística Iterated Local Search (ILS) e compará-la ao algoritmo ILS distribuído utilizando 10, 15 e 20 máquinas remotas. As instâncias dos problemas foram selecionadas a partir da base da TSPLIB (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95), representando 96, 226 e 318 pontos. Os resultados apresentados demonstram que o algoritmo ILS distribuído alcança melhor performance na resolução destes problemas.

O presente artigo está organizado conforme explicado a seguir: seção 1, de caráter introdutório; seção 2, os procedimentos metodológicos; seção 3, com a

construção do modelo de avaliação de desempenho do estudo de caso; e seção 4, com as considerações finais.

2. MATERIAL E MÉTODOS

Nesta sessão, inicialmente serão abordados os trabalhos da literatura científica relacionados à área de heurísticas distribuídas, em ordem cronológica. Posteriormente serão apresentados os conceitos tecnológicos abordados neste estudo, a fim de proporcionar maior compreensão e fundamentação do assunto.

2.1 Literatura relacionada

Santos, Lopes e Júnior (2006) abordam uma proposta de solução para PCV assimétrico que é caracterizado por possuir distâncias distintas entres as cidades. O método foi elaborado com utilização de duas heurísticas, 3-opt e o método do vizinho mais próximo, o primeiro é utilizado para obter melhorias no roteiro e o segundo foi utilizado para construção de roteiros. A partir do seu desenvolvimento, o método proposto foi submetido a dois procedimentos de paralelização, os quais foram denominados “Método 1” e “Método 2”, estes possuem objetivos diferentes. O Método 1 visa a minimização do tempo de processamento e o Método 2 busca a qualidade das respostas obtidas. Para a paralelização foi utilizada a arquitetura *Cluster*, caracterizando-se por ser um sistema distribuído composto por máquinas autônomas. De acordo com os resultados obtidos, os autores afirmam que a utilização de técnicas paralelas pode beneficiar o processamento das soluções propostas para o PCV assimétrico.

A pesquisa abordada por Gazolla (2010) evidencia uma solução que compreende um algoritmo heurístico e paralelo utilizando processamento em *Graphics Processing Unit* (GPU) ou Unidade de Processamento Gráfico. Segundo o autor as GPUs possuem poder de processamento inúmeras vezes maiores que as CPUs, ocasionando a rapidez e agilidade de obtenção das melhores soluções em distintos aspectos da computação. O método de busca local utilizado é o *Random Variable Neighborhood Descent* (RVND). Este método busca o melhoramento da solução por meio de movimentos paralelizados e desenvolvidos em GPU. Através

dos resultados encontrados, o autor conclui que o método proposto conseguiu eficiência ao obter respostas satisfatórias e diversas vezes mais ágeis que o processamento em CPU. Porém, com relação à qualidade, as respostas obtidas foram razoáveis.

Blume (2012) propõe uma solução para o PCV utilizando método baseado no algoritmo *Simulated Annealing* (SA). Segundo o autor, a analogia física do algoritmo SA esta relacionada com a definição do procedimento de recozimento de um sólido, com a finalidade de atingir sua condição desejada minimizando a quantidade de energia utilizada no processo. O método proposto compreende a paralelização do SA com a utilização da biblioteca *Message Passing Interface* (MPI), a mesma consiste em distribuir uma réplica do algoritmo para cada processo, executando-os de forma sincronizada com dados iniciais idênticos. As execuções são realizadas independentemente, até alcançar resultados com temperaturas inferiores à faixa. Posteriormente os resultados são enviados a um processo que concentra todas as respostas obtidas pelas execuções, o melhor resultado é selecionado e enviando novamente para os processos executarem com uma nova faixa de temperatura. Conforme o autor, a partir dos resultados obtidos, o modelo proposto demonstra-se favorável em relação ao encontro de diversas soluções ótimas, sendo que a quantidade de soluções obtidas foi proporcional ao aumento do número de processos.

Em seu trabalho, Medeiros (2012) aponta uma solução para o Problema de roteamento de Veículos Capacitados (PRVC), no qual combina uma coleção de restrições para encontrar o melhor caminho. As restrições determinam que cada rota deve começar e terminar em um depósito definido. Um cliente deve ser atendido por apenas um veículo e a capacidade do mesmo não deve ser ultrapassada. . O método exposto no trabalho foi construído da seguinte forma: a solução inicial foi obtida por meio da aplicação da heurística de Teitz e Bart (1968); a clusterização foi realizada com o algoritmo *Cluster First Route Second*, este relaciona duas etapas: o agrupamento dos clientes e a construção das rotas. Para definição das rotas foi utilizado o algoritmo do vizinho mais próximo com o objetivo de aperfeiçoar as rotas foi aplicado a heurística 2-opt. Para a execução do processamento paralelo foi utilizado um software baseado em *Lanquage Integrated Query* (LINQ), juntamente com a uma classe denominada *Parallel*, pertencente a .NET *Framework* 4 e

responsável pelo processamento paralelo do algoritmo. De acordo com o autor, a principal vantagem da utilização de técnicas paralelas é que os cálculos matemáticos são executados em núcleos de processamento diferentes e ao mesmo tempo. Conforme os resultados obtidos, a técnica de processamento paralelo apresentou um tempo médio inferior a 7 segundos, apontando resultados satisfatórios para problemas complexos.

É impor tantes observar que os trabalhos relacionados buscam agilidade na execução dos métodos utilizados, demonstrando distintas formas para solucionar os problemas de alta complexidade. Os resultados obtidos por meio da execução distribuída dos algoritmos, comparando-se com a solução não distribuída apresentam grande avanço não apenas na melhora do tempo computacional, como na qualidade dos resultados. Dentre os artigos relacionados, Santos, Lopes e Júnior (2006) e Gazolla (2010) descrevem métodos heurísticos similares ao abordado neste trabalho apesar de utilizarem técnicas de programação distribuídas distintas.

2.2 Problema do Caixeiro Viajante

O PCV tem sido alvo de muitos estudos, este pode ser aplicado em diversos problemas reais de inúmeras áreas, entre elas, Matemática, Pesquisa Operacional, Física e Biologia (COSTA, 2008). Para Silva e Sanches (2009), a importância do PCV está relacionada a três requisitos: possibilidade da aplicação prática, relação com outros modelos e a grande dificuldade de solução exata.

Cunha (2000), explica que o PCV é representado por um conjunto de cidades sendo necessário encontrar a menor rota, visitando somente uma vez cada cidade. Ao analisar um grafo do PCV, nota-se que as cidades são representadas pelos pontos e as arestas indicam os caminhos a serem percorridos.

Segundo Goldbarg e Luna (2005), a origem do PCV deve-se a William Rowan Hamilton, um físico e matemático, que propôs o Jogo Icosiano. Este era composto por um dodecaedro que continha em cada um de seus vértices o nome de uma cidade, o objetivo do jogo era traçar um caminho passando uma vez em cada cidade até chegar ao vértice inicial.

Cunha, Bonasser e Abrahão (2002), afirma que o PCV é um problema que possui ordem de complexidade exponencial, que exige grande esforço computacional para que a solução seja encontrada. Existem duas formas de

solucionar o PCV, a primeira por meio de métodos exatos que são limitados e consistem em mecanismos de enumeração implícita em árvore, por exemplo, o método Branch & Bound. O segundo são as heurísticas que são capazes de encontrar boas soluções com maiores dimensões e agilidade (COSTA, 2008).

Apesar do PCV ter uma definição muito simples, este apresenta grande grandes desafios, principalmente no que tange a aplicação de problemas do mundo real, geralmente relacionados a logística de roteamento de veículos, sendo necessária a otimização de grandes instâncias de dados.

2.3 Definição do espaço solução

Espaço de busca ou espaço de soluções consiste em um grupo adequado de soluções que satisfazem a uma instância de um problema e são orientados pela Função Objetivo. Ao se referir a problemas de otimização é necessário cumprir algumas etapas para obter resultados satisfatórios. A primeira e principal etapa é a definição de um espaço de soluções na qual é possível explorar as características do problema e permitir a utilização de um eficaz método de busca (NIEVERGELT, 1995). Becceneri (2008) define o espaço de busca como uma coleção que possui todas as respostas válidas para o problema, sendo este espaço considerado finito ou infinito. A Figura 1 apresenta o espaço solução de um problema de Otimização Combinatória.

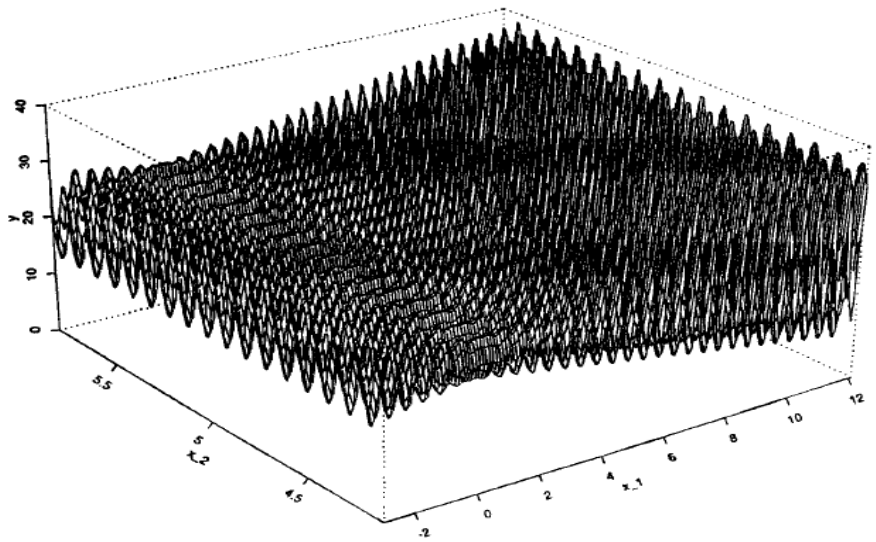

Figura 1 - Espaço solução de um problema de Otimização Combinatória orientado pela Função Objetivo. Faixa de valores para os parâmetros: x1 variando de -3 a 12 e x2 variando de 4,1 e 5,8. Y representa o intervalo de valores da Função Objetivo variando de 0 a 40.
Fonte: Michalewicz (1996)

A Figura 1 apresenta a definição do espaço de solução orientado por meio da Função Objetivo. A medida que são alterados os parâmetros x1 e x2 novamente é calculada a Função Objetivo atribuindo a solução um índice de qualidade. A melhor solução de uma área específica do espaço de solução é conhecida como Ótimo Local enquanto a melhor solução do problema é denominada Ótimo Global.

2.4 Iterated Local Search

A heurística ILS é uma técnica que pode ser utilizada em problemas de Otimização Combinatória, em português significa Busca Local Iterativa. Segundo Neto e Vianna (2011), o ILS procura aprimorar os métodos de Busca Local com a criação de novas soluções a partir da utilização do procedimento de perturbação aplicado na melhor solução encontrada. Andrade, Siqueira e Souza (2012) afirmam que a heurística ILS visa principalmente a variação das buscas, a fim de escapar e propiciar a visita em diversos ótimos locais. Além disso, podem-se definir quatro aspectos importantes que compõem esta meta-heurística: a geração da solução inicial, a aplicação da Busca Local, o procedimento de perturbação e o critério de Aceitação.

Gazolla (2010) cita as quatro fases da heurística ILS, na qual consiste a fase de construção da solução inicial que é feita a partir de métodos heurísticos gulosos ou soluções aleatórias. A etapa em que se aplica o método de Busca Local, visando a buscar soluções mais satisfatórias. A técnica de perturbação aplicada antes da execução do método de busca, sendo utilizada com a finalidade minimizar a probabilidade de achar estacionar a execução do algoritmo em um Ótimo Local não qualificado. Por fim, a solução passa pelo critério de aceitação que consiste em sua aprovação ou reprovação. A heurística ILS apresentada no Quadro 1 aplica método de busca em pequenos espaços de solução compostos por ótimos locais que são encontrados por alguma técnica de otimização. Brito, Montenegro e Ochi (2009) afirmam que o sucesso da aplicação do ILS está ligado a técnica de busca local selecionada e também ao método de perturbação usado na solução, além do critério de aceitação das soluções, esta varia de acordo com o tipo de problema. O desempenho satisfatório da heurística ILS pode se esclarecida devido a ampla flexibilidade obtida a partir divisão dos elementos que a compõe.

1	**Procedimento** ILS()
2	s^1 = SoluçãoInicial();
4	**Enquanto não** Critério_de_Parada() **faça**
5	s^2 = Perturbação(s^1);
6	s^2 = BuscaLocal(s^2);
7	**Se** s^2 **melhor** s^1 **então**
8	$s^1 = s^2$;
9	**Fim Se**;
10	**Fim Enquanto**;
11	**Retorne** s^1;
12	**Fim** ILS;

Quadro 1 - Pseudocódigo padrão da heurística ILS.
Fonte: O próprio autor

Na linha 2, s^1 recebe uma solução inicial atribuída de forma aleatória, enquanto o critério de parada (linha 4) não for atendido, o algoritmo executa a perturbação atribuindo a solução perturbada à s^2 (linha 5)e aplica o método de busca loca à solução (linha 6). Se a solução s^2 for melhor que s^1 (linha 7) então s^1 recebe a solução s^2 (linha 8) e a execução volta a linha 4. Quando o critério de parada for atendido o procedimento ILS retorna a melhor solução encontrada até o momento (linha 11).

2.5 Perturbação

A técnica de perturbação facilita a execução do método de busca local ajudando a atingir opções de vizinhança distintas por meio da alteração da solução atual, gerando assim uma nova solução para que se percorra seu espaço de busca. Ribeiro, Arroyo e Santos (2012), desenvolvem o procedimento de perturbação de diferentes formas, entre elas a perturbação por inclusão que consiste em inserir vários nós ao mesmo tempo e a Perturbação por exclusão este compreende a remoção de nós nas rotas.

A aplicação da perturbação ocorre antes que solução seja enviada para o método de busca local, esse processo se caracteriza em um ciclo cuja melhor

solução passará pelo método de perturbação, e seu resultado pela busca local até que não haja mais melhorias na solução dentro da vizinhança selecionada. A Figura 2 apresenta a aplicação da perturbação em um espaço de solução.

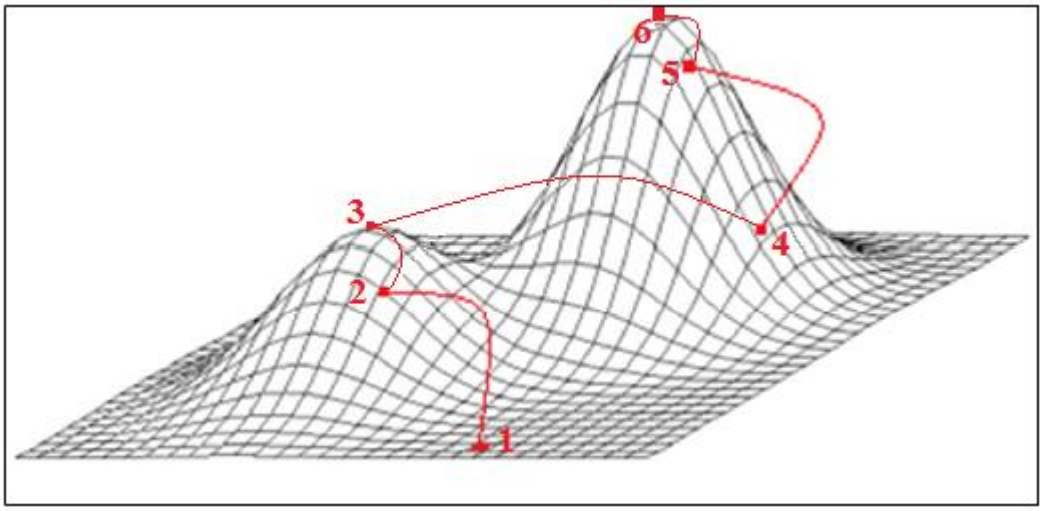

Figura 2 - Espaço de Solução orientado pela Função Objetivo.
Fonte: O próprio autor

A Figura 2 apresenta o espaço solução de um dado problema. O passo 1 marca a solução inicial após sofrer o primeiro processo de perturbação. A solução é melhorada por meio da Busca Local passando pelo passo 2 até chegar no Ótimo Local (Passo 3). A partir desta vizinhança é impossível melhorar a solução. Novamente aplica-se a perturbação com o objetivo de encontrar outras áreas do espaço de solução. A partir do passo 4 inicia-se o processo de busca local passando pelo passo 5 até a identificação do Ótimo Global no passo 6.

2.6 Métodos de definição de vizinhança

Segundo Cordenonsi (2008), a expressão "vizinhança" esta relacionada com os caminhos que estão perto um do outro, dentro do espaço de busca de soluções, estes podem ser atingidos por meio de técnicas de busca.

Os métodos de definição de vizinhança são fundamentais devido a sua concepção, pois podem sintetizar pontos de partidas utilizados por outros tipos de algoritmos. Métodos como estes são caracterizados pela simplicidade das alterações no roteiro e podem ser aplicados no PCV com finalidade de obter respostas mais qualificadas (FARIAS, GOLDBARG e LUNA, 2008).

Mine, Ochi e Souza (2009) explicam que a definição de vizinhanças é dada por movimentos dos elementos da solução tendo como alvo o aperfeiçoamento da rota encontrada original. Segundo Junior, Fernandes e Medeiros (2002), os algoritmos de Busca Local consistem em transformações simples no roteiro, estes buscam efetuar trocas entre os caminhos a fim de minimizar o tamanho de um circuito até que não haja mais condição de melhorias.

A definição de vizinhanças adequadas para o problema abordado permite representar de forma compacta e eficiente o conjunto de vizinhos da solução corrente. Dentre os diversos métodos de definição de vizinhanças, os mais dicutidos na literatura científica são as estruturas k-Opt. Neste trabalho, as estruturas de vizinhanças selecionadas são ADD/DROP, 2-Opt e 3-Opt.

2.6.1 ADD/DROP

Nogueira (2007) explica o ADD/DROP como heurísticas construtivas ou de refinamento, baseadas na inserção e remoção de pontos no roteiro, que podem ser usadas em problemas de otimização de rotas.

Conforme Silva e Sanches (2012), o movimento ADD se baseia na adição do vértice que tem o valor superior de economia de inserção e o método DROP retira o vértice com melhor valor de economia de remoção, esses algoritmos são executados até que não haja viabilidade para otimizar a solução. O custo de economia reflete na função objetivo, pois se este obtiver valor positivo então a função objetivo terá resposta satisfatória após a execução do movimento, é possível analisar que todos os movimentos possuem o objetivo de alcançar a factibilidade da solução (CHAVES e LORENA, 2007).

Segundo Nogueira (2007), as heurísticas ADD e DROP são sustentadas por fundamentos algorítmicos iguais, onde a cada repetição é efetuada a localização ótima de somente um ponto no roteiro, isso acontece por meio da ordenação de todas as possíveis opções de escolha.

As heurísticas ADD/DROP são utilizadas para verificar as possibilidades de melhorar a solução obtida e dessa forma, proporcionar o encontro de soluções que sejam mais satisfatórias.

2.6.2 2-Opt

Segundo Júnior, Fernandes e Medeiros (2002), a heurística 2-Opt foi proposta primeiramente por Flood em 1956 e por Croes em 1958. O método 2-Opt verifica as possibilidades de troca em pares de arcos. Dessa forma, as arestas são excluídas, fragmentando o circuito em dois caminhos, as junções destes caminhos são feitas de outras formas, buscando o aperfeiçoamento do roteiro. O término deste procedimento só acontece quando não houver mais opões de melhoria (ABRAHÃO e GUALDA, 2004). A alteração do roteiro só acontece se as trocas dos arcos proporcionarem a diminuição da distância no roteiro.

A heurística 2-Opt utilizada por Vieira (2006) foi composta pelos seguintes passos: adoção de uma solução inicial com matrizes de custo estático e custo variável, logo após foi selecionada duas arestas, estas não podem pertencer ao mesmo vértice. Posteriormente, analisam-se os valores obtidos por meio das trocas realizadas entre as arestas, verificando se a função objetivo foi minimizada ou não. O passo de troca é feito até que não exista mais a possibilidade de aperfeiçoar a rota. A Figura 3 mostra um exemplo de movimentos que podem ser realizados pela heurística 2-Opt.

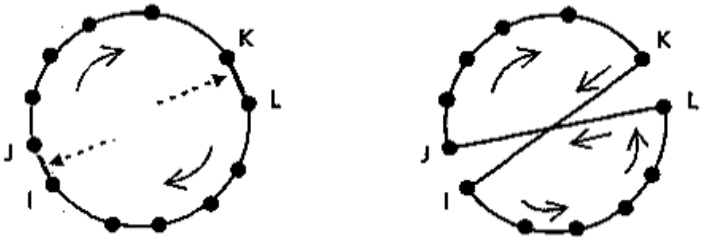

Figura 3 - Esquema do Movimento 2-Opt.Variáveis K, L, I e J representando os pontos de troca entre as arestas.
Fonte: O próprio autor

De acordo com a Figura 3, após eleitas 2 arestas para troca, (*j, i*) e (*k, l*), desliga-se os vértices envolvidos e os religa-os novamente em um arranjo contrário, (*j, l*) e (*k, i*). Após o passo de troca entre as arestas, a nova solução é reavaliada pela Função Objetivo identificando se obteve o aprimoramento da solução.

2.6.3 3-Opt

O método 3-Opt funciona de forma semelhante ao 2-Opt, onde é analisada a troca de três arcos. A reconexão desses arcos pode ocorrer de duas formas

distintas, porém o movimento só é realizado se uma dessas maneiras minimizar o tamanho do roteiro. Caso as duas formas proporcionem a diminuição do circuito, será escolhida aquela que possibilitar maior redução do percurso (JUNIOR, FERNANDES E MEDEIROS, 2002). A Figura 4 representa um exemplo das movimentações de rotas que podem ser feitas pela heurística 3-Opt.

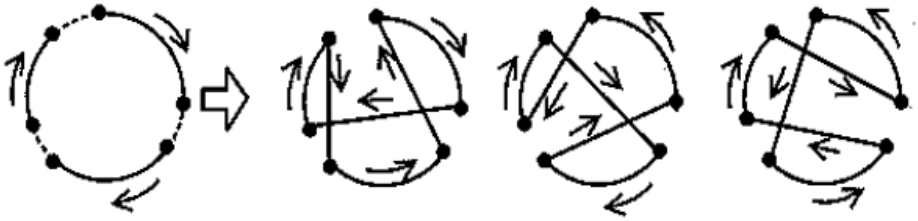

Figura 4 - Esquema do Movimento 3-Opt.
Fonte: O próprio autor

De acordo com a Figura 4, partindo de uma solução válida, método 3-Opt seleciona 3 vértices e fragmenta a solução em 3 pedaços. Após isto, as arestas são reconectadas buscando novas variantes da solução original. Após a avaliação de cada nova solução por meio da Função Objetivo, a solução de menor distância é eleita para continuar o processo.

2.7 Sistemas Distribuídos

Sistemas distribuídos envolvem conceitos relacionados a redes de computadores, meio pelo qual é facilitada a comunicação estabelecida entre as máquinas envolvidas em um determinado processamento. Sistemas distribuídos podem ser entendidos como uma coleção de componentes conectados por um sistema de comunicação, possuindo a capacidade de processar informações. Este conceito é, muitas vezes, confundido com redes de computadores cujo principal objetivo é proporcionar o compartilhamento de recursos, diferente dos sistemas distribuídos, que são criados para aumentar o desempenho e confiabilidade de determinado processo (TANENBAUM, 2003).

A implementação de sistemas distribuídos depende da viabilidade dos recursos e dos custos requisitados, pois os benefícios pode não compensar os gastos. Segundo Coulouris, Dollimore e Kindberg (2007) existem alguns desafios a serem enfrentados para a construção de um sistema distribuído qualificado, são eles:

a) Heterogeneidade: Está relacionada à incompatibilidade de alguns componentes que são utilizados na implementação de sistemas distribuídos, entres os componentes pode-se citar hardware de computadores, sistemas operacionais, redes, linguagens de programação, etc. A heterogeneidade pode ser resolvida por meio da utilização de plataformas intermediárias denominadas *Middleware*;
b) Transparência: Consiste em tornar invisível alguns dos aspectos da distribuição que compõe os sistemas, tais como acessibilidade, localização, concorrência entre processos, ocultação de falhas, entre outros;
c) Escalabilidade: compreende a capacidade do sistema em manipular uma quantidade crescente de serviços ou estar apto para evoluir e agilizar os processamentos;
d) Segurança: Em sistemas distribuídos as informações se tornam mais vulneráveis, ocasionando a necessidade de utilizar mecanismos que protejam a informação trafegada na rede, tais como: assinatura digital, certificado digital e técnicas de criptografia;
e) Tratamento a falhas: refere-se reconhecimento das possíveis falhas que podem ocorrer no sistema e tratá-la de forma apropriada.
f) Concorrência: acontece devido à solicitações simultâneas de serviços por meio dos usuários. Dessa forma, os componentes do sistema devem estar prontos para manter a consistência dos dados requisitados.
g) Sistemas abertos: são caracterizados por permitirem facilidades para a sua ampliação ou extensão. Suas principais características são a interoperabilidade, portabilidade e extensibilidade.

Um projeto de sistemas distribuídos contém alguns aspectos relevantes que contribuem para a distribuição das responsabilidades entre os componentes que o compõem. Segundo Coulouris, Dollimore e Kindberg (2007) esses aspectos se referem ao tipo de estrutura utilizada para estabelecer a comunicação entre os processos ou entre os elementos envolvidos. Sendo assim, a maneira escolhida para compor a estrutura de distribuição pode influenciar no desempenho, na segurança e na confiabilidade do sistema desenvolvido. O autor aborda dois dos

principais modelos de arquiteturas, sendo eles a arquitetura *Peer-To-Peer* e a Cliente-Servidor.

Segundo Candido e Ferreira (2002) arquitetura Cliente-Servidor é caracterizada pela centralização de serviços de redes fornecidos de uma estação ou conjuntos de estações para as outras estações envolvidas no processamento. Os servidores são representados pelas máquinas que realizam o serviço e os clientes solicitam os serviços. Seu funcionamento ocorre da seguinte forma: O processo cliente faz uma requisição (*request*) para o servidor, este processa a informação e retorna (*response*) para ao cliente o resultado obtido

Já a arquitetura *Peer-To-Peer* é composta por nós, sendo que esses podem assumir tanto o papel de servidor quanto o de cliente. Os dados são distribuídos entres os componentes permitindo o equilíbrio de carga das informações. De acordo com Coulouris, Dollimore e Kindberg (2007), o modelo *Peer-To-Peer* visa a realização de tarefas por meio da utilização dos recursos disponibilizados por um conjunto de computadores. Entres as vantagens e desvantagens proporcionadas por este tipo de arquitetura, pode-se especificar como benefícios o equilíbrio das informações e a menor ocorrência de falhas. Porém o tempo de resposta pode ser desfavorável devido a necessidade de averiguar todos os componentes envolvidos, pois as informações estão dispersas entre os mesmos.

3. METODOLOGIA

Nesta seção serão descritos os aspectos metodológicos dos modelos propostos para solucionar o problema abordado, discutindo a arquitetura para distribuição da heurística ILS.

3.1 Formulação do problema

Os métodos exatos proporcionam a obtenção de soluções ótimas, porém exige alto tempo computacional e permitem a utilização de instâncias com poucos nós. Já os chamados métodos heurísticos ou métodos aproximados, não asseguram que as respostas encontradas sejam ótimas, mas permitem que as instâncias

utilizadas tenham maior quantidade de nós e o custo computacional requisitado seja menor (BLUME, 2012). Entretanto, a utilização de métodos aproximados com instâncias maiores não garante a rapidez no processo de obtenção da melhor solução, ou seja, a execução da heurística fica lenta.

Um dos recursos utilizados para minimizar a lentidão na execução das heurísticas é a computação com alto desempenho, conforme Duncan (1990), uma arquitetura paralela proporciona a confecção de soluções por meio do processamento paralelo, este se concretiza com a utilização de processadores que auxiliam a resolução de problemas por meio da execução concorrente.

3.2 Definição da instância de dados

Segundo Blume (2012), existem várias instâncias para o PCV, estas são originadas a partir de diferentes condições, desde situações reais até situações fictícias. O autor explica que os problemas que mais atraem atenção, são aqueles baseados no problema euclidiano, no qual as cidades são equivalentes a pontos no plano, nestes casos as instâncias são euclidianas e possuem natureza geométrica, permitindo a aplicação em situações práticas.

As instâncias utilizadas neste trabalho foram fornecidas pela biblioteca TSPLIB e identificadas como gr96, pr226, linh318, compreendendo 96, 226 e 318 pontos, respectivamente. Reinelt (1991), explica que TSPLIB, consiste em uma biblioteca composta por diversas instâncias do PCV e também de outros problemas relacionados.

3.3 Modelo não distribuído proposto

O modelo não distribuído proposto neste trabalho consiste na execução do algoritmo ILS em somente uma máquina, sem a utilização dos aspectos da programação distribuída. A implementação do algoritmo baseado no pseudocódigo padrão apresentado no Quadro 1, sucedeu da seguinte forma: (a)Inicialmente gera-se uma solução inicial aleatória; (b) invocação do procedimento de perturbação responsável por modificar a solução atual promovendo uma solução intermediária; (c) invocação do método de Busca Local, onde busca-se por meio de trocas sistemáticas de vizinhança, aperfeiçoar a solução original até o limite da vizinhança

adotada; (d) logo após, aplica-se o critério de aceitação, definindo a próxima solução a ser perturbada.

A qualidade das respostas obtidas pelo ILS está relacionada com as perturbações aplicadas às soluções, por isso é preciso medir a perturbação imposta em uma solução a fim de que as modificações sejam suficientes para fugir dos ótimos locais (RIBAS, 2011).

Todo o processo é repetido até satisfazer o critério de parada. A Figura 5 apresenta o modelo não distribuído, e a sequência de execução do algoritmo em um computador.

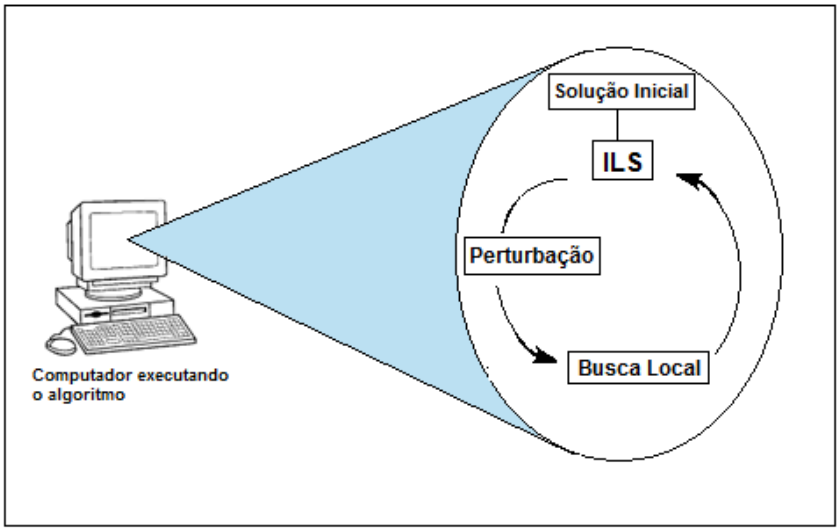

Figura 5 - Modelo não distribuído proposto.Modelo de execução do ILS em um computador. Todas as tarefas são executadas sequencialmente.
Fonte: O próprio autor

O modelo não distribuido proposto e apresentado na Figura 5, demonstra as etapas do algoritmo ILS sendo executadas sequencialmente em um único computador. Esta execução durará até que os critérios de parada sejam atingidos.

3.4 Modelo distribuído proposto

O modelo distribuído proposto consiste na execução do ILS com utilização da programação distribuída, no qual teve a distribuição da execução entre *n* máquinas remotas. O meio pelo qual foi facilitada a distribuição do processamento em rede é o socket TCP, viabilizando o compartilhamento dos recursos computacionais.

A solução desenvolvida implementa um gerenciador de execuções. Este é responsável por controlar as execuções entres as máquinas remotas. As máquinas

remotas assumiram o papel do servidor que possui a função de executar os algoritmos de Busca Local e retornar a solução para o gerenciador de execuções.

A Figura 6 expõe o modelo distribuído, com a finalidade de proporcionar maior compreensão dos procedimentos realizados.

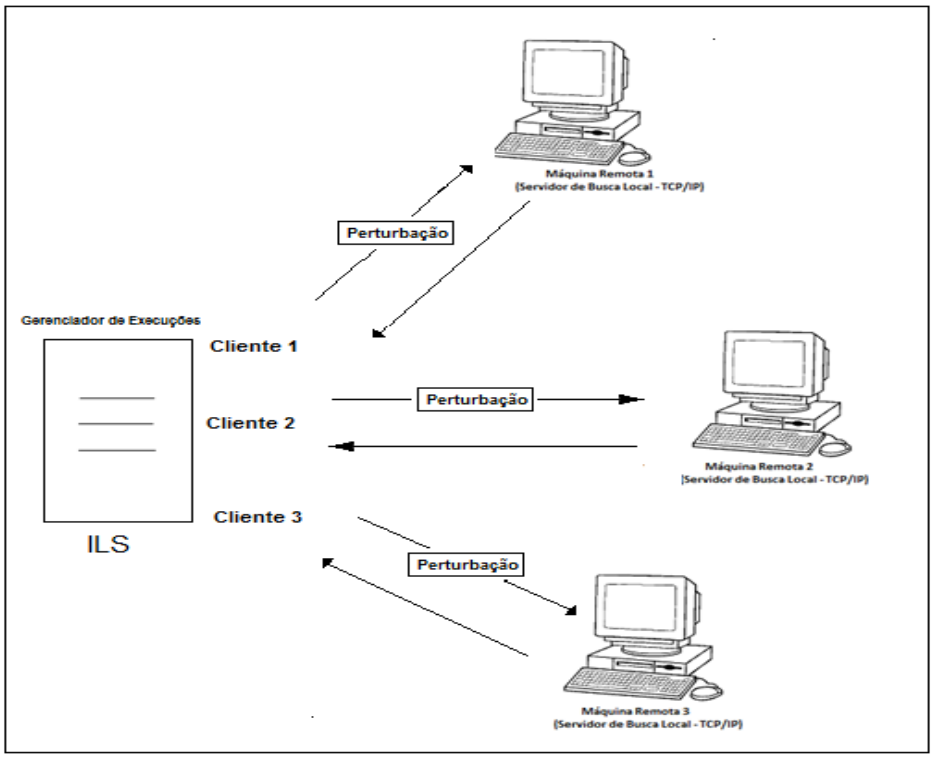

Figura 6 - Modelo distribuído.
Fonte: O próprio autor

A Figura 6 apresenta o modelo de execução do algoritmo distribuído. A execução do algoritmo distribuído segue cinco passos: (a) o servidor ILS gera a solução inicial para a execução; (b) o gerenciador de execuções cria os clientes para conexão com as máquinas remotas; (c) o gerenciador de execuções gera uma lista de soluções perturbadas de acordo com o número de máquinas remotas envolvidas no processo; (d) cada máquina remota recebe uma solução perturbada e processa a Busca Local retornando a melhor solução para o gerenciador de execuções; (e) o gerenciador de execuções compara as soluções geradas e aplica o critério de aceitação definindo a melhor solução até o momento. Após a avaliação do critério de aceitação, o processo se inicia até o critério de parada ser atendido.

3.4 Execução dos algoritmos

A heurística ILS foi executada 10 vezes para cada instância de dados utilizando o algoritmo não distribuído e o algoritmo distribuído com dez, quinze e vinte máquinas remotas, identificadas respectivamente como ND, D10MR, D15MR e D20MR. Os resultados obtidos com a execução compreendem o valor da função objetivo e o tempo consumido para encontrar a melhor solução.

Os microcomputadores utilizados durante os testes possuem as seguintes configurações: processador Intel core i5 3.00 GHz (4 CPUS), 4096 RAM, sistema operacional Windows 7 Professional 64 bits.

4. RESULTADOS E DISCUSSÕES

O modelo não distribuído se caracteriza pela utilização de apenas um computador, dessa forma o processamento é centralizado usufruindo dos recursos fornecidos por apenas uma máquina, na qual o algoritmo está sendo executado.

Já o modelo distribuído proposto para o PCV determina a distribuição do processamento do algoritmo em números distintos de máquinas remotas, possibilitando o compartilhamento dos recursos disponibilizados pelos computadores durantes a execução.

4.1 Instância gr96

Os resultados obtidos por meio da execução do algoritmo, com a instância de 96 pontos utilizando respectivamente o algoritmo não distribuído e o algoritmo distribuído com 10, 15 e 20 máquinas remotas são apresentados na Tabela 1. O tempo computacional para cada algoritmo encontrar a melhor solução está apresentado em milésimos de segundo.

N °	Instância gr96							
	ND		D10MR		D15MR		D20MR	
	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)
1	547,75	1402	526,2	118	535,99	48	534,03	55
2	545,19	1221	539,42	79	536,99	88	533,97	12

3	527,55	96	544,96	99	539,16	20	541,29	67
4	538,79	424	542,29	44	532,15	56	534,1	63
5	542,29	162	539,23	48	538,37	25	534,74	9
6	533,63	220	542,87	90	533,45	51	540,22	5
7	531,56	102	542,87	128	536,7	14	536,86	86
8	548,59	264	534,17	39	535,57	87	535,99	17
9	544,96	200	538,05	24	541,53	11	529,64	34
10	541,29	192	537,05	29	533,49	29	533,18	54

Tabela 1 - Resultado das execuções com a instância gr96.
Fonte: O próprio autor

A Tabela 1 apresenta a comparação entre a execução do algoritmo não distribuído e o algoritmo distribuído em que é possível observar a melhora progressiva, não apenas do tempo computacional, mas também da Função Objetivo (FO) para a instância gr96. A FO melhorou na em média 0,62%. A Figura 7 apresenta o gráfico comparativo entre as médias de tempo das execuções da instância gr96.

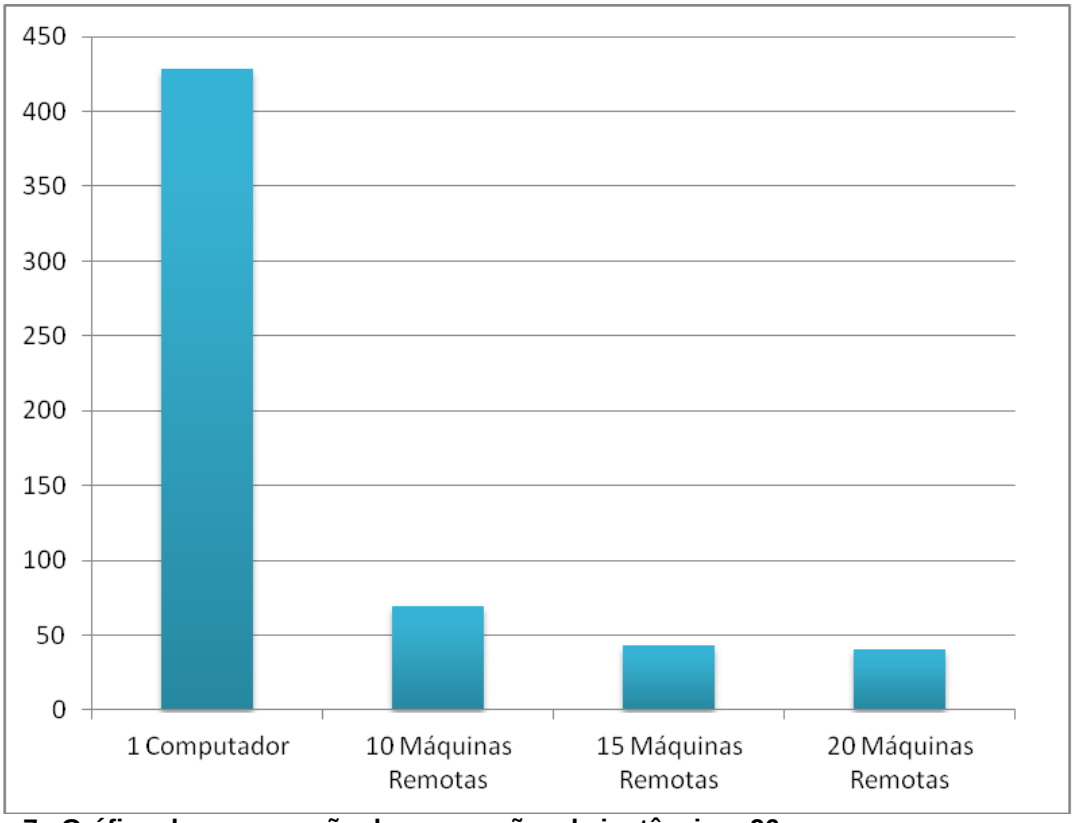

Figura 7 - Gráfico da comparação das execuções da instância gr96.
Fonte: O próprio autor

A média alcançada por meio da execução do algoritmo com a instância gr96 utilizando o algoritmo não distribuído foi de 428,3 milissegundos. Já as médias dos

resultados encontrados com a utilização do processamento distribuído foram de 69,8 milissegundos com 10 máquinas remotas, 42,9 milissegundos utilizando 15 máquinas remotas e por fim 40,2 milissegundos com vinte máquinas.

4.2 Instância pr226

A Tabela 2 apresenta os resultados obtidos pela execução dos algoritmos para a instância pr226.

N °	Instância pr226							
	ND		D10MR		D15MR		D20MR	
	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)
1	87403,27	13578	87728,8	1092	87428,32	9044	87936,73	1033
2	87490,58	8415	87885,34	2644	87923,44	1103	88083,86	910
3	87957,37	7773	87852,67	8637	87969,78	891	88003,71	1517
4	87712,65	4339	87574,09	3824	87888,08	1418	87522,85	1262
5	87988,29	10846	87592,48	1182	87980,16	4844	87978,08	2018
6	87795,14	10855	87980,16	6407	87890,56	274	87716,46	858
7	87662,66	10978	87982,78	3356	88086,8	1197	87696,15	1502
8	87873,74	10220	87244,95	3110	87879,22	302	87914,43	433
9	88452,43	16889	87110,66	6377	87683,13	3670	87735,74	1107
10	87795,14	7165	87809,07	6704	87866,79	1230	87894,67	3224

Tabela 2 - Resultado das execuções com a instância pr226.
Fonte: O próprio autor

A comparação entre a execução do algoritmo não distribuído e o algoritmo distribuído para a instância pr226 apresentado na Tabela 2, descreve uma melhora média de 0,02% na FO. A Figura 8 apresenta o gráfico comparativo entre as médias de tempo das execuções da instância pr226.

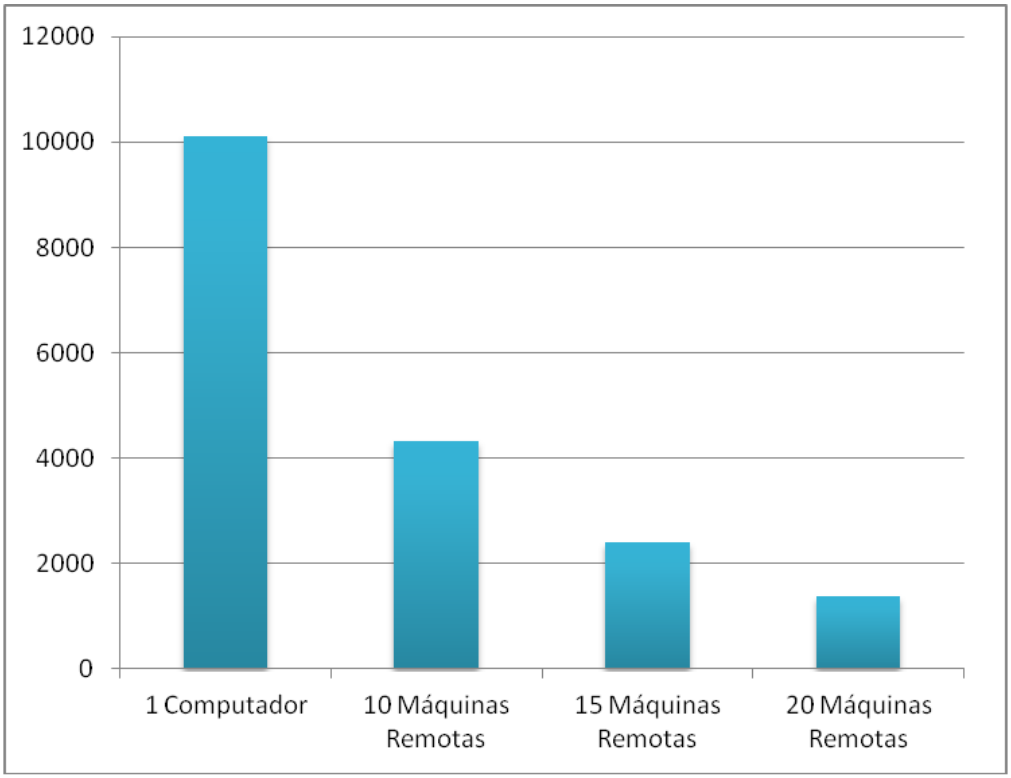

Figura 8 - Gráfico da comparação das execuções da instância pr226.
Fonte: O próprio autor

A média do tempo computacional obtida com a execução do algoritmo utilizando a instância pr226 com apenas uma máquina foi de 10105,8 milissegundos. Porém, as médias dos tempos alcançadas utilizando dez, quinze e vinte máquinas remotas foram respectivamente 4333,3 milissegundos, 2397,3 milissegundos e 1386,4 milissegundos.

4.3 Instância linh318

A Tabela 3 apresenta os resultados obtidos pela execução dos algoritmos para a instância linh318.

N °	Instância linh318							
	ND		D10MR		D15MR		D20MR	
	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)
1	49889,97	12110	49602,88	2283	49997,58	628	49955,97	3220
2	49740,68	10043	49585,8	11485	49857,07	2541	49837,01	1510

3	49906,54	9748	49749,6	7183	49658,64	2141	49737,02	638
4	49667,56	9951	49417,4	5917	49507,67	1767	49767,45	2616
5	49882,66	10218	49079,91	1337	49665,63	1489	49602,94	1681
6	49982,06	12197	49436,89	6403	49804,63	2495	49738,23	1267
7	49779,96	12445	49295,25	2616	49794,27	3114	49744,9	1873
8	50055,95	11200	49283,22	6723	49839,63	1942	49808,68	684
9	50182,56	6043	49584,73	7062	49805,64	1973	49862,92	1582
10	49740,18	7921	49390,16	4276	49896,67	1171	49905,3	1238

Tabela 3 - Resultado das execuções com a instância linh318.
Fonte: O próprio autor

A Tabela 3 apresenta a comparação entre a execução do algoritmo não distribuído e o algoritmo distribuído para a instância linh318. É possível observar uma melhora média de 0,42% na FO. A Figura 9 apresenta o gráfico comparativo entre as médias de tempo das execuções da instância linh318.

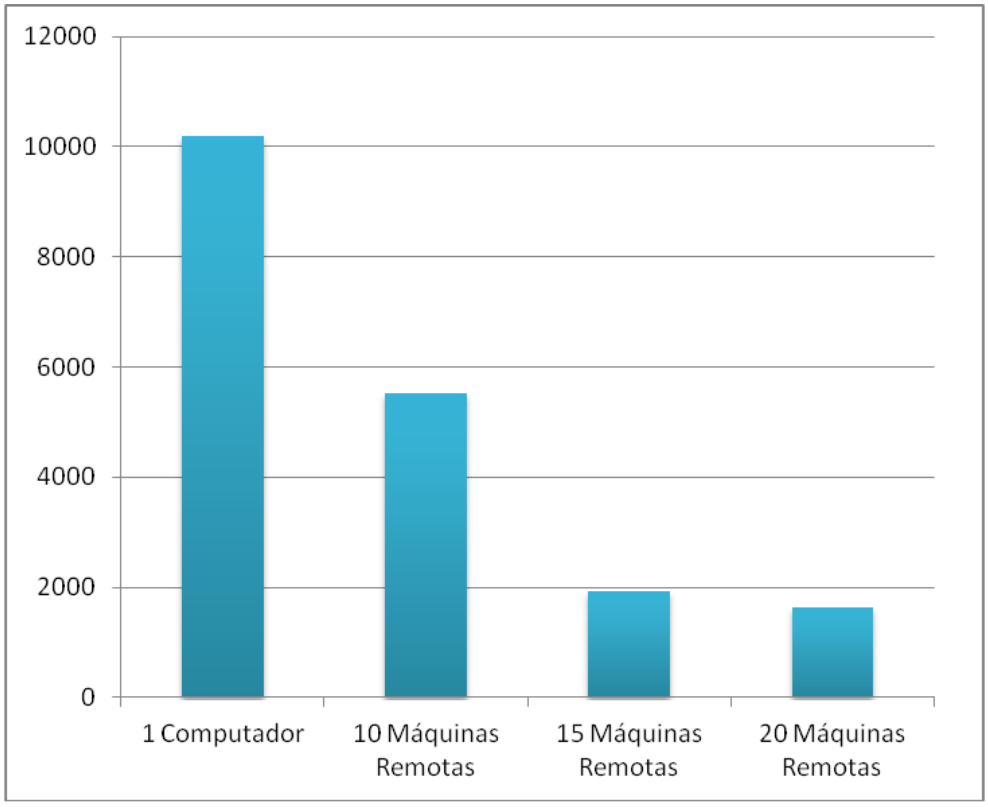

Figura 9 - Gráfico da comparação das execuções da instância linh318.
Fonte: O próprio autor

A partir dos resultados da execução do algoritmo utilizando a instância linh318, a média obtida foi 10187,6 milissegundos com um computador. As médias encontradas utilizando dez, quinze e vinte máquinas remotas foram

respectivamente, 5528,5 milissegundos, 1926,1 milissegundos e 1630,9 milissegundos.

4.4 Comparação entre os resultados das instâncias

A partir das soluções encontradas é perceptível o ganho computacional alcançado na execução do algoritmo distribuído. Isto pode ser verificado por meio das Figuras 7, 8 e 9 que apresentam os gráficos das médias dos resultados encontrados. Ao relacionar os tempos computacionais obtidos pelo modelo não distribuído e modelo distribuído é possível observar que existem grandes melhorias. A Tabela 4 apresenta as médias dos valores encontrados para a Função Objetivo e para o tempo computacional encontrado nas execuções realizadas com as três instâncias utilizando uma, dez, quinze e vinte máquinas remotas.

Máquinas Remotas	Instância gr96		Instância pr226		Instância linh318	
	FO	TEMPO (ms)	FO	TEMPO (ms)	FO	TEMPO (ms)
1	540,16	428,3	87813,127	10105,8	49882,812	10187,6
10	538,711	69,8	87676,1	4333,3	49442,584	5528,5
15	536,34	42,9	87859,628	2397,3	49782,743	1926,1
20	535,402	40,2	87848,268	1386,4	49796,042	1630,9

Tabela 4 - Média dos resultados das execuções dos algoritmos.
Fonte: O próprio autor

A execução da instância gr96 com dez, quinze e vinte máquinas remotas proporcionou melhorias em relação aos resultados encontrados utilizando apenas um computador. A porcentagem de ganhos executando o algoritmo em dez, quinze e vinte máquinas foram respectivamente, 83,71%, 89,99% e 90,62%.

O modelo distribuído proporciona melhorias significativas dos tempos computacionais. Dessa forma, para a instância pr226, os ganhos de tempo alcançados na execução do algoritmo foram de 57,13% com dez máquinas remotas, 76,28% utilizando 15 máquinas remotas e 86,29% com vinte máquinas remotas.

A melhoria obtida com a execução da instância linh318, comparando os resultados com um computador, foi de 45,74% para dez máquinas remotas, 81,10% com quinze máquinas remotas e 84% utilizando vinte máquinas remotas.

4.5 Comparação com trabalhos relacionados

A Tabela 5 apresenta uma comparação entre os trabalhos relacionados que foram descritos no item 2.1.

Autores	Problema	Heurísticas	Técnicas de Processamento	Instâncias	Resultados
Blume (2012)	PCV	Simulated Annealing (SA)	Programação paralela com MPI (Message Passing Interface)	Entre 100 e 200 pontos	Soluções ótimas a partir do aumento do número de processos.
Gazola (2010)	PCV	Algoritmos de Busca local (2-Opt, swap, oropt -1,oropt -2, oropt -3)	Processamento em GPU (Graphics Processing Unit)	Entre 100 e 1002 pontos	Tempos computacionais favoráveis e qualidade da solução razoável.
Medeiros (2012)	PRVC (Problema de roteamento de Veículos Capacitados)	Cluster First Route Second e 2-Opt	LINQ (Lanquage Integrated Query)/classe denominada Parallel da .NET Framework 4	Entre 32 e 80 pontos	Resultados satisfatórios para problemas mais complexos.
Santos et al. (2006)	PCV Assimétrico	Método do "Vizinho mais Próximo" e 3-Opt	Arquitetura Cluster	Entres 16 e 1001 pontos	Soluções de qualidade e obtenção de muitos benefícios com o processamento paralelo.

Tabela 5 - Comparação entre trabalhos relacionados.
Fonte: O próprio autor

Observa-se que os autores mostrados na Tabela 5 são unânimes em afirmar as vantagens da utilização de técnicas de programação distribuída para o processamento de grandes volumes de dados. Dentre os trabalhos apresentados é importante ressaltar que apenas os estudos de Gazola (2010) e Santos et al. (2006) utilizam instâncias maiores que 300 pontos. Isto demonstra a grande complexidade dos problemas de roteamento de veículos além da confecção de rotinas previamente otimizadas para este fim.

5. CONSIDERAÇÕES FINAIS

Este trabalho abordou a problemática do processamento de uma heurística ILS para a resolução de Problemas do Caixeiro Viajante com instâncias de 96, 226 e 318 pontos, sendo estas disponibilizadas por uma biblioteca denominada TSPLIB. Para a resolução destes problemas utilizou-se técnicas de programação distribuída a fim de reduzir o tempo computacional para a solução do problema. Após a execução dos algoritmos os resultados foram comparados verificando as melhorias obtidas por meio da metodologia distribuída. A partir da análise dos resultados obtidos pode-se observar ganhos nos tempos computacionais durante a execução de todas as instâncias, sendo que as melhorias variam de acordo com o número de máquinas utilizadas no processamento.

Para instâncias de 96 pontos, a evolução da melhoria do tempo computacional ocorre na ordem de 83,71% para 10 máquinas remotas, 89,99% para 15 máquinas remotas e 90,62% para 20 máquinas na comparação com a execução do algoritmo não distribuído. Com a instância de 226 pontos, o progresso do tempo computacional foi de 57,13% para 10 máquinas remotas, 76,28% para 15 máquinas remotas e 86,29% para 20 máquinas comparando os resultados com o algoritmo não distribuído. Finalmente, para a instância de 318 pontos, a melhoria do tempo de execução foi em torno de 45,74% para 10 máquinas remotas, 81,10% para 15 máquinas remotas e 84% para 20 máquinas remotas na comparação com o algoritmo não distribuído.

Observa-se que para a instância gr96 houve uma menor variação em relação às porcentagens dos tempos computacionais e a execução dos algoritmos distribuídos, sendo que quanto maior o número de máquinas menos perceptível é a melhoria. Já a instância pr226 obteve ganhos menos discretos dentre os algoritmos distribuídos, sendo que a comparação com o algoritmo não distribuído se apresentou muito eficiente. De acordo com os resultados, utilizando a instância linh318, obteve-se maior porcentagem de melhoria de tempo utilizando-se 15 máquinas remotas.

Os resultados computacionais apresentados neste trabalho são conclusivos em apontar grandes melhorias para o processamento de heurísticas ILS e grandes

instâncias de dados. A contribuição deste trabalho é significativa a partir do momento que expõe um modelo para implementação de heurísticas distribuídas baseadas em trocas sistemáticas de vizinhanças variáveis e associadas à busca local.

Como trabalhos futuros, propõe-se a utilização de instâncias com números de pontos superiores a 540 e um modelo distribuído com 20, 40 e 60 máquinas remotas.

REFERÊNCIAS

BORMUTH, R. J. Cloze test readability: Criterion reference scores. *Journal of Educational Measurement*, 1968. 5, p. 189-196. Disponível em: http://dx.doi.org/10.1111/j.1745-3984.1968.tb00625.x BRASIL. Ministério da Educação. *Relatório Nacional PISA 2000*. Brasília, 2001, 88p.

ABRAHÃO, F. T. M.; GUALDA, N. D. F. Aplicação da meta-heurística colônias de formigas e das heurísticas 2-opt e 3-opt na solução de problemas logísticos da força aérea brasileira. **XXVI Simpósio Brasileiro de Pesquisa Operacional. Anais...**, São João Del Rei, MG, p. 856-866, 2004.

ANDRADE, G. P.; SIQUEIRA, E. C.; SOUZA, S. R. Meta-heurísticas grasp, ils e iterated greedy aplicadas a problemas de agendamento de cirurgias eletivas em hospitais de grande porte. **XLIV Simpósio Brasileiro de Pesquisa Operacional. Anais...**, Rio de Janeiro, p. 2624-2635, 2012.

BECCENERI, J. C. Computação e Matemática Aplicada às Ciências e Tecnologias Espaciais. Meta-heurísticas e Otimização Combinatória: Aplicações em Problemas Ambientais**. INPE, São José dos Campos**, p.65-81, 2008. Disponível em < http://www.lac.inpe.br/ELAC13/arquivos/MiniCurso_02ELAC2012.pdf>. Acesso em: 20/12/2013

BLUME, E. Algoritmo de otimização paralelo: Um Modelo Proposto E Implementado. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Florianópolis, 2012.

BRITO, J. A. M.; MONTENEGRO, F. M. T.; OCHI, L. S. Um algoritmo ILS para melhoria de eficiência da estratificação estatística. **XLI Simpósio Brasileiro de Pesquisa Operacional. Anais...** Porto Seguro, p. 2133-2144, 2009.

CÂNDIDO, G.; FERREIRA M.; C. T. Estudo De Protocolos Em Rede – Uma Aplicação Com Socket. Teste (Doutorado) – Universidade Estadual de Mato Grosso do Sul, 2002.

CHAVES, A. A.; LORENA, L. A. N. Aplicação do Algoritmo Clustering Search aos Traveling Salesman Problems with Profits. **XXXIX Simpósio Brasileiro de Pesquisa Operacional. Anais...** Fortaleza, CE, p.1472-1483, 2007.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T. **Sistemas distribuídos: conceitos e projeto.** Bookman Editora, 2007.

CONSTANTINO, A. A.; JUNIOR, A. M. G. Um algoritmo genético híbrido para o problema de corte industrial bidimensional.**Acta Scientiarum. Technology**, Maringá, v. 24, n. 6, p. 1727-1731, 2002. ISSN 1807-8664.

CORDENONSI, A. Z. Ambientes, objetos e dialogicidade: uma estratégia de ensino superior em heurísticas e meta-heurísticas. Teste (Doutorado) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2008.

COSTA, C. S. A. G. Problema do Caixeiro Viajante – Resolução e Depuração. Dissertação (Mestrado) - Universidade de Aveiro, Portugal, 2008.

CUNHA, C. B. Aspectos práticos da aplicação de modelos de roteirização de veículos a problemas reais. Transportes, v. 8, n. 2, 2000. Disponível em <http://sites.poli.usp.br /ptr/ptr/docentes/cbcunha/files/roteirizacao_aspectos_praticos_CBC.pdf>. Acesso em 20/12/2013.

CUNHA, C. B.; BONASSER, U. O.; ABRAHÃO, F. T. M. Experimentos computacionais com heurísticas de melhorias para o problema do caixeiro viajante. In: **XVI Congresso da Anpet**, 2002.

DUNCAN, R. A survey of parallel computer architectures. **Computer**, v.23 , n. 2, p. 5-16, 1990. DOI: 10.1109/2.44900.

FARIAS, M. S. R.; GOLDBARG, M. C.; LUNA, H. P. L. Aplicação de um algoritmo evolucionário de busca de soluções de pareto para o problema do caixeiro viajante objetivo. **XL Simpósio Brasileiro de Pesquisa Operacional. Anais...** João Pessoa, PB, p.905-913, 2008.

GAZOLLA, J. G. F. M. Uma abordagem heurística e paralela em GPUs para o problema do caixeiro viajante. Dissertação (Mestrado) - Universidade Federal Fluminense, Niterói, RJ, 2010. Disponível em < http://www2.ic.uff.br/PosGraduacao/ Dissertacoes/472.pdf>. Acesso em 20/12/2013.

GOLDBARG, M. C.; LUNA, H. P. L. **Programação linear e otimização combinatória: modelos e algoritmos.** Rio de Janeiro: Campus, v. 1, p. 639, 2000.

JUNIOR, D. O. M.; FERNANDES, E. L. R.; MEDEIROS, M. L. **Problema do Caixeiro Viajante.** Universidade Federal de Mato Grosso do Sul. 2002. Disponível em < http://www.eraldoluis.pro.br/lib/exe/fetch.php/2002.1-tsp.pdf>. Acesso em 20/12/2013.

TANENBAUM, A. S. **Redes de Computadores.** São Paulo: Ed. 2003.

MEDEIROS, W. J. N. Abordagem heurística paralelizada para a resolução do problema de roteamento de veículos capacitados com base na estratégia “cluster first route second”. Trabalho de Conclusão de Curso, Curitiba, PR, 2012. Disponível em<http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/380/1/CT_EPC_2011_2_16.PDF>. Acesso em 20/12/2013.

MICHALEWICZ, Z. **Genetic algorithms + data structures = evolution programs.** Springer, Berlin,1996.

MINE, M. T.; SILVA, M. S. A.; OCHI, L. S.; SOUZA, M. J. F. Um algoritmo heurístico híbrido para o problema de roteamento de veículos com entrega e coleta simultânea. **XXIX Encontro Nacional de Engenharia de Produção.** Salvador, BA, 2009.

NETO, E. P.; VIANNA, D. S. Heurísticas eficientes para o problema de geração de grade escolar automatizada**. Revista Eletrônica Produção & Engenharia**, v. 4, n. 1, p. 322-329, 2011.

NIEVERGELT, J. Complexity, algorithms, programs, systems: The shifting focus. **Journal of Symbolic Computation.** v. 17, p.297–310, 1994.

NOGUEIRA, T. A. F. M. P. Localização de unidades de fornecimento de gás natural na rede primária de gasodutos. Tese (Doutorado) - Universidade de Trás-os-Montes e Alto Douro. Vila Real, RJ, 2007.

REINELT, G. **The Traveling Salesman – Computational Solutions for TSP Applications.** Berlin: Springer- Verlag,1994

RIBAS, S. Um algoritmo híbrido para o problema de roteamento de veículos com janelas de tempo. Dissertação (Mestrado) - Universidade Federal Fluminense, Niterói, RJ, 2011.

RIBEIRO, Wellington Gomes; ARROYO, José Elias Claudio; DOS SANTOS, André Gustavo. METAHEURÍSTICA ILS PARA O PROBLEMA DE COBERTURA E ROTEAMENTO EM REDES DE SENSORES SEM FIO. **XLIV Simpósio Brasileiro de Pesquisa Operacional. Anais...**, Rio de Janeiro, p. 2576-2587, 2012.

RUSSELL, S.; NORVIG, P. **Inteligência Artificial.** 2ª Ed. Editora Campus, 2004.

SANTOS, D. G.; LOPES, R. H. C.; JÚNIOR, H. S. Análise da paralelização do 3-OPT aplicado sobre o problema ATSP**. XIII CLAIO - Congreso Latino-Iberoamericano de Investigación Operativa**, 2006, Montivideo. Anais do CLAIO 2006, 2006.

SILVA, D. F.; SANCHES, A. L. Aplicação Conjunta do Método de Dijkstra e Otimização Combinatória para Solução do Problema do Caixeiro Viajante. **Simpósio de Excelência em Gestão e Tecnologia.** São Francisco, 2012. Disponível em <http://www.economia.aedb.br/seget/artigos09/224_224_224_Artigo_Seget.pdf>. Acesso em 20/12/2013.

TEITZ, M. B.; BART, P. Heuristic methods for estimating the generalized vertex median of a weighted graph. **Operations Research**, v.16, p.955-961, 1968. ISSN 0096-3984.

VIEIRA, L. E. Algoritmo evolutivo para o problema do caixeiro viajante com demandas heterogêneas. Dissertação (Mestrado) - Universidade Federal de Santa Maria, Santa Maria, RS. 2006.

Sobre os autores

Beatriz Damica Freitas – Graduada em Sistemas de Informação pelo Centro Universitário São Camilo. Analista de Sistemas da Companhia de Tecnologia da Informação de Cachoeiro de Itapemirim-ES (DATACI).

Bruno Missi Xavier – Mestre em Pesquisa Operacional e Inteligência Computacional pela Universidade Cândido Mendes. Professor do Colegiado de Sistemas de Informação e Análise de Desenvolvimento de Sistemas do Centro Universitário São Camilo - ES. Analista de Sistemas da Companhia de Tecnologia da Informação de Cachoeiro de Itapemirim-ES (DATACI).

Fabio Machado de Oliveira – Mestrando em Cognição e Linguagem pela Universidade Estadual do Norte Fluminense Darcy Ribeiro – UENF. Especialista em Docência no Ensino Superior, Graduado em Ciência da Computação pela Universidade Cândico Mendes - RJ. Licenciatura em Matemática pela Faculdade de Filosofia, Ciências de Letras de Alegre – ES. Bolsista da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES no período de 2013 a 2014.

Marcelo Albuquerque Schuster - Mestre em Pesquisa Operacional e Inteligência Computacional pela Universidade Cândido Mendes. Coordenador dos Cursos de Sistemas de Informação e Análise de Desenvolvimento de Sistemas do Centro Universitário São Camilo - ES.