UNIVERSIDADE FEEVALE

ANDERSON HANSEN

USO DE ALGORITMOS GENÉTICOS ASSOCIADOS À GOOGLE DIRECTIONS API PARA OTIMIZAÇÃO DE ROTAS DE DISTRIBUIÇÃO NO ERP SIGER

Novo Hamburgo
2012

ANDERSON HANSEN

USO DE ALGORITMOS GENÉTICOS ASSOCIADOS À GOOGLE DIRECTIONS API PARA OTIMIZAÇÃO DE ROTAS DE DISTRIBUIÇÃO NO ERP SIGER

Trabalho de Conclusão de Curso
apresentado como requisito parcial
à obtenção do grau de Bacharel em
Ciência da Computação pela
Universidade Feevale

Orientadora: Marta Rosecler Bez

Coorientador: Ricardo Ferreira de Oliveira

Novo Hamburgo
2012

AGRADECIMENTOS

Agradeço aos professores da Universidade Feevale pela indispensável contribuição na minha vida acadêmica e pessoal.

Agradeço a meu amigo Luiz Henrique Feltes pela inestimável ajuda durante o trabalho. E o mais importante, obrigado pelos mais de 20 anos de amizade sincera e por permitir que eu lhe chame de amigo.

Agradeço ao meu co-orientador Ricardo pelos saberes compartilhados no percurso de elaboração deste trabalho que foram essenciais para a conquista deste resultado.

Agradeço a minha orientadora Marta pela dedicação e pela competência na orientação deste trabalho. Não há palavras na história da ciência que possa resumir sua postura. Obrigado por me mostrar que nada é tão contagioso quanto o exemplo.

E não deixando de agradecer de forma sincera a meus pais, Abílio e Bernardete, pela paciência, dedicação e pelo amor que me é dado todos os dias. A vocês eu rogo todas as noites a minha existência.

RESUMO

Com o acirramento da concorrência mundial, cada vez mais está sendo discutida a importância da informação para as organizações. A correta obtenção e manipulação da informação já é vista como um dos aspectos inerentes à gestão de uma organização. Como forma de atender a premissa supracitada, frequentemente as empresas optam pela utilização dos sistemas de gestão integrada, ou ERP (*Enterprise Resource Planning*). Esses tornam mais seguro e rápido o processo de apoio à tomada de decisão dentro das empresas. Este trabalho propõe o desenvolvimento de uma solução para o problema de geração de rota mínima de distribuição. Este problema de ordem real é encontrado atualmente na funcionalidade de controle de cargas do ERP SIGER, visto que, atualmente, a geração das rotas de entrega das mercadorias é feita de modo manual, contribuindo assim, para que as entregas das mercadorias não sejam feitas de forma eficiente. Como forma de solucionar este problema, propõem-se que as rotas de distribuição sejam geradas utilizando-se uma solução meta-heurística algoritmo genético. Para a maximização das chances do algoritmo genético gerar uma solução ótima, este será utilizado em conjunto com a *Google Directions API* em virtude deste serviço oferecer parâmetros precisos para a função de aptidão do algoritmo genético.

Palavras-chave: Problemas de Roteamento, Logística, Inteligência Artificial, Meta-Heurística, Algoritmo Genético, Otimização Combinatória, Heurística, *Google Directions API*.

ABSTRACT

With the intensification of global competition, increasingly is being discussed the importance of information for organizations. The proper collection and handling of information is already seen as one aspect of managing an organization. In order to meet the above premise, companies often opt for the use of integrated management systems, or ERP (Enterprise Resource Planning). These become more secure and fast process to support decision making within companies. This paper proposes the development of a solution to the problem of generating route minimum distribution. This problem of the actual order is currently found in the control functionality of the ERP charges Siger, as now, the generation of routes of delivery of goods is done in manual mode, thus contributing to the delivery of the goods are not made of efficiently. In order to solve this problem, we propose that the distribution routes are generated using a solution meta-heuristic genetic algorithm. To maximize the chances of the genetic algorithm to generate an optimal solution, this will be used in conjunction with the Google API Directions under this service offering precise parameters for the fitness function of genetic algorithm.

Keywords: Routing Problems, Logistics, Artificial Intelligence, Meta-Heuristics, Genetic Algorithms, Combinatorial Optimization, Heuristics, Google Directions API.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

SIBC	Sistema de Informação Baseado em Computador
ERP	*Enterprise Resource Planning*
SAP	*Systems, Applications, and Products in Data Processing*
CRM	*Customer relationship management*
SIGER	Sistema de Gestão Empresarial Rech
GCS	Gestão da Cadeia de Suprimentos
SRPV	Sistemas de Roteirização e Programação de Veículos
PRPV	Problemas de Roteirização e Programação de Veículos
PCV	Problema do Caixeiro Viajante
TSP	*Traveling Salesman Problem*
API	*Application Programming Interface*
PRV	Problemas de Roteirização de Veículos
ID	*Identify Duplicates*
XML	*Extensible Markup Language*
AJAX	*Asynchronous Javascript And XML*
JSON	*JavaScript Object Notation*
HTTP	*Hypertext Transfer Protocol*
URL	*Uniform Resource Locator*
IP	*Internet Protocol*
PMX	*Partially Mapped Crossover*
OX	*Order Crossover*

SUMÁRIO

INTRODUÇÃO

Com o acirramento da concorrência mundial oriunda principalmente do processo de globalização vivido nas últimas décadas, muitos conceitos relacionados à importância da informação vem sendo discutidos. Dentre os conceitos mundialmente aceitos, está o de que a obtenção e manipulação correta da informação consiste em um fator competitivo poderoso. Esta constatação coloca a informação como um recurso de competitividade real e efetivo.

No entanto, para que haja uma maximização das oportunidades geradas pela correta manipulação das informações, é necessário que estas estejam integradas e organizadas de forma que gerem conhecimento. Como forma de integrar todos os dados e processos de uma organização em um único sistema de informação, utiliza-se os sistemas de gestão integrada, ou ERP (*Enterprise Resource Planning*).

Cada vez mais os sistemas ERP estão tornando-se essenciais no processo de gerenciamento de um negócio. A aquisição de um sistema dessa família, torna mais seguro e rápido o processo de apoio à tomada de decisão dentro da empresa, de forma que esta decisão seja a mais correta possível, levando-se em consideração o momento e a situação.

Para um determinado ERP se autodenominar completo, este deve integrar e controlar todas as fases de um negócio, como, por exemplo: estoque, custos, compra de materiais e controle de cargas.

Uma das atribuições do controle de cargas está relacionada à distribuição das entregas. Este gerenciamento das entregas deve ser feito de modo que o custo gerado por este processo seja o menor possível. Além do caráter financeiro, um bom gerenciamento das entregas implica diretamente na qualidade do atendimento ao cliente.

Dentre as maneiras do custo gerado pelas entregas ser minimizado, uma das mais eficientes é a correta roteirização das entregas. Esta roteirização deve sempre passar pela premissa de que as entregas devem ser efetuadas pelo menor trajeto possível e no menor tempo possível.

Utilizando como cenário o ERP SIGER (Sistema Integrado de Gestão Empresarial Rech), este trabalho aborda como um sistema de controle de cargas pode gerar uma rota de entrega das mercadorias vendidas, de modo que seja sugerido ao usuário do sistema, uma rota mínima cuja distância percorrida pelo veículo que efetuará as entregas, seja a menor possível.

Neste trabalho sugere-se que a criação da rotina mínima de entrega das mercadorias seja gerada utilizando um método heurístico evolutivo baseado em um algoritmo genético associado a *Google Directions API.*

Os algoritmos evolucionários surgiram como uma alternativa aos algoritmos determinísticos para a resolução de problemas de otimização combinatória. Problemas de otimização combinatória, como o do Caixeiro Viajante, pertencem à classe de problemas NP-difícil (CUNHA, 2000) ou NP-árduo (GOLDBARG, LUNA, 2005). Esses nada mais são do que o conjunto de problemas para os quais não existe um algoritmo polinomial conhecido que encontre uma solução ótima em tempo computacional aceitável.

Os algoritmos genéticos podem ser definidos como uma técnica de busca baseada no processo biológico de evolução natural, onde os indivíduos altamente adaptados ao seu ambiente possuem naturalmente mais oportunidades para se reproduzir do que aqueles indivíduos considerados mais fracos.

Utilizando-se dos fatores evolutivos, um algoritmo genético tem como base gerar uma população inicial de soluções candidatas, e após isto, submeter esta população de soluções aos operadores genéticos seleção natural, recombinação e mutação, por diversas gerações, até que o critério de parada seja atendido.

A seleção natural, ou avaliação, tem por objetivo avaliar a aptidão das soluções candidatas. Essa avaliação é feita através da função de aptidão onde é atribuído um valor para a solução candidata de modo que este valor represente a capacidade da solução candidata em se adaptar ao problema proposto. Este valor atribuído será utilizado para diferenciar as melhores soluções entre a população de soluções candidatas.

A função de aptidão é o elemento que irá calcular a distância total de todas as rotas candidatas a fim de verificar qual a rota que sugere a menor distância para as entregas. Como forma de garantir que a função de aptidão receba dados precisos, neste caso a distância entre os locais de entrega, sugere-se que seja utilizado as *Google Maps APIs* disponibilizadas pelo Google. Essas APIs oferecem todo o suporte necessário para os aspectos relacionados à georreferenciamento. Dentre estas APIs está a *Google Directions API*. Esta nada mais é do que um serviço web que calcula rotas entre dois locais usando uma solicitação HTTP. Será feito uma pesquisa sobre os diferentes paradigmas de obtenção da distância real entre dois pontos no globo terrestre afim de que se coloquem estes paradigmas em contraponto com a *Google Directions API.*

O objetivo geral deste trabalho é desenvolver uma solução que, através de parâmetros oriundos do ERP SIGER, seja capaz de gerar uma rota mínima de distribuição das entregas através da utilização de um algoritmo meta-heurístico algoritmo genético associado a *Google Directions API*.

Os resultados encontrados através da geração de conhecimento obtida no desenvolvimento do protótipo serão dirigidos à solução de um problema real e específico encontrado no ERP SIGER. Desta forma, este trabalho não tem como finalidade somente acumular conhecimentos e informações que poderão eventualmente levar a resultados acadêmicos.

A utilização dos algoritmos genéticos em conjunto com a *Google Directions API* tem caráter exploratório em virtude desta objetivar a familiarização com um assunto ainda pouco explorado que é a conciliação destas duas abordagens. A utilização desta conciliação permite que seja adquirida uma visão geral mais aguçada sobre este dois temas. Esse aguçamento poderá contribuir para a formulação de soluções mais precisas para os problemas de roteamento, criação de novas hipóteses ou para a criação de novas propostas de conciliações que possam ser pesquisadas por estudos posteriores.

Como fundamentação teórica, serão pesquisados os conceitos básicos e os fundamentos teóricos relacionados aos sistemas de informação, mais especificamente os softwares ERP, a fim de se verificar a importância destes sistemas de informação para as organizações, bem como servirá para embasar o entendimento de que, sendo o ERP SIGER um membro do grupo dos softwares ERP, este deve oferecer uma solução eficaz para o problema de roteamento de entregas, de forma a minimizar o impacto deste tipo de problema em seus clientes.

Este trabalho está dividido em cinco capítulos. No primeiro é abordada a discussão da importância da informação para as organizações e como esta pode ser usada como um fator competitivo pelas empresas. No capítulo dois é realizado um estudo sobre o ERP SIGER e como este trata a questão das entregas das mercadorias. No terceiro capítulo é apresentado o problema de roteirização de veículos, bem como os problemas que este enfrenta para a sua resolução. No capítulo seguinte é realizada uma pesquisa sobre os algoritmos genéticos, e como estes, baseando-se em heurística, podem ser utilizados na resolução dos problemas de roteirização. No quinto é apresentado o desenvolvimento do protótipo que irá unir a eficácia dos algoritmos genéticos em encontrar soluções para problemas de otimização, e a precisão disponibilizada pela *Google Directions API*. Esta união pode proporcionar um serviço de

entrega mais preciso que será revertido em menores custos de transporte e, consequentemente, em maior lucro para os clientes do ERP SIGER. No sexto e último capítulo, será realizado um estudo de caso de modo que os resultados obtidos através do protótipo apresentado no quinto capítulo sejam estudados. A figura a seguir demonstra o sequenciamento dos objetivos a serem alcançados no decorrer do desenvolvimento do trabalho.

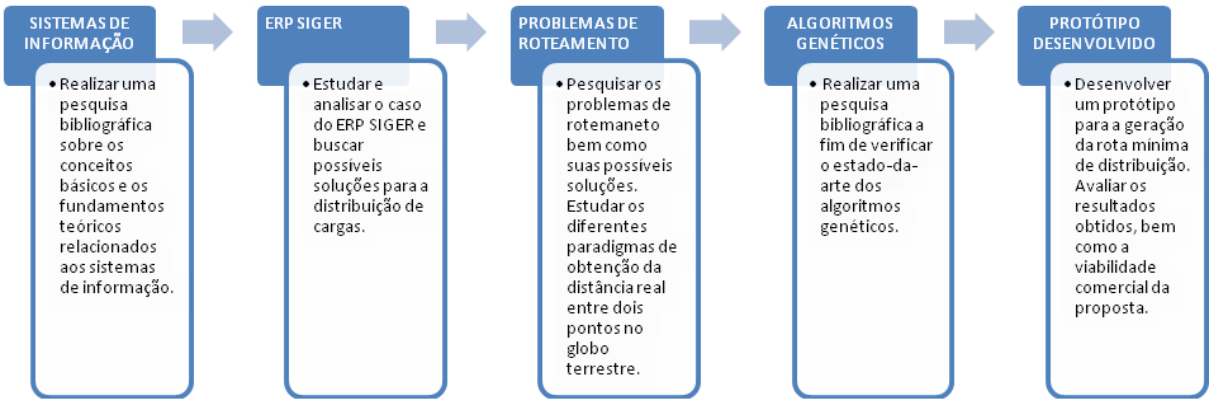

Figura 1 – Sequenciamento dos objetivos.
Fonte: Do próprio autor.

1 A IMPORTÂNCIA DA INFORMAÇÃO PARA AS ORGANIZAÇÕES

Nas últimas décadas está se vivenciando um fenômeno gerado pela necessidade do capitalismo de formar uma aldeia global: a globalização. Esta integração de caráter econômico, social, cultural e político entre os diferentes países, surgiu pela necessidade dos países desenvolvidos expandirem suas economias em virtude de seus mercados de origem encontrarem-se saturados.

Não se nega que a globalização estreitou as relações comerciais, sociais e culturais entre os países e as empresas, todavia, essa integração mundial, decorrente do processo de globalização, trouxe consigo a abertura dos mercados e o aumento da concorrência.

Com este aguçamento da concorrência, as empresas buscam cada vez mais vantagens competitivas ou diferenciais competitivos que façam com que estas se sobressaiam sobre seus concorrentes. Contudo, algumas vantagens competitivas que no passado eram consideradas diferenciais, atualmente tornaram-se essenciais. Uma destas vantagens competitivas que ganhou força nos últimos anos diz respeito à informação e ao conhecimento.

De acordo com Beal (2004), as organizações do século XXI existem num ambiente repleto de inter-relações que permanecem em constante estado de mutação e, nesse contexto, informação e conhecimento representam patrimônios cada vez mais valiosos, necessários para que se possa prever, compreender e responder às mudanças ambientais e alcançar ou manter uma posição favorável no mercado.

A informação e o conhecimento norteiam além das atividades operacionais das empresas, a elaboração e a definição de estratégias empresariais. Outra atividade norteada, diz respeito à orientação dos administradores nos processos de tomada de decisão. Isto por que quanto mais informação de qualidade for processada, menores são as incertezas da tomada de decisão.

A importância da informação para as organizações é universalmente aceita, constituindo, se não o mais importante, pelo menos um dos recursos cuja gestão e aproveitamento estão diretamente relacionados com o sucesso desejado. A informação também é considerada e utilizada em muitas organizações como um fator estruturante e um instrumento de gestão. Portanto, a gestão efetiva de uma organização requer a percepção objetiva e precisa dos valores da informação e do sistema de informação (MORESI, 2000).

Segundo Souza e Melhado (2007), a maior parte das empresas sabe como obter dados, mas precisam aprender a usá-los, pois uma base de dados, por maior que seja, não pode ser considerada como informação e, para que se transforme em informação, precisa ser organizada para um determinado objetivo, dirigida para um desempenho específico e aplicada a uma decisão.

Conforme Beal (2004), um conjunto de dados não produz necessariamente uma informação, nem um conjunto de informações representa necessariamente um conhecimento. Dados podem ser entendidos como registros ou fatos em sua forma primária, não necessariamente físicos. Quando esses registros ou fatos são organizados ou combinados de forma significativa, eles se transformam numa informação. Essa consiste em dados coletados, organizados, orientados, aos quais são atribuídos significados e contexto. (MCGEE; PRUSAK, 1994 apud BEAL, 2004).

Ao contrário da informação, o conhecimento não pode ser descrito e é subjetivo. Para sua obtenção, é necessária uma reflexão pessoal frente às informações. Esta reflexão tem por objetivo abstrair alguma ideia ou noção da informação. O ato ou efeito deste processo é o que se pode chamar de conhecimento. Esse pode ser definido como uma mistura fluida de experiência condensada, valores, informação contextual e *insight* experimentado, a qual proporciona uma estrutura para a avaliação e incorporação de novas experiências e informações. (DAVENPORT; PRUSAK, 1998 apud BEAL, 2004).

Para que a informação possa ser utilizada como vantagem competitiva, é essencial que além de existir uma forma eficiente de coleta de dados, e um meio de transformação destes dados em informação, exista também uma metodologia para que esta possa ser organizada e distribuída da forma correta.

O gerenciamento estratégico da informação transforma-se, assim, em uma necessidade dentro das empresas, visto que as informações são recursos organizacionais de suma importância, utilizados como forma de obtenção de sucesso estratégico.

1.1 Gestão estratégica da informação

O gerenciamento estratégico da informação é o processo dentro da empresa que parte da premissa que a informação certa e de qualidade deve ser disponibilizada no momento certo para a pessoa certa. A informação de qualidade deriva inexoravelmente de dados de

qualidade. De acordo com Turban; Rainer Jr.; Potter (2007), dados de qualidade devem ser precisos, completos, oportunos, coerentes, acessíveis, relevantes e concisos.

Entretanto, o que vem ocorrendo não é a ausência de informação, mas um grande volume de informações disponíveis, muitas vezes irrelevantes, ambíguas, e até mesmo inverídicas, o que prejudica os tomadores de decisão na hora de saber em qual informação confiar (BARBOZA, 2008).

Segundo Beal (2004), a boa gestão da informação evita que informações críticas para o sucesso da organização deixem de ser exploradas, diminuindo as chances de que o volume excessivo de informação possa acabar mascarando as informações relevantes para a solução dos problemas e que recursos sejam desperdiçados na obtenção e manutenção de informação sem utilidade.

O processo de gestão estratégica da informação utiliza-se de mecanismos, elementos e componentes que visam estruturar e disponibilizar de maneira eficaz a informação. O conjunto destes mecanismos, elementos e componentes que interagem entre si para prover informação, recebem o nome de Sistemas de Informação.

1.2 Sistemas de Informação

Considera-se um sistema de informação todo sistema que manipula dados e gera informação, usando ou não recursos de tecnologia da informação.

Conforme Beal (2004) um sistema é um conjunto de elementos ou componentes que interagem para atingir seus objetivos. Os sistemas têm entradas, mecanismos de processamento, saídas e *feedback* – uma saída usada para fazer ajustes na atuação do sistema. Turban; Rainer Jr.; Potter (2003) indicam que como qualquer outro sistema, um sistema de informação abrange entradas (dados) e saídas (relatórios, cálculos), processa essas entradas e gera saídas que são enviadas para o usuário ou outros sistemas. É possível incluir um mecanismo de resposta – *feedback* – que controle a operação. E como qualquer outro sistema, um sistema de informação opera dentro de um ambiente.

Num sistema de informação, a entrada corresponde a dados capturados, e a saída envolve a produção de informações úteis, muitas vezes na forma de relatório. O processamento envolve a conversão ou transformação dos dados em saídas úteis, e o *feedback* pode ser encontrado, por exemplo, nos procedimentos de detecção e correção de erros em dados de entrada (tais como a não aceitação de dados entrados em duplicata ou emissão de

alerta sobre valores digitados fora da faixa de valores válidos) (BEAL, 2004). A figura a seguir demonstra o sistema de informação no contexto da organização.

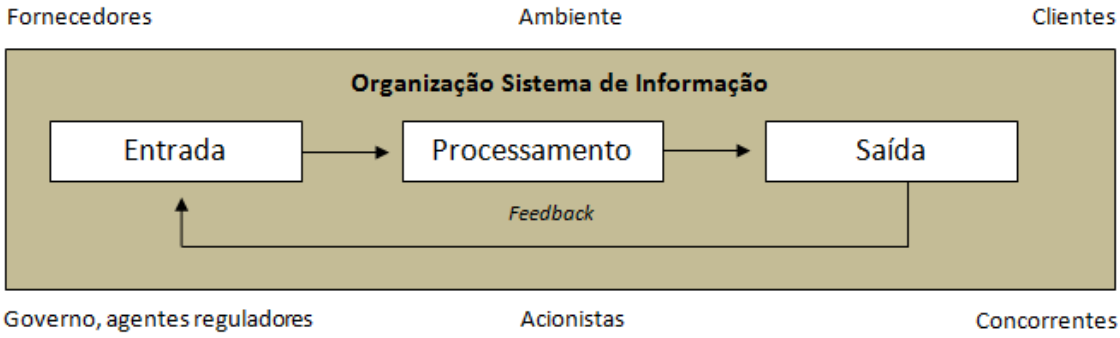

Figura 1.1 – O sistema de informação no contexto da organização.
Fonte: Beal, 2004, p. 16.

Os sistemas de informação podem ser tanto manuais quanto baseados em tecnologia da informação (BEAL, 2004) ou baseados em computador (TURBAN; RAINER JR.; POTTER (2003), TURBAN; RAINER JR.; POTTER (2007)).

De acordo com esses autores, um sistema de informação baseado em computador (SIBC) é um sistema de informação que usa tecnologia de computador para realizar algumas ou todas as tarefas pretendidas. Embora nem todos os sistemas de informação sejam computadorizados, a maioria é. Por essa razão, o termo "sistema de informação" normalmente é usado como sinônimo de "sistema de informação baseado em computador".

Os principais componentes dos sistemas de informação são descritos a seguir por Turban; Rainer Jr.; Potter (2003):

- Hardware: um conjunto de dispositivos, com processador, monitor, teclado e impressora, que aceita dados e informações, processa-os e os exibe.
- Software: um grupo de programas de computador que permite o processamento de dados no hardware.
- Banco de dados: um conjunto organizado de arquivos ou registros relacionados, que armazenam dados e as associações entre eles.
- Rede: um sistema de conectividade que viabiliza o compartilhamento de recursos entre computadores diferentes.

- Procedimentos: estratégias, políticas, métodos e regras para utilizar o sistema de informação.
- Pessoa: o componente mais importante nos sistemas de informação; inclui aqueles que trabalham com o próprio sistema ou usam sua saída.

A figura a seguir demonstra os componentes de um sistema de informação baseado em computador.

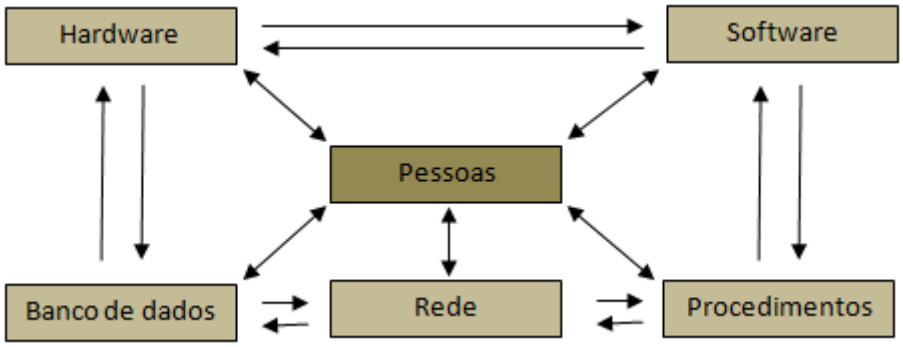

Figura 1.2 – Componentes de um sistema de informação baseado em computador.
Fonte: Turban; Rainer Jr.; Potter, 2003, p. 19.

Em uma organização podem existir diversos tipos de sistemas de informação. A existência desta variedade se deve ao fato de que em uma organização existem diferentes interesses, especialidades e níveis. Muitos destes sistemas de informação são desenvolvidos na forma de sistemas independentes, que não se comunicam com os demais sistemas de informação. Esta fragmentação da informação implica, muitas vezes, na compreensão errônea de informações, impossibilitando assim, que seja realizada uma análise global da situação real da organização.

Como forma de garantir a unicidade das informações, bem como garantir que todas as atividades da organização possuam registro de informação, as empresas utilizam cada vez mais os chamados Sistemas de Gestão Integrada, mais conhecidos como ERP (*Enterprise Resource Planning*).

1.3 Sistemas ERP

Os anos 90 assistiram ao surgimento e a um expressivo crescimento dos sistemas ERP (*Enterprise Resource Planning*) no mercado de soluções corporativas de informática. Entre as explicações para esse fenômeno estão as pressões competitivas sofridas pelas empresas, que as obrigaram a buscar alternativas para a redução de custos e diferenciação de produtos e serviços. Em função desse contexto, as empresas foram forçadas a rever seus

processos e sua maneira de trabalhar. Elas reconheceram a necessidade de coordenar melhor suas atividades dentro de sua cadeia de valor para eliminar desperdícios de recursos, reduzindo o custo e melhorando o tempo de resposta às mudanças das necessidades do mercado. (SOUZA; SACCOL, 2003).

Conforme Corrêa, Gianesi e Caon (2006), em uma tradução livre, *Enterprise Resource Planning* pode significar "Planejamento de Recursos da Corporação". Este termo tem sido cunhado como o estágio mais avançado dos sistemas tradicionalmente chamados de MRP II (*Manufacturing Resource Planning*), à medida que, além do controle dos recursos diretamente utilizados na manufatura (materiais, pessoas, equipamentos), também permitem controlar os demais recursos da empresa utilizados na produção, comercialização, distribuição e gestão.

Souza e Saccol (2003) indicam que a ideia de sistemas de informação integrados existe desse o início da utilização dos computadores em empresas na década de 60. Contudo, uma série de dificuldades de ordem prática e tecnológica não permitiu que essa visão fosse implementada na maior parte das empresas. Os mesmos autores concluem que os fatores que contribuíram para a expansão dos sistemas ERP foram o amadurecimento das opções disponíveis no mercado, a evolução da tecnologia utilizada por esses pacotes (banco de dados relacionais, processamento cliente/servidor e mais recentemente a Internet) e algumas histórias de sucesso em empresas que os adotaram no início da década de 90.

Apesar das empresas poderem desenvolver internamente sistemas com as mesmas características, o termo ERP está normalmente associado a pacotes comerciais. Conforme Souza e Saccol (2003), os sistemas ERP são sistemas de informação integrados adquiridos na forma de pacotes comerciais de software com a finalidade de dar suporte à maioria das operações de uma empresa industrial (suprimentos, manufatura, manutenção, administração financeira, contabilidade, recursos humanos, etc).

Embora tenham-se originado para atender basicamente a empresas industriais, os sistemas ERP estão atualmente ampliando sua abrangência. Empresas das áreas comerciais, distribuição, utilidades, financeira entre outras, já os têm implementado. Os próprios fornecedores têm buscado essa ampliação por meio do oferecimento de funcionalidades adequadas a esses e a outros tipos de empresas.

Os sistemas ERP possuem características que, se tomadas em conjunto, permitem distingui-los de sistemas desenvolvidos internamente nas empresas e de outros tipos de pacotes comerciais. Dentre estas características, Souza e Saccol (2003) citam:

- São pacotes comerciais de software;
- Incorporam modelos de processos de negócios (as chamadas *best practices)*;
- São sistemas de informação integrados e utilizam um banco de dados corporativo;
- Possuem grande abrangência funcional;
- Requerem procedimentos de ajuste para que possam ser utilizados em determinada empresa.

Souza e Saccol (2003) ainda indicam como exemplos de ERPs o R/3 da alemã SAP, o *iBaan Enterprise* da holandesa Baan, o *Oracle E-Business Suite* da americana Oracle, o EMS, o Magnus da brasileira Datasul e o AP7 Master da brasileira Microsiga.

Segundo Corrêa, Gianesi e Caon (2006), embora as melhores alternativas disponíveis de sistemas ditos ERP do mercado tenham um escopo que lhes permitiria chamarem-se ERPs, não é possível ainda, com segurança, afirmar que uma solução ERP tenha tido sucesso completo na sua utilização por um usuário que tenha passado por todos os seus módulos. Isso simplesmente por que ainda não houve tempo suficiente para uma empresa implantar todos os módulos de uma solução ERP disponíveis. Além disso, no Brasil, em particular, a maioria das soluções ERP mais robustas ainda passam por um grande esforço de tropicalização, ou, em outras palavras, adaptação dos módulos originais às particularidades brasileiras (legais, por exemplo).

No Brasil, são comercializados tanto ERPs de origem nacional como os de origem estrangeira. No caso dos estrangeiros, é necessário observar antes de uma eventual implementação, se estes estão adequados às leis fiscais, contábeis e trabalhistas do Brasil.

Destacam-se, no âmbito nacional, o sistema ERP SAP, desenvolvido pela empresa alemã SAP AG, e o ERP TOTVS da empresa TOTVS. Se for levada em conta somente a quantidade de implantações em empresas, independentemente do porte das mesmas, as empresas SAP AG e TOTVS lideram o mercado brasileiro de sistemas de gestão empresarial, possuindo juntas, cerca de 65% de penetração no mercado (AFONSO, 2012).

1.3.1 SAP ERP

Conforme SAP (2012a), o SAP ERP é o software de gestão empresarial desenvolvido pela empresa alemã SAP AG (*Systems, Applications, and Products in Data Processing*). A empresa surgiu na cidade de Mannheim, Alemanha, onde em 1972, cinco ex-empregados da IBM uniram-se para criar a sua própria empresa de desenvolvimento de sistemas: a SAP AG.

Um ano depois da criação da empresa, o primeiro aplicativo ficou pronto. Tratava-se de um sistema de contabilidade financeira. Este foi utilizado como base para o contínuo desenvolvimento de outros componentes de software para aquilo que mais tarde veio a ser conhecido como sistema "R/1" (SAP, 2012a).

A versão SAP R/2 ou *Real time System Version 2*, foi o primeiro produto de sucesso criado pela empresa, pois era constituído por vários módulos e foi utilizado até 1995 por quase duas mil empresas no mundo (SAP, 2012d).

Nos anos 90, a versão SAP R/3 surgiu, elevando a SAP a um novo patamar de inovação tecnológica. Isto por que a nova versão abdicou dos antigos paradigmas para utilizar o conceito de cliente-servidor. Houve mudanças quanto à aparência uniforme de suas interfaces gráficas, o uso consistente de bancos de dados relacionais e a capacidade de ser executado em computadores de diferentes fornecedores. Até hoje, a arquitetura cliente-servidor é o padrão utilizado pela SAP. Em 1996, a companhia ganhou 1.089 novos clientes com o SAP R/3. No final daquele ano, o SAP R/3 estava instalado em mais de 9.000 sistemas no mundo todo (SAP, 2012c).

Ao longo de três décadas, a SAP evoluiu de uma empresa pequena e regional a uma organização de alcance mundial. Atualmente, a SAP conta com locais de desenvolvimento e vendas em mais de 50 países em todo o mundo. As aplicações e serviços da SAP permitem a mais de 183.000 clientes em todo o mundo operar com rentabilidade, adaptar-se às constantes mudanças do mercado e crescer de forma sustentável (SAP, 2012c).

O SAP ERP destina-se exclusivamente ao planejamento dos recursos corporativos. Demais processos relacionados ao negócio, como, por exemplo, a gestão das relações com clientes (CRM), possuem aplicativos distintos, que assim como o SAP ERP, são tratados como módulo no aplicativo SAP *Business Suite*.

Buscando uma melhora na eficiência e no alinhamento estratégico dos processos financeiros, operacionais e de capital humano, o SAP ERP disponibiliza quatro soluções

individuais que sustentam as principais áreas funcionais das organizações. Segundo SAP (2012b), são elas:

- SAP ERP *Financials*: Permite que as empresas adquiram maior visibilidade e controle sobre as operações financeiras da organização como um todo. O aplicativo automatiza a contabilidade e a gestão financeira, além de gerenciar toda a cadeia de suprimentos de finanças.
- SAP ERP *Human Capital Management* – Maximiza o potencial da força de trabalho e propicia a inovação, crescimento e a flexibilidade. A solução também automatiza a gestão de talentos e os principais processos de RH, incluindo alocação mais eficiente da força de trabalho e maior conformidade com os sempre mutantes requisitos regulatórios globais e locais.
- SAP ERP *Operations* – Permite gerenciar os processos de negócios de compras e logística para cobrir toda a extensão dos ciclos de negócios – desde requisições de compras por autoatendimento até o faturamento e processos de pagamento mais flexíveis, otimizando o fluxo de materiais.
- SAP ERP *Corporate Services* – Otimiza tanto os serviços centralizados como os serviços descentralizados de gerenciamento de imóveis, recursos corporativos, portfólios de projetos, viagens, meio-ambiente, segurança e saúde no trabalho, qualidade e serviços de comércio exterior.

Segundo CIA (2012), como líder do mercado mundial de aplicações de software empresarial, a SAP não pode ser considerada somente um provedor de software. Ela também é uma consultoria de serviços, trabalhando junto aos clientes ao longo de todo ciclo de vida. Com cerca de 10.000 consultores em todo o mundo, a SAP oferece experiência, conhecimento e as soluções mais completas e atualizadas em nível global e local.

No Brasil, ela conta com mais de 200 consultores atuando com clientes dos mais diversos ramos, como Manufatura, Varejo, Petrolíferas, Mineradoras, Indústria Financeira entre outros (CIA, 2012).

1.3.2 ERP TOTVS

De acordo com TOTVS (2012c), o ERP TOTVS é o software de gestão empresarial, criado e desenvolvido pela empresa TOTVS. A origem da empresa remete a 1983, quando Ernesto Haberkorn e Laércio Cosentino criam a Microsiga Software S.A.

Ao longo dos anos, a Microsiga realizou diversas incorporações que fizeram com que fosse criada a TOTVS com o objetivo de reunir as diversas marcas provenientes das aquisições realizadas. Atualmente, a TOTVS é a controladora das marcas Microsiga, Datasul e Logocenter, por exemplo (TOTVS, 2012c).

Conforme TOTVS (2012c), a TOTVS é uma empresa de software, inovação, relacionamento e suporte à gestão, líder absoluta no Brasil, com 48,6% de participação de mercado, e também na América Latina, com 34,5%. É a maior empresa de softwares aplicativos sediada em países emergentes e a 6ª maior do mundo no setor.

A solução ERP TOTVS é dinâmica, racional e eficiente, e prepara a companhia para administrar processos e recursos na busca de integração de informações. A adoção da solução ERP TOTVS elimina o uso de interfaces manuais e a redundância de atividades, proporcionando integração de diversos departamentos, automatização e armazenamento de todas as informações de negócios (TOTVS, 2012a).

Segundo TOTVS (2012c), a empresa possui mais de 26 mil clientes ativos e conta com o apoio de aproximadamente 10 mil participantes em unidades próprias e franqueadas. Possui unidades próprias no México, Argentina e Portugal e está presente em 23 países.

Dentre as inúmeras soluções ERP disponíveis no mercado, destaca-se também o ERP SIGER. Este consolidou-se no mercado em virtude da sua versatilidade, eficiência e flexibilidade nas adaptações. Tais características, garantem uma total aderência do ERP a uma grande quantidade de ramos de atividades.

2 ERP SIGER

O SIGER é o ERP desenvolvido pela Rech Informática Ltda desde 1990. Fundada pelos irmãos Carlos Vanderlei Rech e Rovani Marcelo Rech, a empresa foi constituída a partir de um alicerce sólido de valores humanos, conjugando experiência, seriedade e competência nas áreas de informática e gestão empresarial. Atualmente, a Rech Informática Ltda conta com mais de 600 clientes, que juntos, totalizam mais de 5.000 usuários utilizando o SIGER em seu dia-a-dia (RECH, 2012).

A empresa possui clientes nos estados do Rio Grande do Sul, Santa Catarina, Paraná, São Paulo, Minas Gerais, Rio de Janeiro, Espírito Santo, Bahia, Pernambuco, Paraíba, Ceará, Mato Grosso, Distrito Federal e Goiás, atuando mais especificamente no Vale dos Sinos e região metropolitana de Porto Alegre. A figura a seguir apresenta de forma gráfica a carteira de clientes por atividade.

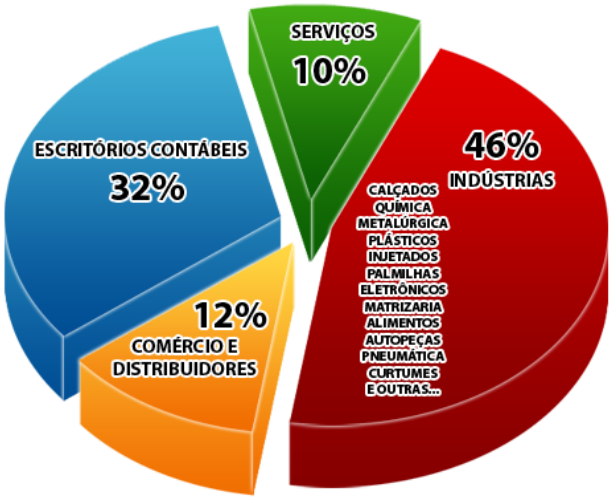

Figura 2.1 – Representação gráfica da carteira de clientes por atividade.
Fonte: Rech Informática, 2012

A missão principal da empresa é fornecer produtos e serviços de alta qualidade, contribuindo para que clientes e colaboradores atinjam seus objetivos estratégicos, explorando ao máximo seus potenciais e otimizando seus processos através da integração de tecnologia e sistema de informações. Para que o resultado implícito na missão possa ser atingido em sua totalidade, a empresa conta com os seguintes princípios:

- Satisfação: Superar as expectativas dos clientes com soluções de qualidade.
- Tecnologia: Utilizar tecnologias confiáveis, viáveis e produtivas.
- Recursos Humanos: Valorização, espírito de equipe e aprimoramento contínuo.

- Valores: Honestidade, humildade, seriedade, transparência, crescimento sustentado, solidez, habilidade e competência.
- Qualidade: Foco na qualidade total como receita para o sucesso.
- Ética: Ser ético e justo em todos os relacionamentos internos e externos.
- Dinamismo: Flexibilidade e agilidade no atendimento e soluções.
- Lucro: Consequência dos princípios adotados e objetivos alcançados.

A Rech Informática Ltda está instalada no município de Novo Hamburgo, Rio Grande do Sul, onde possui uma sede própria de aproximadamente 1.200m² de área construída, em um terreno de mais de 3.200m². O prédio conta com um amplo estacionamento que abriga com total segurança até 50 automóveis, auditório para conferências, centro de treinamento e um salão de festas com capacidade para mais de 80 pessoas. Todo o ambiente operacional é de alta qualidade, com tecnologia Dell.

De acordo com Rech Informática (2012), a estrutura organizacional da empresa está divida da seguinte forma:

- Direção: Constituída pelos irmãos Carlos Vanderlei Rech e Rovani Marcelo Rech, responsáveis pela estratégia, visão e administração da Rech Informática.
- Controladoria: Divisão responsável pelo monitoramento funcional e operacional da gestão Rech e seu relacionamento com o mercado. Objetiva apontar os sinais vitais da empresa, e por coerência utilizam as mesmas soluções que são oferecidas ao mercado.
- Suporte: Divisão responsável pelo canal direto e aberto permanentemente nos relacionamentos com os clientes e mercado. Objetiva a manutenção do uso do SIGER.
- Consultoria: Divisão responsável pelos relacionamentos comerciais, de marketing e de negócios com o mercado. Objetiva reconhecer o processo de gestão das empresas e modelar soluções em ERP. Este processo controlado de identificação, modelagem e implantação personalizado garante que se continue a criar demanda utilizando o efeito multiplicador a partir da satisfação dos clientes.

Desenvolvimento: Divisão responsável pelo planejamento e execução de projetos de desenvolvimento no SIGER, relacionando demanda e novas tecnologias. Objetiva a recepção, interpretação e análise da demanda, produzindo a solução adequada no SIGER.

O amplo conjunto de ferramentas operacionais e de gestão desenvolvidos pela Rech Informática Ltda estão contidos e são comercializados na forma de um único produto, a solução ERP da Rech, o SIGER.

2.1 SIGER

O ERP SIGER é comercializado na forma de mensalidades, nas quais já estão incluídos o custo do levantamento de requisitos; a implantação do sistema; o treinamento de usuários; as atualizações de versões; as horas técnicas de visitas no cliente e o suporte via telefone, e-mail ou fax. Se por ventura o ERP SIGER não contemplar alguma necessidade de um determinado cliente, existe a possibilidade da contratação de desenvolvimentos específicos por parte deste cliente.

Por ser um sistema altamente parametrizável, o SIGER pode ser implantado e configurado de acordo com as necessidades e interesses de cada negócio. Antes de ser realizada a implantação do sistema, são avaliados os processos e rotinas dos diversos setores da empresa e constatadas as necessidades de customização e configuração para maximizar a eficiência da empresa. Só então dá-se início ao processo de implantação do sistema, onde todas as informações levantadas são analisadas e parametrizadas. (RECH, 2012).

Considerado um aplicativo de grande porte, o SIGER conta com mais de 8.300.000 (oito milhões e trezentas mil) linhas de código, 3.800 programas, 1.200 tabelas de banco de dados, 3.500 telas de interface de usuário e milhares de formulários e relatórios gráficos.

Todo o desenvolvimento do ERP SIGER é feito na sede da empresa, utilizando como ferramenta de desenvolvimento, o Microfocus NetExpress 3 e repositório de dados nativo. Está sendo desenvolvida uma nova versão do SIGER que tem como principal característica ser independente de plataforma. Esta nova versão é baseada na plataforma Java, e é gerada através da tecnologia COBOL-to-Java da empresa Veryant. A figura a seguir apresenta algumas interfaces da nova versão.

Figura 2.2 – Interfaces da versão multiplataforma do SIGER.
Fonte: Rech Informática, 2012

O SIGER é composto por diversos módulos, que são comercializados de forma independente, mas que ao serem adquiridos juntos, funcionam de forma totalmente integrada. Isto possibilita que cada empresa adquira o que é realmente necessário para seu negócio. A figura a seguir demonstra a abrangência modular que o ERP SIGER disponibiliza para contratação de seus clientes:

Figura 2.3 – Principais módulos do SIGER por área de gestão.
Fonte: Rech Informática, 2012

Dentre os módulos da área comercial destaca-se o módulo de Faturamento/Vendas, que será abordado na sequencia.

2.2 Módulo de Faturamento/Vendas

Este módulo tem por objetivo controlar os processos de vendas da empresa. Dentre os recursos disponíveis, encontram-se diversas ferramentas para geração de informações estatísticas referentes às vendas, controle da rentabilidade, comissões pagas a representantes, etc.

No processo de vendas utilizando o SIGER, na maioria das empresas, o primeiro movimento na comercialização de uma mercadoria é a geração de um orçamento. Esta entidade representa a intenção de compra por parte de um cliente. Como dito anteriormente, dependendo da política interna da empresa, a inclusão do orçamento não se faz necessária, sendo o primeiro movimento a inclusão do pedido.

Pedido é a entidade que representa e formaliza uma intenção de compra por parte de um cliente. É no pedido que encontram-se as informações referentes ao fechamento do negócio, como, por exemplo, as condições de pagamento negociadas com o cliente, bem como, a política de entregas das mercadorias.

Com o pedido incluso é possível emitir a nota fiscal, que nada mais é do que o documento que comprova uma transferência de bens, produtos e/ou serviços entre duas entidades legais, independentemente da característica desta transferência; ou seja, venda, remessa, empréstimo, transferência, devolução, etc. É um documento obrigatório devido ao seu caráter fiscal e demonstrativo para arrecadação de impostos

No módulo de Faturamento/Vendas, pode-se encontrar a funcionalidade de controle de cargas. Esta permite que haja um total controle sobre as cargas através das rotinas de emissão de ordem de carga, controle das entregas dos pedidos, roteiros e gerenciamento da distribuição e logística.

2.3 Funcionalidade de Controle de Cargas

Quando se inclui um pedido, pode-se vinculá-lo a uma ordem de carga. Ordem de carga é uma entidade utilizada para agrupar os pedidos por afinidade e que é utilizada para montar a logística do transporte. Alguns dos atributos que encontram-se na ordem de carga são: transportador, via de transporte, data de entrega, etc. Geralmente, as empresas criam uma ordem de carga por dia para cada representante da empresa. Isto facilita a rastreabilidade diária das entregas de mercadorias vendidas por um determinado representante.

Uma vez que se obtenha uma quantidade mínima de ordens de carga que justifique e viabilize o transporte, deve-se organizar as entregas das mercadorias.

Essa é feita através da criação de um resumo de ordens de carga, que nada mais é do que uma fila de entrega. Este resumo de ordens de carga é importante para a qualidade da entrega, pois é através dele que os carregadores saberão a ordem que as mercadorias serão alocadas dentro do veículo, bem como, os motoristas do veículo saberão a rota de entrega das mercadorias.

A solução ótima no processo de alocamento das mercadorias no veículo é que as mercadorias que serão entregues por último, devem ser embarcadas primeiro, ocupando assim, o fundo do veículo. As mercadorias que serão entregues primeiro devem ser alocadas por último, ocupando assim, a parte mais próxima da porta do veículo. Se por ventura uma mercadoria que será entregue por primeiro for alocada primeiro, ela consequentemente vai ocupar a parte mais distante da porta do veículo, o fundo. Logo, no momento que esta for entregue, todas as demais mercadorias terão que ser retiradas do veículo antes de alcançar esta mercadoria. Isto ocasiona trabalho desnecessário aos motoristas do veículo, pois para pegar a mercadoria a ser entregue, estes terão que descarregar todo o veículo.

A solução ótima para o problema da rota de entrega é que esta seja cumprida com o custo mínimo, isto é, a distância de todo o trajeto decorrentes das entregas deve ser o menor possível.

Atualmente, a montagem do resumo de ordens de carga no SIGER é feita manualmente, portanto, a ordenação das entregas das ordens de carga é feita levando em consideração a experiência do usuário designado para tal tarefa. Como toda tarefa manual, a criação dos resumos de ordens de carga está sujeita a falhas humanas que geram custos de transporte desnecessários para o cliente.

Como forma de automatizar a ordenação das entregas, pode-se aplicar uma das soluções para os tradicionais problemas de roteirização de veículos (PRV). Embora exaustivamente estudado, este tipo de problema possui uma grande importância econômica, visto que existe um considerável percentual do valor da mercadoria que chega aos clientes que é de exclusiva responsabilidade dos gastos obtidos através de sua distribuição.

3 PROBLEMAS DE ROTEAMENTO

Com a era da globalização e a introdução da filosofia de gestão da cadeia de suprimentos (GCS), os clientes têm se tornado cada vez mais exigentes no que diz respeito à qualidade e prazos de entrega, gerando uma competitividade crescente e uma busca por serviços cada vez mais customizados que, para as empresas de distribuição de produtos, tem se tornado um fator importante na obtenção de vantagem competitiva e conquista de fatias cada vez maiores do mercado (MELO; FILHO, 2001).

De acordo com Hermel (2008), a logística de distribuição de produtos e mercadorias nas organizações assume um papel de grande importância. O avanço tecnológico advindo dos computadores, celulares, redes sem fio e internet proporcionam agilidade desde a compra, faturamento até a expedição dos pedidos para o cliente. As empresas focadas em atender seus clientes com agilidade, rapidez e pontualidade, mas com economias e custos que garantam a continuidade dos seus negócios, procuram implantar e adotar formas de otimizar os ganhos e minimizar custos com a distribuição.

A distribuição física dos produtos contribui com cerca de 16% do custo final do item. Por outro lado, certos produtos carecem de uma distribuição eficiente por motivos não só econômicos, como de segurança. Exemplos do caso são medicamentos e os combustíveis (BODIN, 1983 apud GOLDBARG; LUNA, 2005).

Conforme Melo e Filho (2001), muitas empresas de transporte têm tentado dar maior confiabilidade, mais velocidade e flexibilidade, assim como praticar a intermodalidade em todos os seus canais de distribuição. O objetivo dessa busca é encontrar uma maior eficiência e pontualidade nas tarefas de entrega e/ou coleta; um melhor aproveitamento da frota e dos motoristas; menores tempos de ciclo e melhor planejamento das rotas. Por consequência, há uma sensível redução dos custos operacionais, melhora da imagem da empresa no mercado, maior fidelidade de clientes e, em função disso, uma conquista cada vez maior de fatias de mercado. Nesse sentido, de modo a obter excelência nos processos de distribuição física, muitas empresas têm adquirido os chamados sistemas de roteirização e programação de veículos (SRPV).

Sistemas de roteirização e programação de veículos ou, simplesmente, roteirizadores, são sistemas computacionais que, através de algoritmos, geralmente heurísticos, e uma apropriada base de dados, são capazes de obter soluções para problemas de roteirização e programação de veículos (PRPV) com resultados relativamente satisfatórios, consumindo

tempo e esforço de processamento pequeno quando comparados aos gastos nos tradicionais métodos manuais (MELO; FILHO, 2001).

Para Cunha (1997 apud SILVA, 2010) a relevância dos problemas de roteirização e programação de veículos pode ser medida pelo expressivo número de artigos publicados na literatura especializada, além da permanente busca de novas tecnologias e métodos de solução para a resolução de modelos cada vez mais complexos e abrangentes.

Os problemas de roteamento de veículos possuem um número extraordinário de aplicações práticas, pois implicam tipicamente em uma série de situações reais que afetam principalmente a indústria, o comércio, o setor de serviços, a segurança, a saúde pública e o laser. Dentre outras se destacam, segundo Goldbarg e Luna (2005), os seguintes:

- Distribuição de manufaturados;
- Serviços de emergência;
- Limpeza de ruas com veículos vassoura;

Ainda segundo os autores, estes problemas abordam basicamente a determinação de sequências de visitas que objetivem atender a uma determinada função objetivo. As visitas podem tanto estar associadas às ligações (arestas) ou a pontos de visita (nós ou vértices) do grafo que representa as possíveis conexões entre os pontos de visita (ou pontos de ligações entre as arestas).

Grafo, de acordo com Boaventura Netto e Jurkiewicz (2009), pode ser caracterizado como um objeto matemático (ou estrutura matemática) composto por dois conjuntos de elementos. O primeiro é o conjunto dos vértices e o segundo são as relações entre os vértices, chamadas arestas. Para se conhecer um grafo precisa-se conhecer os vértices e "quem está ligado com quem". A figura a seguir demonstra um grafo direcionado (Digrafo) que representa as cidades de uma região, onde os vértices ou cidades, estão representados por letras, e as arestas ou caminhos entre elas, estão representados por um valor numérico que indica a distância entre elas.

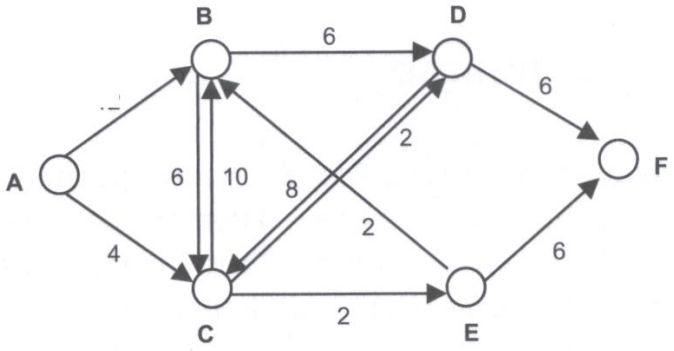

Figura 3.1 – Grafo das cidades de uma região.
Fonte: Boaventura Netto, Jurkiewicz, 2009, p. 38.

Os problemas de roteamento de veículos estão entre os mais complexos da área de otimização combinatória. Problemas de otimização combinatória, como o do Caixeiro Viajante, pertencem à classe de problemas NP-difícil (CUNHA, 1997 apud CUNHA, 2000; WU, 2007) ou NP-árduo (GOLDBARG; LUNA, 2005). Esses nada mais são do que o conjunto de problemas para os quais não existe um algoritmo polinomial conhecido que encontre uma solução ótima em tempo computacional aceitável. Conforme Cunha (1997 apud CUNHA, 2000), em outras palavras, o esforço computacional para a sua resolução cresce exponencialmente com o tamanho do problema (dado pelo número de pontos a serem atendidos). Da mesma forma, WU (2007) complementa indicando que os problemas de roteirização de veículos são conhecidos pela natureza combinatória, o que significa que a dificuldade em resolvê-los aumenta de forma exponencial à medida que o tamanho do problema cresce.

Uma forma de classificar os problemas de roteamento proposta por Maganti (1981 apud GOLDBARG; LUNA, 2005), consiste em separar os clássicos problemas de roteamento em grafos das demais variantes. Nesse sentido, os problemas de roteamento em geral poderiam ser classificados em duas grandes classes: roteamento em grafos e roteamento de veículos propriamente ditos. A classe geral dos problemas de roteamento em grafos é constituída pelas seguintes subclasses:

1) Problemas de roteamento em nós.

2) Problemas de roteamento em arcos.

Segundo WU (2007), problemas como o da coleta de lixo residencial e limpeza das ruas, caracterizam-se por problemas em atendimento de arcos. Já os problemas como o da coleta de peças e coleta de lixo industrial, caracterizam-se por problemas de atendimento em nós, tipo Caixeiro Viajante.

3.1 Problema do Caixeiro Viajante

De acordo com Goldbarg e Luna (2005), os problemas de roteamento lidam em sua maior parte com passeios ou *tours* sobre pontos de demanda ou oferta. Esses pontos podem ser representados por cidades, postos de trabalho ou atendimento, depósitos, etc. Dentre os tipos de passeios um dos mais importantes é o denominado hamiltoniano. Seu nome é devido a Willian Rowan Hamilton que, em 1857, propôs um jogo que denominou *Around the World*. O jogo era feito sobre um dodecaedro em que cada vértice estava associado a uma cidade importante na época. O desafio consistia em encontrar uma rota através dos vértices do dodecaedro que iniciasse e terminasse em uma mesma cidade sem nunca repetir uma visita. A figura a seguir demonstra o grafo do problema.

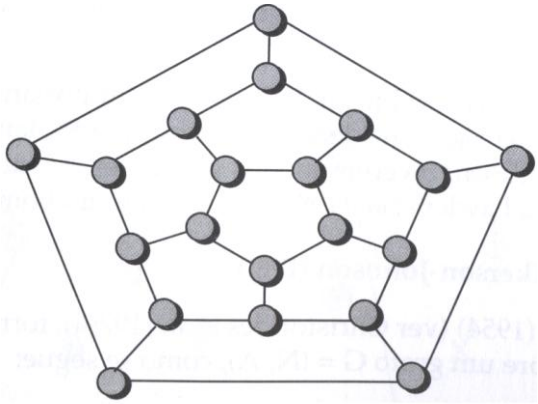

Figura 3.2 – Jogo de Hamilton.
Fonte: Goldbarg, Luna, 2005, p. 331.

O problema do Caixeiro Viajante (PCV) é um problema de otimização associado ao da determinação dos caminhos hamiltonianos em um grafo qualquer. O objetivo do PCV é encontrar em um grafo G = (N, A), o caminho hamiltoniano de menor custo, onde "N" representa os nós ou vértices e "A" representa as arestas (GOLDBARG; LUNA, 2005).

Conforme Cunha (2000), o primeiro problema de roteirização a ser estudado foi o do folclórico caixeiro viajante (no inglês "*traveling salesman problem*" ou TSP), que consiste em encontrar o roteiro ou sequência de cidades a serem visitadas por um caixeiro viajante que minimize a distância total percorrida e assegure que cada cidade seja visitada exatamente uma vez.

Desde então, novas restrições vêm sendo incorporadas ao problema do Caixeiro Viajante, de modo a melhor representar os diferentes tipos de problemas que envolvem roteiros de pessoas e veículos, entre as quais: restrições de horário de atendimento (conhecidas na literatura como janelas de tempo ou janelas horárias); capacidades dos

veículos; frota composta de veículos de diferentes tamanhos; duração máxima dos roteiros dos veículos (tempo ou distância); restrições de tipos de veículos que podem atender determinados clientes (CUNHA, 2000).

Um sistema de roteamento pode ser considerado como um conjunto organizado de meios que objetiva o atendimento de demandas localizadas nos arcos ou nos vértices de alguma rede de transportes (GOLDBARG; LUNA, 2005).

Desta forma, é possível modelar problemas reais de roteirização através da utilização de grafos onde os vértices (ou nós) são as abstrações dos locais a serem visitados e as arestas (ou arcos) são os percursos que ligam os locais. Para as arestas, são indicados valores ou pesos. Estes pesos, quando dirigidos à solução de um problema real, irão representar a distância entre os pontos que são ligados por esta aresta. Para a correta resolução de problemas de roteirização de ordem real, é necessária que a distância entre os locais visitados seja a mais verossímil possível. Para isto, é imprescindível que seja definido qual o paradigma de obtenção da distância real entre dois pontos no globo terrestre que retorna a distância mais precisa para que esta seja utilizada na carga do peso da aresta.

3.2 Paradigmas de obtenção da distância entre dois pontos no globo

Como forma de calcular a distância entre dois pontos no globo terrestre, primeiramente é necessário definir com precisão a posição de cada um deles. Atualmente esta tarefa é considerada relativamente fácil em virtude dos avanços tecnológicos das últimas décadas. No entanto, em um passado não muito distante, questões mais simples como a forma do planeta, foram motivos de calorosas discussões entre os pensadores da época.

Desde os primeiros anos escolares, é ministrado o conceito errôneo de que a Terra é redonda. Segundo Rocha (2002), a Terra, ao longo da história da humanidade, já foi conhecida sob diversas formas. Pitágoras (6° séc. A.C) e Aristóteles (4° séc. A.C) definiram a Terra como esférica. Newton (séc. XVII) considerou-a elipsoidal. Gauss (séc. XVIII) concluiu que a melhor forma seria a Geoidal.

De acordo com Rocha (2002), geoide é uma superfície ondulada e não possui uma forma matemática ou geométrica conhecida. Ela não pode, portanto, ser usada como uma superfície de referência para o posicionamento de pontos da superfície topográfica. O geoide é considerado como a superfície de nível de altitude igual a zero e coincidente com o nível médio dos mares. Mesmo não sendo esta a forma utilizada como referência para cálculos em

cartografia, Fitz (2008) complementa indicando que o geoide é a superfície que melhor representa a superfície real do planeta.

Ainda conforme Rocha (2002), a superfície adotada como referência para os cálculos de posição, distância, direções e outros elementos geométricos da Cartografia é o elipsóide. O elipsóide é formado a partir de uma elipse rotacionada em torno do seu semi-eixo menor (norte-sul).

Para que o posicionamento de um ponto sobre o elipsóide de referência seja realizado de maneira unívoca, foram estabelecidas linhas de referencia imaginárias sobre ele. As linhas de referência permitem determinar a posição de um ponto sobre a superfície esférica (ROCHA, 2002).

Segundo Friedmann (2009), chama-se linha do equador o conjunto de pontos da superfície da Terra que estão situados à mesma distância do Pólo Norte e do Pólo Sul. Este conjunto de pontos é, muito aproximadamente, uma circunferência. Denomina-se Pararelo, a circunferência obtida por interseção da superfície da esfera com um plano perpendicular ao eixo da esfera e passando por um ponto do eixo situado entre o centro da esfera e um dos pólos. Já Meridiano é a circunferência definida na superfície da esfera por um plano passante pelo seu eixo. Rocha (2002) simplifica indicando que as linhas desenhadas no sentido Norte/Sul são denominadas Meridianos e as linhas desenhadas no sentido Leste/Oeste são denominadas Paralelos.

O cruzamento de um meridiano com um paralelo caracteriza de forma inequívoca um ponto na superfície da esfera. Observa-se que tanto a posição de um paralelo quanto a de um meridiano podem ser adequadamente expressas como ângulos planos. Para os paralelos a referência é a linha do Equador. Para os meridianos, não há uma referência notável como há para os paralelos, sendo a solução a fixação de um meridiano de referência (FRIEDMANN 2009).

As duas medidas angulares, ou ângulos, que representam a localização do paralelo e do meridiano são chamadas, respectivamente, de latitude e longitude.

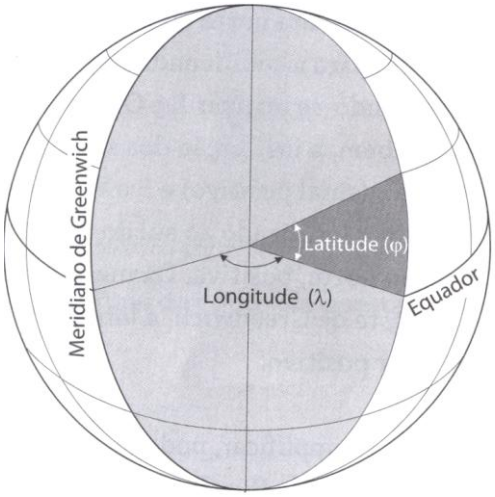

Figura 3.3 – Latitude e Longitude.
Fonte: Fitz, 2008, p. 35.

Latitude é o ângulo correspondente ao arco entre o Equador e o paralelo considerado, estando o vértice no centro da Terra. A latitude varia entre 0° e 90°. No Equador, a latitude é de 0°, aumentando progressivamente até 90° à medida que os paralelos se aproximam dos polos. Longitude é o ângulo correspondente ao arco equatorial entre o meridiano considerado e o meridiano de referência 0°, denominado Meridiano de Greenwich (FRIEDMANN 2009).

Uma vez que se possa definir a longitude e a latitude de um ponto de origem e outro de destino, é possível traçar uma trajetória entre eles. Algumas das técnicas, que através das informações da localização destes pontos calculam uma via de acesso ou rota entre eles, são descritas a seguir.

3.2.1 Teorema de Pitágoras

Apesar do formato da Terra ser muito aproximado com o de uma esfera, este também pode ser representado através de uma superfície plana, como, por exemplo, os mapas. Assim como a localização de pontos sobre a superfície da Terra se dá através da utilização de um sistema de coordenadas, é possível também representar pontos sobre uma superfície plana através da utilização deste sistema. Segundo Lehmann (1998), o estudo que utiliza o conceito de um sistema de coordenadas para a resolução de problemas algébricos em uma superfície plana é chamado de Geometria Analítica.

O conceito de um sistema de coordenadas que caracteriza a Geometria Analítica foi aplicado pela primeira vez em 1637 pelo matemático francês René Descartes (1596-1650). Está é a razão de a Geometria Analítica muitas vezes ser chamada de Geometria Cartesiana (LEHMANN, 1998).

De acordo com Lehmann (1998), sejam *P1(x1, y1)* e *P2(x2, y2)* dois pontos quaisquer dados, em geometria analítica, a distância entres eles pode ser calculada utilizando o Teorema de Pitágoras da seguinte forma:

$$d^2 = (x1 - x2)^2 + (y1 - y2)^2 \qquad (1)$$

donde

$$d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \qquad (2)$$

logo

$$d(P1,P2) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \qquad (3)$$

A fórmula supracitada, quando trabalhada com valores de coordenadas em uma superfície plana, resulta em uma reta. No entanto, para calcular a distância real entre dois pontos no globo terrestre, deve-se utilizar como valores de *x1*, *x2*, *y1* e *y2* as coordenadas reais dos pontos de origem e destino no globo terrestre, sendo *x1* e *y1* as coordenadas de latitude e longitude respectivamente do ponto de origem (*P1*) e *x2* e *y2* as coordenadas de latitude e longitude respectivamente do ponto de destino (*P2*).

Como latitude e longitude são valores angulares, o resultado da fórmula supracitada será em graus, o qual representa o arco da distância entre o ponto de origem e o ponto de destino.

Para obter uma distância que possa ser utilizada na resolução do problema de roteamento, é necessário fazer uma conversão do ângulo encontrado para uma distância em metros. Isto pode ser feito através de uma regra de três levando em consideração a afirmação de Friedmann (2009) que indica que os 360° da circunferência da Terra podem ser associados aproximadamente ao comprimento de 40.000 km.

O valor aproximado de 40.000 km associado aos 360° da circunferência da Terra pode ser confirmado utilizando a seguinte fórmula do comprimento da circunferência indicada por Dolce e Pompeo (1993):

$$C = 2\pi R \quad (4)$$

Segundo Fitz (2008), de acordo com o *Geodetic Reference System* 1980 (Sistema Geodésico de Referência, de 1980) o raio equatorial da Terra é 6.378.137 metros, logo tem-se:

$$C = 2 \times 3.141592 \times 6378137 \quad (5)$$

$$C(metros) = 40.075.008 \quad (6)$$

$$C(km) = 40.075 \quad (7)$$

Sendo *x* o ângulo que representa o arco da distância entre os dois pontos que foi obtido através do Teorema de Pitágoras e 40.075.008 o valor em metros aproximado da circunferência da Terra, tem-se a seguinte regra de três:

$$y = \frac{x \times C(metro)}{360^{o}} \quad (8)$$

Onde *y* é a distância em metros entre o ponto de origem e o ponto de destino.

3.2.2 Fórmula de Haversine

De acordo com Haversine (2012), a fórmula de Haversine é uma importante equação usada em navegação, fornecendo distâncias entre dois pontos de uma esfera a partir de suas latitudes e longitudes.

Na época anterior as calculadoras digitais, o uso de tabelas náuticas detalhadas para a haversine, haversine/inverso e seus logaritmos (para ajudar nas multiplicações) poupou aos navegantes calcular os quadrados dos seios, o cálculo de raízes quadradas, etc., um processo árduo e que podia agravar os pequenos erros (ENCYDIA, 2012).

A fórmula de Haversine considera a curvatura da Terra para indicar a distância entre dois pontos, no entanto, ela representa apenas uma aproximação, visto que esta assume como premissa que o formato da Terra é de uma esfera perfeita. Isto ocasiona distorções, pois o raio da Terra é variável: nos polos é da ordem de 6357 km; enquanto que no equador é de 6378 km. A imprecisão dos cálculos aumenta conforme nos afastamos da linha do equador (HAVERSINE, 2012).

Dados dois pontos quaisquer $P1(\phi 1, \lambda 1)$ e $P2(\phi 2, \lambda 2)$ em uma esfera de raio R, onde $\phi 1$ e $\lambda 1$ representam a latitude e longitude do ponto de origem e $\phi 2$ e $\lambda 2$ representam a latitude e longitude do ponto de destino, à distância entre estes dois pontos pode ser localizada utilizando a fórmula de Haversine expressa a seguir:

$$haversin\left(\frac{d}{R}\right) = haversin(\Delta\phi) + cos(\varphi 1)cos(\phi 2)haversin(\Delta\lambda) \quad (9)$$

A função *versin(θ)* significa que se está calculando o seno verso do ângulo θ. Já *haversin(θ)* significa que o que se está calculando é a metade do seno verso do ângulo θ (*half versine*). Segundo Encydia (2012), *haversin()* de um ângulo θ possui a seguinte relação:

$$havhaversin(\theta) = sen^2\left(\frac{\theta}{2}\right) \quad (10)$$

A variável $\Delta\phi$ representa a diferença em radiano das latitudes dos pontos $\phi 1$ e $\phi 2$. Da mesma forma $\Delta\lambda$ representa a diferença em radiano das longitudes dos pontos $\lambda 1$ e $\lambda 2$.

Conforme a fórmula de Haversine supracitada, a variável da distância *d* não está isolada, logo, é necessário isolá-la, afim de que a distância possa ser localizada. Para isto Encydia (2012) indica a utilização do arco seno da seguinte maneira:

$$d = R \times haversin^{-1}(h) = 2 \times R \times arcsin\left(\sqrt{h}\right) \quad (11)$$

onde

$$h = haversin\left(\frac{d}{R}\right) \quad (12)$$

3.2.3 Google Directions API

Mesmo que se obtenha um valor aproximado da distância entre dois pontos na superfície terrestre utilizando os paradigmas supracitados, não é possível utilizar este valor para solucionar um problema de ordem real. Isso se deve ao fato de que um problema real deve considerar questões de percurso da malha rodoviária em que o veículo circula e altitude do terreno em que se encontra a malha. Estes fatores influenciam diretamente na distância entre os pontos de entrega e, consequentemente, na solução do problema de roteamento.

Para o ajuste fino da distância entre os pontos visitados, pode-se utilizar as *Google Maps APIs* disponibilizadas pelo Google. Estas APIs oferecem todo o suporte necessário para os aspectos relacionados à georreferenciamento.

Uma destas APIs é a *Google Directions API*. Este é um serviço que calcula rotas entre os locais usando uma solicitação HTTP. Para o cálculo das rotas, é possível passar como parâmetro para o serviço *web*, tanto coordenadas de latitude/longitude como endereços do tipo texto (GOOGLE, 2012).

Além de especificar nos parâmetros de solicitação do serviço um ponto de origem e outro de destino, é possível adicionar ao cálculo do trajeto locais adicionais que serão considerados como obrigatórios na construção da rota.

Segundo Google (2012), a resposta da solicitação do serviço vem no formato de um entre os seguintes tipos de arquivos: JSON ou XML. Estes arquivos possuem estruturas internas totalmente diferentes, no entanto, seu conteúdo nada mais é do que os parâmetros que caracterizam a rota solicitada.

Apesar da grande quantidade de parâmetros retornados, somente a distância entre os pontos de origem e destino é necessária para a resolução do problema de roteamento, em virtude desta ser utilizada na carga dos pesos das arestas do grafo que representa os locais a serem visitados e os percursos que ligam estes locais.

Para a utilização das *Google Maps APIs* em modo comercial, deve-se levar em consideração as limitações na quantidade de requisições que atualmente não podem passar de 2.500 por dia para clientes sem cadastros no *Google Maps Premier*. Já para os clientes do *Google Maps Premier* o limite de requisições aumenta para 100.000 solicitações por dia. Também deve-se obedecer às restrições de licença encontradas nos termos de serviço das *Google Maps APIs*.

3.3 Resolução de problemas de roteirização

Não basta para uma correta resolução dos problemas de roteirização definir com precisão a distância entre os pontos visitados. É necessário também que o algoritmo que processará esta informação seja funcional a ponto de encontrar em um tempo computacional aceitável, soluções de alta qualidade, não necessariamente ótimas.

A solução ótima para a resolução de problemas de roteirização seria que este fosse submetido a um algoritmo exato que analisasse todas as rotas possíveis afim de que no final

desde processo, fosse indicada a melhor rota a ser seguida. No entanto, esta espécie de problema possui uma grande complexidade matemática (NP-Difícil), que inviabiliza a resolução desta por métodos exatos.

Conforme Cunha (1997, apud Cunha, 2000) a complexidade da resolução dos problemas de roteirização, bem como a sua relevância no contexto logístico atual, explicam o constante interesse em busca de novas estratégias de solução que vem sendo observada desde a década de 60, resultando em um número muito expressivo de artigos publicados na literatura especializada (Roteamento de helicópteros (TIMLIN; PULLEYBLANK, 1990), Roteamento de satélites (LEE et al., 2003)).

Uma das abordagens mais estudadas diz respeito à utilização de algoritmos probabilísticos, também conhecidos como algoritmos heurísticos, ou simplesmente heurísticas. Algoritmos heurísticos compõem o conjunto de algoritmos polinomiais que não possuem garantia nenhuma sobre a qualidade da solução encontrada, mas que usualmente tendem a encontrar a solução ótima ou ficar bem próximo a ela (Linden, 2008).

Uma das técnicas baseada em algoritmos heurísticos mais estudada é inspirada no processo genético e evolutivo dos organismos vivos, baseados na Teoria da Evolução de Darwin e que usam uma analogia direta com o comportamento natural da reprodução. Esta técnica, baseada em algoritmos evolucionários, chama-se algoritmos genéticos.

4 ALGORITMOS EVOLUCIONÁRIOS

Os algoritmos evolucionários podem ser vistos como técnicas de Computação Evolutiva (ARTERO, 2009), Computação Bioinspirada (LINDEN, 2008) ou Computação Natural (CASTRO, 2010). Nessa família de algoritmos, encontram-se todos os algoritmos cujas técnicas são baseadas na teoria evolucionária proposta por Charles Darwin em seu livro de 1859, "*On the Origin of Species by Means of Natural Selection*". Até a publicação por Darwin da teoria da evolução das espécies, os cientistas mais proeminentes acreditavam em uma dentre as teorias do criacionismo ou da geração expontânea (LINDEN, 2008).

Em meados do Século XIX, Charles Darwin e Alfred R. Wallace, dois pesquisadores ingleses, descobriram que os organismos vivos capazes de se reproduzir sexuadamente herdam algumas características de seus pais, que são combinadas através de mecanismos então desconhecidos. Eles também verificaram que as proles estão sujeitas a pequenas variações em suas características e que, em consequência disso, sofrem um processo de seleção natural, através do qual somente aquelas que estão mais adaptadas ao ambiente são capazes de sobreviver e se reproduzir (DARWIN, 1998 apud CASTRO, 2010).

De acordo com Goldbarg e Luna (2005), Charles Darwin desenvolveu uma teoria completa explicando como o meio ambiente atuaria sobre os indivíduos, forçando uma seleção dos mais aptos. Johann Gregor Mendel, em 1866, apresentou uma nova teoria para explicar como ocorreriam os fenômenos hereditários, o que viria a se transformar no que hoje é conhecido como genética, que segundo Castro (2010), apesar de inovadora, só pôde ser melhor compreendida depois que Mendel desvendou seus princípios.

Russel e Norvig (2004) indicam que a teoria de Darwin foi desenvolvida sem qualquer conhecimento de como as características dos organismos podem ser herdadas e modificadas. As leis probabilísticas que governam esses processos só foram identificas após Mendel conduzir experiencias com ervilhas usando o que ele denominou de fertilização artificial.

A teoria de Darwin baseia-se em alguns princípios bem definidos que, segundo Goldbarg e Luna (2005), são:

- Reprodução com herança: há uma população de indivíduos que se reproduzem;
- Variação: os filhos herdam algumas características de seus pais, mas também estão sujeitos a pequenas modificações.

- Seleção natural: o ambiente oferece um tipo de pressão seletiva que privilegia aqueles indivíduos mais bem adaptados.

A computação evolucionária é um campo da inteligência artificial que toma os princípios básicos da evolução e os aplica na forma de um programa de computador (WHITBY, 2003).

Conforme Linden (2008), apesar de haver uma grande variedade de modelos computacionais propostos, todos eles têm em comum o conceito de simulação da evolução das espécies através de seleção, mutação e reprodução. Processos estes que dependem do "desempenho" dos indivíduos de uma determinada espécie dentro do "Ambiente".

Dentre outros, um dos paradigmas da computação evolutiva que se baseia nos mecanismos evolutivos encontrados na natureza para solucionar uma gama de problemas propostos, são os Algoritmos Genéticos.

4.1 Algoritmos Genéticos

Os algoritmos genéticos são um ramo dos algoritmos evolucionários e como tal podem ser definidos como uma técnica de busca baseada na metáfora do processo biológico de evolução natural. (LINDEN, 2008).

Artero (2009) complementa indicando que os algoritmos genéticos são inspirados em modelos biológicos, usando uma medida para avaliar a capacidade que os indivíduos de uma população têm para sobreviver e se reproduzir. Com isto, espera-se que as características dos indivíduos vencedores, que se reproduzem, sejam transferidas para a próxima geração, enquanto que as características dos indivíduos com menos capacidade de sobrevivência, e consequentemente, de reprodução, sejam perdidas. Assim, em relação à característica que se está levando em consideração no momento de avaliar a capacidade de sobrevivência dos indivíduos, espera-se que sejam obtidas novas gerações cada vez mais próximas da perfeição em relação a esta característica.

Conforme Goldbarg e Luna (2005), os primeiros trabalhos nessa linha são originados de John Holland, e objetivaram replicar os processos utilizados pelos sistemas auto-adaptativos em um contexto computacional. Partindo do modelo de cromossomos para representar o processo evolutivo, Holland utilizou uma lista de símbolos binários (0,1) para representar as cadeias do ácido nucleico. Os algoritmos genéticos possuem as seguintes características gerais:

- Operam em um conjunto de pontos (denominado população) e não a partir de pontos isolados.
- Operam em um espaço de soluções codificadas e não diretamente no espaço de busca.
- Necessitam como informação somente o valor de uma função objetivo (denominada função de aptidão ou *fitness*).

De acordo com Linden (2008), os algoritmos genéticos funcionam mantendo uma população de estruturas, denominadas indivíduos ou cromossomos, que se comportam de forma semelhante à evolução das espécies. A estas estruturas são aplicados os chamados operadores genéticos, como recombinação, mutação, entre outros. Cada indivíduo recebe uma avaliação que representa uma quantificação da sua qualidade como solução do problema em questão. Com base nesta avaliação, serão aplicados os operadores genéticos de forma a simular a sobrevivência do mais apto.

A população, segundo Artero (2009), é formada por um conjunto de indivíduos que irão competir pela sobrevivência e pela reprodução, objetivando perpetuar suas características. No contexto dos algoritmos genéticos, população é um conjunto de indivíduos onde cada indivíduo corresponde a uma solução em potencial para o problema proposto.

A geração da população inicial de soluções candidatas é o primeiro passo para a resolução de um problema utilizando algoritmos genéticos. Linden (2008) afirma que a inicialização da população, na maioria dos trabalhos feitos na área, é realizada da forma mais simples possível, fazendo-se uma escolha aleatória, independente para cada indivíduo da população inicial. A lei das probabilidades sugere que a população inicial terá, desta forma, uma distribuição que cubra praticamente todo o espaço de soluções possíveis, mais isto não pode ser garantido, pois a população tem tamanho finito. Caso a solução não seja encontrada entre os indivíduos da população inicial, uma boa biodiversidade de soluções candidatas, de forma geral, garante que se encontre a solução do problema nas gerações seguintes.

Um indivíduo da população pode ser representado através do seu cromossomo. O cromossomo corresponde a um representação unívoca de um indíviduo. Linden (2008) afirma que a representação cromossomial é fundamental para um algoritmo genético. Basicamente, ela consiste em uma maneira de traduzir a informação de um problema em uma maneira viável de ser tratada pelo computador. Segundo Goldbarg e Luna (2005), o cromossomo de um indivíduo pode ser interpretado como uma configuração ou solução do problema. Artero

(2009) complementa indicando que os cromossomos correspondem a uma cadeia de genes que representam cada indivíduo da população. A figura a seguir demonstra a representação genética de um algoritmo genético.

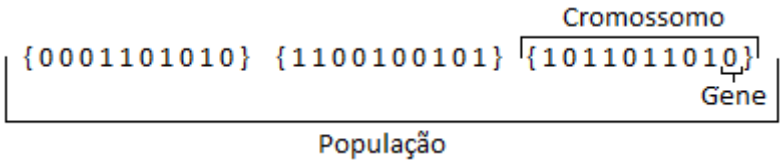

Figura 4.1 – Representação genética de um algoritmo genético.
Fonte: Do próprio autor.

Cada cromossomo é constituído de vários genes que, segundo Goldbarg e Luna (2005), representam um componente do problema ou então uma variável do problema. Estes pedaços indivisíveis da representação do cromossomo são chamados de gene, por analogia com as partes fundamentais que compõem um cromossomo biológico. (LINDEN, 2008).

Conforme Filitto (2008), a técnica para codificar os genes dos cromossomos varia conforme o tipo do problema. Pode-se utilizar uma cadeia de *strings* de bits (0, 1) para representar os cromossomos ou pode-se utilizar uma variável numérica contínua que armazena o seu próprio valor real.

Os processos evolutivos que ocorrem na natureza deram origem a Teoria Sintética da Evolução, ou neodarwinismo. Conforme Darwin já havia proposto, essa teoria considera a população como a unidade evolutiva. Uma população pode ser definida como um grupamento de indivíduos da mesma espécie que ocorre em uma mesma área geográfica, em um mesmo intervalo de tempo (BIOLOGIA, 2012). A Teoria Sintética da Evolução baseia-se nos seguintes fatores evolutivos: mutação, recombinação, seleção natural, migração e oscilação genética.

De acordo com Linden (2008), no contexto dos algoritmos genéticos, os operadores genéticos se consistem em aproximações computacionais de fenômenos na natureza, como a reprodução sexuada e a mutação genética. Goldbarg e Luna (2005) complementam indicando que os operadores genéticos são as regras que permitem a manipulação dos cromossomos.

Utilizando-se dos fatores evolutivos, um algoritmo genético tem como base gerar uma população inicial de soluções candidatas, e após isto, submeter esta população de soluções candidatas aos operadores genéticos: seleção natural, recombinação e mutação. A figura a seguir demonstra o fluxograma básico do funcionamento de um algoritmo genético.

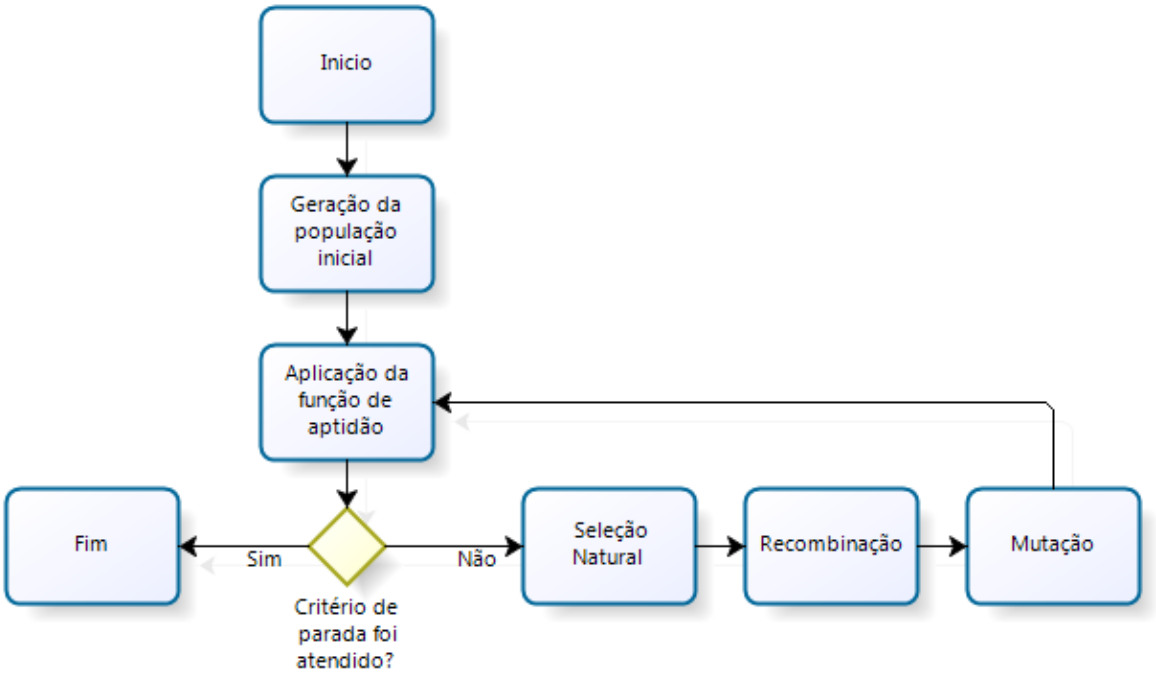

Figura 4.2 – Fluxograma do funcionamento de um algoritmo genético.
Fonte: Do próprio autor.

4.1.1 Seleção Natural

O objetivo principal do operador de seleção natural, no contexto dos algoritmos genéticos, é selecionar quais os indivíduos ou soluções candidatas que irão participar do processo de reprodução. Como no processo biológico, esta seleção deve ser feita de forma a privilegiar as soluções candidatas com maior aptidão. Para isto, é atribuído um valor para a solução candidata de modo que este valor represente a capacidade da solução candidata em se adaptar ao problema proposto. Este valor será utilizado para diferenciar as melhores soluções entre a população inicial de soluções candidatas. Esta avaliação é feita através da função de aptidão.

Segundo Goldbarg e Luna (2005), a função de aptidão ou *fitness* pode ser considerada como a medida de aptidão de um indivíduo. Linden (2008) complementa indicando que a função de aptidão é a maneira utilizada pelos algoritmos genéticos para determinar a qualidade de um indivíduo como solução do problema em questão. Pode-se entendê-la mais facilmente se for olhado para a função de aptidão como sendo a nota dada ao indivíduo na resolução do problema. Artero (2009) confirma as informações supracitadas indicando que a função de aptidão é usada para medir a habilidade do indivíduo para sobreviver e se reproduzir.

O método de seleção dos pais que serão utilizados na reprodução deve simular o mecanismo de seleção natural que atua sobre as espécies biológicas, em que os pais mais capazes geram mais filhos, ao mesmo tempo em que os pais menos aptos também podem gerar descendentes (LINDEN, 2008).

De acordo com Artero (2009), a seleção dos pais pode ser realizada de diferentes formas, porém, as propostas mais comuns são: seleção aleatória, seleção por torneio e seleção da roleta. Na primeira forma, são escolhidos de forma aleatória dois indivíduos da população para que estes cruzem e gerem as soluções candidatas filhos. Na seleção por torneio, são escolhidos, ao acaso, dois indivíduos da população e, o melhor deles, segundo a função de aptidão, será o escolhido como um dos pais para a reprodução. Este processo deverá ser repetido para se encontrar o segundo indivíduo a participar do cruzamento.

Segundo Filitto (2008), a forma mais conhecida de se fazer a seleção dos pais é a seleção por roleta. Nesta, cada indivíduo da população ocupa uma porção de uma roleta fictícia, proporcionalmente ao seu índice de aptidão. Com isto, os indivíduos que possuem uma alta aptidão ocuparão uma porção maior do que os indivíduos que possuem uma aptidão menor. Esta roleta é girada várias vezes, onde a quantidade de giros varia conforme o tamanho da população. Em cada giro da roleta é selecionado um indivíduo que participará do processo de geração da nova população.

4.1.2 Recombinação

Uma vez que todas as soluções estejam avaliadas, deve-se aplicar o fator de recombinação. Este fator caracteriza-se pelo fato de que as soluções mais capazes, ou seja, com maiores valores de aptidão, devem combinar-se entre si, gerando uma nova solução que substituirá a anterior.

Conforme Parreiras (2006), o operador de recombinação ou cruzamento simula o processo de reprodução sexuada como ocorre na natureza, em que os filhos são gerados a partir da combinação dos genes dos pais.

Goldbarg e Luna (2005) indicam que os genes dos indivíduos avaliados como bons se propagam através das populações, de modo que possam ser aperfeiçoados e gerem proles cada vez mais adequadas. Os indivíduos selecionados serão transformados por operadores genéticos em novos indivíduos.

Existem várias maneiras de realizar uma recombinação entre os indivíduos representados por cromossomos. Linden (2008) indica que a maneira mais simples é chamada de *crossover* de um ponto ou cruzamento de um ponto (FILITTO, 2008). Ainda existem o *crossover* multi-ponto, no qual mais de um ponto de corte é selecionado e o *crossover* uniforme onde não se utiliza pontos de cruzamento, mas se determina, através de um parâmetro global, qual será a probabilidade de cada ponto sofrer troca entre os pais (MITCHELL, 2001).

No *crossover* de um ponto, depois de selecionados dois pais através do operador de seleção natural, um ponto de corte é selecionado. Um ponto de corte constitui uma posição entre dois genes de um cromossomo e pode ser indicado de forma aleatória ou pré-fixada.

Após a seleção de um ponto de corte, os pais são separados em duas partes: uma à esquerda do ponto de corte e outra à direita. O primeiro filho é composto através da concatenação da parte esquerda do primeiro pai com a parte direita do segundo pai. O segundo filho é composto através da concatenação das partes que sobraram. A figura a seguir demonstra um cruzamento simples com posição aleatória.

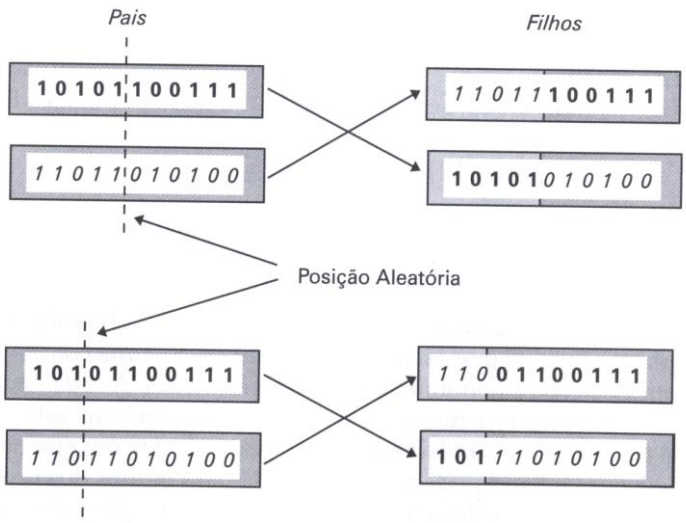

Figura 4.3 – Cruzamento simples.
Fonte: Goldbarg; Luna, 2005, p. 353.

A recombinação de soluções candidatas pode gerar novas soluções filhas com um ou mais genes repetidos. Para problemas de roteamento do tipo Caixeiro Viajante, onde os genes representam locais de entrega que só devem ser visitados uma única vez, a situação supracitada deve ser tratada de modo que esta não ocorra.

Para isto, pode-se aplicar uma multa (por exemplo acrescentando-se metros na distância total percorrida durante a entrega) nas soluções candidatas filhas que possuam locais

de entrega (ou genes) repetidos, de modo que na próxima geração, estas recebam avaliações baixas na função de aptidão e sejam automaticamente eliminadas do processo evolucionário.

Existe também a possibilidade de aplicação de técnicas de recombinação que não permitem que os genes dos cromossomos se repitam na solução candidata. Dentre outras, destacam-se os tipos de recombinação a seguir:

4.1.2.1 Partially Mapped Crossover (PMX)

A técnica PMX consiste em selecionar dois pontos de corte nos pais que estão sendo recombinados para que esta subsequência de genes possa ser copiada para a mesma posição nos filhos, preservando-se a ordem e a posição dos genes. Após isto, as subsequências entre os pontos de corte são trocadas entre os filhos e as posições que não possuem genes são completadas com valores que ainda não foram utilizadas, obedecendo a mesma ordem que se encontram nos pais.

Para demonstrar o funcionamento da técnica PMX considere os seguintes pais:

$$p1 = (9\ 4\ 3\ 8\ 1\ 2\ 5\ 7\ 6)$$

$$p2 = (1\ 3\ 5\ 7\ 9\ 2\ 4\ 6\ 8)$$

Após ser definido os pontos de corte e ser feita a troca das subsequências tem-se os filhos *f1* e *f2*:

$$f1 = (X\ X\ X \mid 7\ 9\ 2\ 4 \mid X\ X)$$

$$f2 = (X\ X\ X \mid 8\ 1\ 2\ 5 \mid X\ X)$$

Em seguida define-se o seguinte mapeamento entre os filhos: 7-8, 9-1, 2-2, 4-5. O próximo passo é substituir as posições que ainda não possuem gene (Elementos X) copiando-se os elementos de *p1* para *f1* que não estão entre os pontos de corte. No momento da cópia, deve-se levar em consideração o mapeamento elaborado de modo que nenhum elemento esteja repetido, ou seja, se o gene a ser copiado já estiver no cromossomo, utiliza-se o outro elemento associado a este no mapeamento. Ainda segundo o exemplo, tem-se:

$$f1 = (9\text{-}1\ 4\text{-}5\ 3 \mid 7\ 9\ 2\ 4 \mid 7\text{-}8\ 6) = (1\ 5\ 3\ 7\ 9\ 2\ 4\ 8\ 6)$$

$$f2 = (1\text{-}9\ 3\ 5\text{-}4 \mid 8\ 1\ 2\ 5 \mid 6\ 8\text{-}7) = (9\ 3\ 4\ 8\ 1\ 2\ 5\ 6\ 7)$$

4.1.2.2 Order Crossover (OX)

A técnica OX também consiste em selecionar dois pontos de corte nos pais para que a subsequência resultante de genes possa ser copiada para a mesma posição nos filhos preservando-se a ordem e a posição dos genes. Uma das diferenças entre esta técnica e a anterior está no fato de que os filhos gerados não irão trocar suas subsequência. O próximo passo consiste em, a partir do segundo ponto de corte de um dos pais, os genes do outro pai serem copiados para o filho na mesma ordem que estes se encontram e ignorando-se os genes que já estão presentes entre os pontos de corte. Utilizando-se os mesmos pais (*p1* e *p2*) da técnica anterior, tem-se:

$$f1 = (X\ X\ X \mid 8\ 1\ 2\ 5 \mid X\ X)$$

$$f2 = (X\ X\ X \mid 7\ 9\ 2\ 4 \mid X\ X)$$

Os filhos *f1* e *f2* herdaram a mesma subsequência dos pais *p1* e *p2*, respectivamente. O próximo passo é, partindo-se do segundo ponto de corte de *p2*, copiar os genes em ordem para *f1*, ignorando-se os genes que estão entre os pontos de corte. Da mesma forma, deve-se copiar em ordem para *f2* os genes de *p1* que se encontram após o ponto de corte. Desta forma, tem-se os seguintes resultados:

$$f1 = (7\ 9\ 4 \mid 8\ 1\ 2\ 5 \mid 6\ 3)$$

$$f2 = (8\ 1\ 5 \mid 7\ 9\ 2\ 4 \mid 6\ 3)$$

Tanto a técnica OX quanto a PMX evitam que os cromossomos possuam genes duplicados, sendo ideais para problemas de roteamento. No entanto, cromossomos com genes repetidos podem existir, sendo a sua aceitação definida de acordo com o tipo de problema no qual está sendo aplicado o algoritmo genético. Completada a reprodução por recombinação, a população pode ser submetida (não obrigatoriamente) a um processo de mutação (GOLDBARG; LUNA, 2005).

4.1.3 Mutação

O último fator da evolução a ser aplicado é a mutação. Este tem por objetivo alterar uma característica da solução resultante do processo de recombinação, a fim de aumentar a diversidade da população.

De acordo com Filitto (2008), o operador de mutação é responsável pela inserção de pequenas mudanças aleatórias nos cromossomos dos filhos. Artero (2009) indica que nesta

etapa o cromossomo pode ser modificado em algumas de suas partes, assumindo características que não pertencem aos pais.

O operador do tipo mutação de bit é o operador mais fácil de se aplicar, podendo ser aplicado em qualquer forma de representação binária dos cromossomos. Este método gera uma probabilidade de mutação para cada bit do cromossomo. Caso esta probabilidade seja menor que uma probabilidade predeterminada, o operador age sobre o gene em questão, indicando para este um valor aleatório dentre os valores que podem ser assumidos pelo cromossomo (FILITTO, 2008).

Segundo Poli; Langdon; McPhee (2008), hoje em dia, a mutação é amplamente utilizada em algoritmos genéticos, especialmente em aplicações de modelagem. Apesar da mutação geralmente consistir em mudanças aleatórias nos valores de bit dos cromossomos, existem outros tipos de operador, onde, dentre outros, destacam-se: *Subtree mutation*, *Size-fair subtree mutation*, *Node replacement mutation*, *Hoist mutation*, *Shrink mutation*, *Permutation mutation*, *Mutating constants at random* e *Mutating constants systematically*.

4.1.4 Controle da população

O processo envolvendo seleção, recombinação e mutação é repetido por diversas gerações, de modo que a cada iteração as soluções mais aptas passem suas características às próximas gerações.

Uma geração corresponde a uma população em um certo período. No caso dos algoritmos genéticos, corresponde aos valores dos indivíduos da população obtidos em uma dada interação (ARTERO, 2009).

Conforme Linden (2008), por uma questão de simplicidade, é aconselhável que a quantidade de indivíduos da população não cresça se comparada com a quantidade de indivíduos da população inicial. Para isto, os pais têm que ser substituídos conforme os filhos vão nascendo. Sabe-se que a cada geração, dois pais geram dois filhos através do seu cruzamento. Estes vão sendo armazenados em um espaço auxiliar até que o número de filhos criados seja igual ao tamanho da população inicial. Após isto, todos os pais são descartados e os filhos substituem os pais como a população atual.

Em todas as etapas do algoritmo genético selecionam-se os indivíduos mais aptos para resolver um determinado problema, efetua-se o cruzamento destes indivíduos para gerar

uma nova população e aplica-se a mutação na mesma. Este processo é repetido até que se obtenha uma solução que satisfaça o critério de parada (FILITTO, 2008).

Frente à constatação de que a heurística dos algoritmos genéticos pode ser utilizada como forma de resolução dos problemas de roteamento, estes serão utilizados como parte da proposta de desenvolvimento de um protótipo que tem por objetivo gerar uma rota mínima de distribuição.

5 PROTÓTIPO DESENVOLVIDO

Como visto nos capítulos anteriores, o ERP SIGER permite que as ordens de carga sejam agrupadas e organizadas na ordem que devem ser entregues através do resumo de ordens e carga. Esta ordenação é feita de forma manual, o que faz com que esta tarefa nem sempre resulte em um percurso onde a distância percorrida pelo veículo de transporte, seja a menor possível.

Como forma de auxiliar na tomada de decisão de qual rota escolher para que a entrega das mercadorias seja maximizada, sugere-se, através deste trabalho que o ERP SIGER possua um algoritmo implícito em sua estrutura que indique uma rota mínima de entrega.

Frente à relevância do problema supracitado, bem como da complexidade de resolução deste problema de ordem real, foi optado pela utilização do paradigma meta-heurístico algoritmo genético como forma de determinar a melhor rota de entrega das mercadorias.

A proposta deste trabalho consiste na construção de um protótipo que busque simular o algoritmo genético que será, posteriormente a esse trabalho, integrado com o ERP SIGER. Em uma aplicação comercial, a integração entre ERP e algoritmo genético deve ser online e encapsulada, no entanto, como se trata de um protótipo, este será desenvolvido separadamente do ERP SIGER, sendo a sua carga feita utilizando dados importados para o protótipo.

Uma vez que o protótipo seja desenvolvido, será efetuado um estudo de caso considerando uma situação real de entrega, no qual o protótipo será aplicado em busca da rota mínima de distribuição para o problema proposto. Neste estudo de caso, o protótipo será aplicado ao problema contendo diferentes configurações de amplitude, afim de que se possa estabelecer uma comparação entre estas. Através destas comparações será possível estabelecer a eficiência e a eficácia da utilização de algoritmos genéticos como forma de resolução para problemas de roteamento.

O protótipo tem como objetivo o estudo dos tópicos relacionados aos algoritmos genéticos e será desenvolvido para fins acadêmicos, sendo este restrito somente ao uso educacional ou para pesquisas não comerciais. Desta forma, a integração com o ERP SIGER será tratada como trabalho futuro, pois envolverá a equipe da empresa.

5.1 Visão geral

O protótipo será desenvolvido utilizando a linguagem de programação JAVA. A escolha deste paradigma de programação se deu em virtude dessa ser extremamente simples e totalmente orientada a objeto. De acordo com Linden (2008), a construção de programas verdadeiramente orientados a objetos permite que o código seja reutilizável, além de simples e seguro. A interface gráfica está sendo construída utilizando-se a API gráfica *Swing*. A figura a seguir demonstra a janela inicial do protótipo.

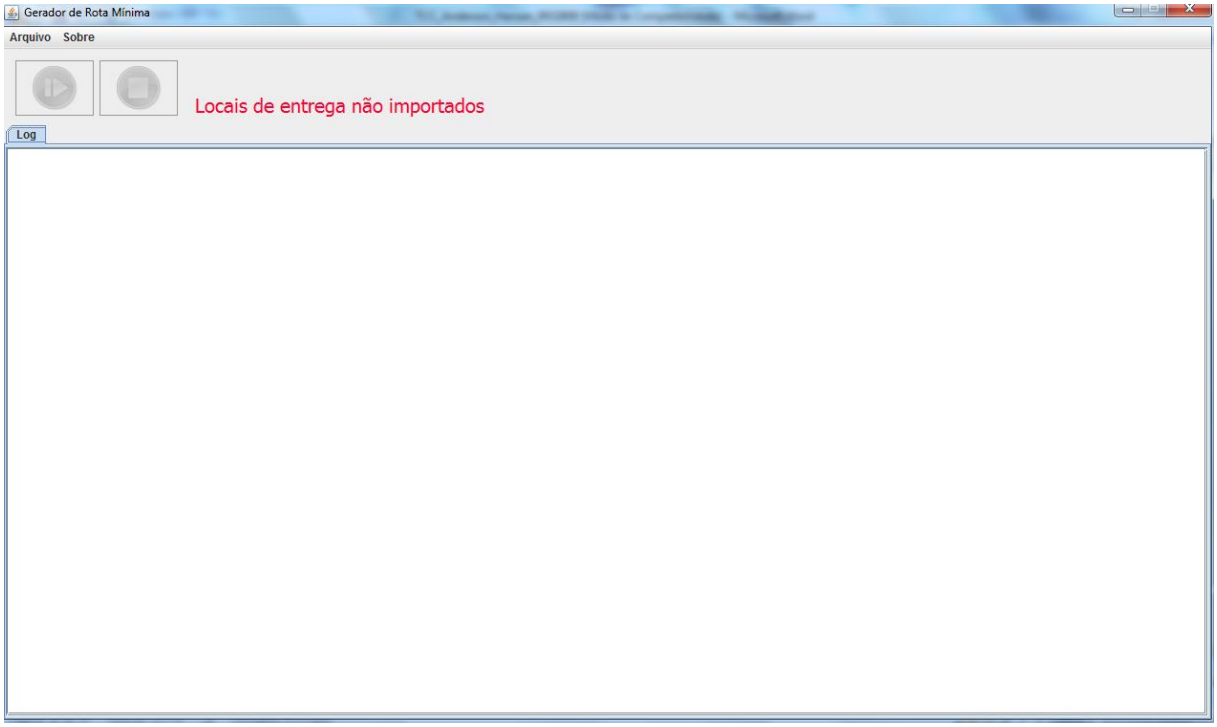

Figura 5.1 – Janela inicial do protótipo.
Fonte: Do próprio autor.

Como supracitado, o protótipo desenvolvido não será, neste momento, encapsulado no ERP e foi construído separadamente deste, logo, os dados necessários para a geração da rota mínima de distribuição serão importados para o protótipo. Os únicos dados a serem importados são os endereços de entrega que devem ser, obrigatoriamente, visitados durante o processo de entrega das mercadorias. Esses deverão ser informados em um arquivo com extensão TXT, sendo reservada uma linha para cada endereço de entrega.

Para importar o arquivo com os endereços é necessário se dirigir ao menu “Arquivo”, e acionar o submenu “Importar Locais Entrega”. Uma vez que este menu tenha sido acionado, será aberta uma janela para selecionar o diretório e o arquivo contendo os endereços de entrega.

Quando o arquivo é importado, cada endereço recebe um ID (IDentificação) que será o identificador do endereço durante o processo de geração da rota mínima de distribuição. A geração do ID se dará de forma crescente pela ordem em que os endereços estão informados no arquivo TXT e, consequentemente, pela ordem que serão importados. A figura 5.3 demonstra os endereços já importados com os seus respectivos IDs.

Com os endereços importados, é necessário configurar as informações que darão a amplitude de busca em que o algoritmo genético irá se basear para gerar a rota mínima. A janela onde é possível informar as configurações encontra-se no menu “Arquivo”, submenu “Configuração”. Nesta janela, é possível informar a quantidade de gerações, quantidade de indivíduos da população e quantidade de locais de entrega. A figura a seguir demonstra a janela de configuração.

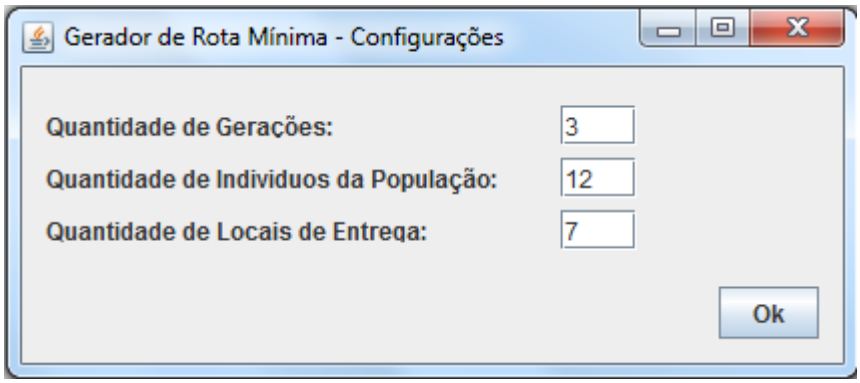

Figura 5.2 – Janela de configuração.
Fonte: Do próprio autor.

A quantidade de gerações determinará quantas iterações o algoritmo genético executará em busca da rota mínima de distribuição. Em cada iteração, o algoritmo genético aplicará a função de aptidão nas soluções candidatas, selecionará as mais aptas, aplicará o fator de evolução mutação, e verificará se as soluções candidatas resultantes atendem ao critério de parada.

No caso do protótipo desenvolvido, o critério de parada é a própria quantidade de gerações, ou seja, o algoritmo genético somente interromperá o processo de geração da rota mínima quando a quantidade de iterações for igual à quantidade de gerações informada. Apesar do critério de parada definido para o protótipo ser o supracitado, existem outros que poderiam ser aplicados, como é o caso do critério de parada por eficiência. Neste, o algoritmo é interrompido quando a eficiência (indicada pela função de aptidão) das soluções candidatas for superior a da população inicial.

A quantidade de indivíduos da população determina a quantidade de soluções candidatas que cada geração possuirá. Esta quantidade não sofre mudança no decorrer das gerações, e corresponde a entidade “cromossomo” da analogia indicada na figura 4.1. A quantidade de indivíduos da população informada na janela de configuração deve ser sempre a quantidade de endereços que serão importados mais um. A necessidade desta adição se dá, pois esta representa o acréscimo do local de partida/chegada que deve ser considerado na geração da rota mínima. Este local de partida/chegada representará a analogia de uma entrega que sai de um centro de distribuição, por exemplo, e após efetuar as entregas retorna ao ponto de origem.

Já a quantidade de locais de entrega indica quantos endereços de entrega cada solução candidata possuirá. Cada local de entrega corresponde a um endereço a ser visitado durante a entrega. O local de entrega corresponde à entidade “gene” na analogia indicada na figura 4.1.

Uma vez que os endereços de entrega estejam importados e o protótipo esteja configurado de acordo com a amplitude que se deseja utilizar, o algoritmo genético está pronto para gerar as populações de soluções candidatas afim de simular o processo evolutivo em busca da rota mínima de distribuição. A figura a seguir demonstra o protótipo com os endereços de entrega já importados e pronto para a utilização.

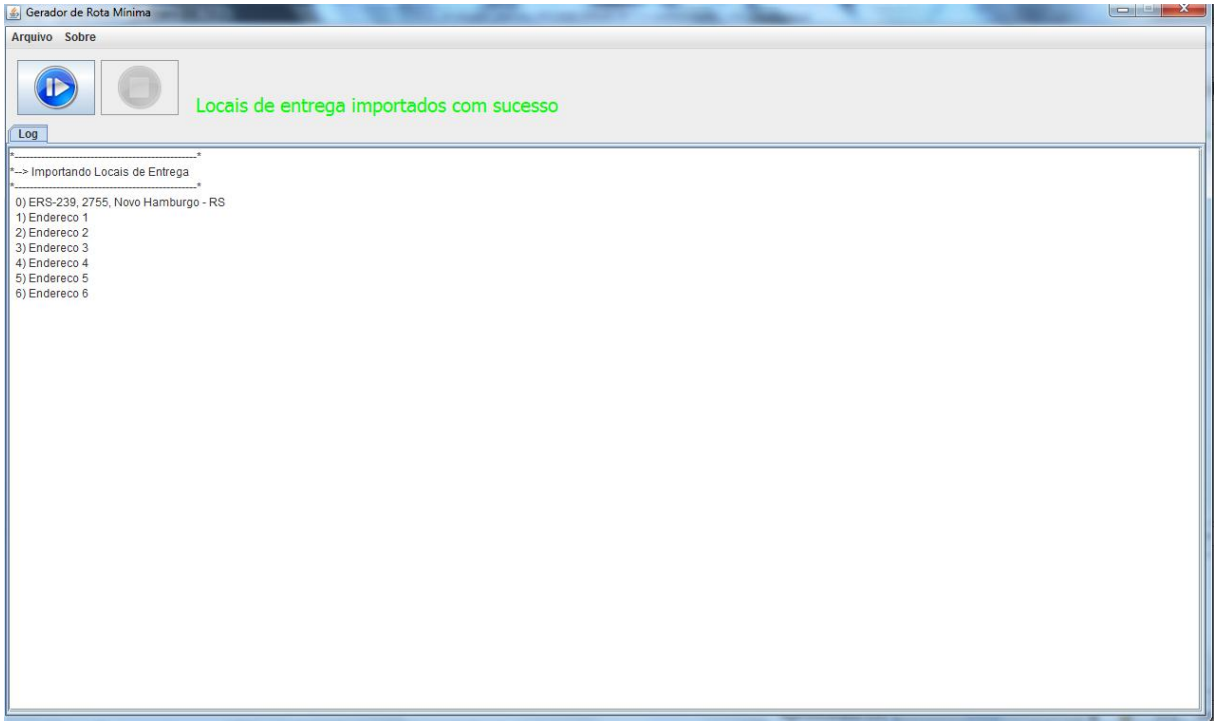

Figura 5.3 – Protótipo com os endereços importados e pronto para utilização.
Fonte: Do próprio autor.

5.2 Gerando as populações

Para a geração da rota mínima de distribuição, primeiramente gera-se uma população de soluções candidatas inicial. Essa será criada de forma que simule a passagem por todos os endereços de entrega que foram importados. A ordem de visitação nos endereços de entrega é feita de forma aleatória para cada solução candidata, e cada solução candidata terá uma ordem de visitação diferente das demais.

Para iniciar o algoritmo genético é necessário acionar o botão de "iniciar" que está localizado na janela inicial do protótipo. O protótipo desenvolvido também possui um botão que finaliza o algoritmo genético. Esse se localiza ao lado do botão de "iniciar". Se o botão de finalizar for acionado antes da conclusão do algoritmo genético, a rota mínima de distribuição não será gerada.

Como forma de abstrair computacionalmente o conceito de população, será utilizada a classe Java *ArrayList*, de modo que cada objeto *ArrayList* criado seja a abstração de uma população em um determinado período. A classe *ArrayList* disponibiliza métodos para manipular elementos via seus índices. Para criar um objeto *ArrayList*, é necessário instanciar um objeto da classe, sendo permitindo passar ou não como parâmetro no construtor da classe a quantidade inicial de elementos que a lista deve possuir. Um objeto pode ser definido como uma entidade cujas propriedades podem incluir desde suas características até suas atividades.

Como dito anteriormente, a classe *ArrayList* manipula elementos que são inseridos nesta. Como o *ArrayList* é a abstração da população, os elementos que devem ser inseridos neste, no contexto dos algoritmos genéticos, são as soluções candidatas. No protótipo desenvolvido, a abstração das soluções candidatas será feita utilizando-se de objetos do tipo solução candidata. A quantidade de objetos, ou instâncias da classe solução candidata, que serão criados, equivale à quantidade de indivíduos da população informada na janela de configuração. Após a criação das soluções candidatas, estas serão alocadas no *ArrayList* chamado "SolucoesCandidatas".

Conforme a figura 4.2, que demonstra o fluxograma do funcionamento de um algoritmo genético, a partir de uma população inicial de soluções candidatas, são aplicados os operadores genéticos seleção natural, recombinação e mutação, até que o critério de parada seja atendido. Cada uma destas iterações simula o processo natural da reprodução de indivíduos com o intuito de gerar uma nova população. Estas iterações serão feitas no

protótipo desenvolvido dentro de uma classe java que permite a execução de processos concorrentes. Esta classe chama-se *Thread*.

Segundo Deitel (2005), a linguagem Java disponibiliza a concorrência para o programador de aplicativos por meio de suas APIs. O programador especifica os aplicativos Java que contêm *threads* de execução, em que cada *thread* designa uma parte do programa que pode executar concorrentemente com outras *threads*. Essa capacidade, chamada *multithreading*, fornece opções poderosas para o programador Java.

A figura a seguir demonstra o método responsável pela criação da população inicial de soluções candidatas e pelo início da simulação da seleção natural, onde, para isso, é executado o *start* na *thread* da seleção natural.

```
// Inicia a seleção natural
    public static void IniciaSelecaoNatural(MainFrame mf, int QTDE_GER, int QTDE_POP, int QTDE_LOC) {
// Cria o ArrayList das rotas candidatas
        ArrayList<SolucaoCandidata> SolucoesCandidatas = new ArrayList<SolucaoCandidata>();
// Gera a população inicial e carrega o ArrayList das soluções canditadas
        SolucoesCandidatas = GeraPopulacaInicial(mf, SolucoesCandidatas, QTDE_POP, QTDE_LOC);
// Cria a thread de seleção natrual
        selecao = new SelecaoNatural(SolucoesCandidatas, QTDE_POP, QTDE_LOC, QTDE_GER, mf);
// Inicia a seleção natrual
        selecao.start();
    }
```

Figura 5.4 – Método responsável pela criação da população inicial de soluções candidatas.
Fonte: Do próprio autor.

Ao se chamar o método *start()* de uma determinada *thread,* este requisita os recursos do sistema necessários para que a *thread* seja executada e após inicia a execução da mesma chamando o método *run()* do objeto. A interrupção de uma *thread* pode acorrer pelo término do método *run(),* ou pela chamada do método *stop().*

A figura 5.5 demonstra a implementação do método *run()* contendo o laço principal da *thread* que simula os operadores genéticos. No laço principal da *thread,* cada iteração representa uma geração. Como critério de parada do laço que simula a seleção natural, este será executado gerando sucessivas gerações até que a quantidade de gerações seja igual a variável *threadQTDE_GER.* Esta representa a quantidade de gerações informada na janela de configurações do protótipo.

A cada iteração do laço, a população de soluções candidatas (variável *threadSolucoesCandidatas*) passará, dentre outros, pelos métodos *AplicaFuncaoAptidao(),* que aplicará a função de aptidão nas soluções candidatas e simulará o operador genético seleção natural, pelo método *CombinaSolucoes()* que combina as melhores soluções de

acordo com a função de aptidão, e pelo método *AplicaMutacao()* que irá aplicar uma pequena mutação nas soluções obtidas no método anterior.

```
// Executa a thread de seleção natural
    public void run(){
// Exibe título
        mf1.add("*-------------------------------------------------* \n");
        mf1.add("*--> Seleção Natural \n");
        mf1.add("*-------------------------------------------------* \n");
// Laço principal (executa até o fim da thread)
// OBSERVAÇÃO: Cada iteração representa uma geração
        for (int p=0; p < threadQTDE_GER; p++){
// Exibe a literal da geração
            ExibeGeracao(p);
// Aplica a função de aptidão nas soluções candidatas
            AplicaFuncaoAptidao();
// Avalia a média das notas das soluções candidatas da geração
            CalculaMediaSolucoes();
// Ordena as soluções candidatas de acordo com as avaliações optidas através da função de aptidão
            OrdenaSolucoesCandidatas();
// Combina as melhores soluções
            CombinaSolucoes();
// Aplica Mutação
            AplicaMutacao();
        }
    }
```

Figura 5.5 – Método principal.
Fonte: Do próprio autor.

Dentre os métodos que cada população de soluções candidata irá passar em cada iteração, o mais importante é o método *AplicaFuncaoAptidao()*. Este irá avaliar cada solução candidata da população individualmente, de modo que se possa ordenar as soluções candidatas de acordo com a sua capacidade de adaptação ao problema proposto. No caso do protótipo desenvolvido, o problema proposto é localizar quais soluções candidatas possuem os menores trajetos de entrega.

5.3 Função de aptidão

A função de aptidão irá verificar a ordem de visitação proposta por uma solução candidata, e calcular a distância entre os pontos subsequentes. No final, serão somadas as distâncias entre os pontos, e este valor resultante será o valor de aptidão da solução candidata. Uma vez que se tenha calculada a aptidão de todas as soluções candidatas, deve-se ordenar estas soluções na ordem decrescente, de modo que a solução com menor valor (menor distância percorrida) seja considerada a melhor solução desta geração.

Como supracitado, cada solução candidata terá uma ordem de visitação dos endereços de entrega diferente das demais, ou seja, unívoca. Para que se possa atribuir o valor de aptidão da solução candidata, é necessário calcular a distância entre todos os pontos onde serão efetuadas as entregas de modo que a ordem proposta pela solução candidata seja

respeitada. O cálculo da distância dos pontos será sempre feita em pares, definindo-se sempre qual dos indivíduos do par é o ponto de origem e qual o ponto de destino.

Se, por exemplo, forem importados três endereços de entrega, estes irão receber os IDs 1, 2 e 3 respectivamente. O algoritmo genético pode indicar uma solução candidata que passe pelos endereços na ordem de ID 2, 3 e 1. A função de aptidão irá calcular a distância entre os IDs 2 - 3, 3 - 1 e 1 - 2. Isso irá simular o roteamento saindo do ponto de origem com ID 2 (um empresa de entrega, por exemplo), passando pelos pontos de ID 3 e 1, e após passar pelo ponto com ID 1, retornar ao ponto de origem com ID 2. Não serão passados todos os pontos da solução candidata de uma só vez para que a API calcule a distância total entre elas, pois as solicitações de rotas individuais da API podem conter até 8 pontos de referência intermediários na solicitação. Para um ERP que trata de entregas reais, 8 pontos de entrega é um número muito baixo. Para se calcular a rota entre os pontos será usada a *Google Directions API*.

Esta leva em consideração no seu cálculo de rota entre dois pontos a malha rodoviária por onde o veículo que efetuará o transporte irá passar. Por este motivo, a utilização deste serviço torna a distância entre os pontos mais precisa do que os métodos tradicionais que consideram a distância como uma linha reta entre os pontos. Por outro lado, questões como obras públicas na via, acidentes de trânsito, engarrafamentos e semáforos ao longo da via, não são levados em consideração no momento do cálculo, o que pode implicar no tempo total decorrente da entrega e na própria distância calculada.

A solicitação HTTP utilizada pela *Google Directions API* deve ser obrigatoriamente uma URL (*Uniform Resource Locator*) com a seguinte estrutura: "http://maps.googleapis.com/maps/api/directions/output?parameters", onde *output* indica o formato do arquivo que a resposta oriunda do serviço *web* irá retornar e *parameters* indica os parâmetros de solicitação para o serviço *web*. Basicamente, a *Google Directions API* define que os parâmetros de solicitação sejam dois, são eles:

- *Origin*: O endereço ou valor textual de latitude/longitude do ponto de origem.
- *Destination*: O endereço ou valor textual de latitude/longitude do ponto de destino.

Na importação dos endereços de entrega, foi atribuído um ID para cada endereço. É através desde ID que os endereços de origem e destino serão localizados para a montagem da URL. Utilizando-se do exemplo anterior, onde uma solução candidata indicou que a ordem de

visitação dos pontos de entrega seria ID 2, 3 e 1, a primeira requisição ao serviço irá considerar na URL o endereço do ID 2 como valor para o parâmetro "O*rigin"* e o endereço do ID 3 como valor do parâmetro "*Destination*". Após esta requisição, será feita outra requisição, sendo o endereço do ID 3 o parâmetro "O*rigin*" e o endereço do ID 2 como "*Destination* ".

A tabela a seguir apresenta dois IDs com os seus respectivos endereços.

Tabela 5.1 – Exemplo de IDs com seus respectivos endereços.

ID	Local	Endereço
1	Prefeitura de Novo Hamburgo	Rua Guia Lopes, 4201, Canudos, Novo Hamburgo - RS
2	Prefeitura de Estância Velha	Rua Presidente Lucena, 3454, Estância Velha – RS

A figura a seguir apresentada a URL que será usada para requisitar a distância entre os IDs da tabela anterior.

```
http://maps.googleapis.com/maps/api/directions/json?origin=Rua%20Guia%20Lopes,%204
201,%20Canudos,%20Novo%20Hamburgo%20-
%20RS&destination=Rua%20Presidente%20Lucena,%203454,%20Est%C3%A2ncia%20V
elha%20-%20RS&sensor=false
```

Figura 5.6 – Solicitação HTTP que calcula a rota entre os IDs da tabela 5.1.
Fonte: Do próprio autor.

O parâmetro *output* é obrigatório e de vital importância pois as respostas oriundas do serviço web serão retornadas no formato indicado pelo parâmetro *output* no caminho da solicitação HTTP. O parâmetro *output* pode ser substituído por um dos seguintes valores: JSON (*JavaScript Object Notation*) ou XML (*Extensible Markup Language*). Segundo GOOGLE (2012), o formado recomendado para a geração de rota é o formato JSON.

5.3.1 JSON

JSON é um acrônimo para "*JavaScript Object Notation*". Este formato é uma estrutura de dados utilizada para intercâmbio de dados computacionais. Apesar de sua utilização ter sido generalizada com a popularização do conjunto de tecnologias AJAX (*Asynchronous Javascript And XML*), o formato JSON não pode ser considerado uma

tecnologia nova, visto que, esta deriva da linguagem de *script JavaScript*. Apesar de o JSON ser nativo do *JavaScript*, este não obriga a utilização de nenhum script para trabalhar, logo, diversas linguagens já possuem suporte a JSON, tendo a sua utilização disseminada por diversas implementações nas mais diversas linguagens.

Cada vez mais o formato JSON está sendo adotado como padrão para os serviços disponíveis na internet utilizados no tráfego de dados em ambientes heterogêneos via solicitações HTTP (*Hypertext Transfer Protocol*). Esta popularidade crescente do JSON se deve principalmente a sua simplicidade em comparação com a sua principal alternativa, o XML. Além da simplicidade de geração e leitura, o formato JSON possui uma sintaxe muito simples e sucinta. Estas características tornam seu transporte mais leve e rápido, além de mais rápido e eficiente o seu processamento.

O formato JSON se baseia basicamente no conceito de atributo – valor, e possui basicamente duas estruturas:

- Objeto: Pode ser representado como uma coleção de pares de nome/valor. Cada objeto inicia pelo caractere "{" (chave esquerda) e é finalizado pelo caractere "}" (chave direita). A atribuição de valores para o nome se dá através do caractere ":" (dois pontos).
- *Array*: Pode ser representado por uma coleção de valores. Cada *array* inicia pelo caractere "[" (colchete esquerdo) e é finalizado pelo caractere "]" (colchete direito).

A figura a seguir demonstra a estrutura básica de um JSON que possui um objeto "veiculos" e um *array* implícito neste contendo o conjunto de carros.

```
{"veiculos" : [
   {"nome":"gol",   "ano":2004, "cor":"preto" },
   {"nome":"palio", "ano":2005, "cor":"branco" },
   {"nome":"uno",   "ano":2006, "cor":"vermelho" }
]};
```

Figura 5.7 – Estrutura básica de um JSON.
Fonte: Do próprio autor.

O arquivo JSON retornado através do serviço disponibilizado pela *Google Directions API* possui como estrutura principal uma matriz chamada "*routes*". Mesmo se o serviço não retornar resultados (por exemplo, se a origem e/ou destino não existir), ainda assim ele

retornará uma matriz vazia de *routes*. Cada elemento da matriz de *routes* contém um único resultado da origem e do destino especificados. Dentro da matriz *routes* é encontrada a submatriz "*legs*" que contém as informações do trajeto requerido. Dentre as informações disponibilizadas pela submatriz *legs* está o atributo "*distance*". Este expressa em metros a distância entre o local de origem e o local de destino passados como parâmetro na chamada do serviço (GOOGLE, 2012). A figura a seguir demonstra parte do resultado JSON obtido da solicitação HTTP exemplificada na figura 5.6, demonstrando que a distância entre os IDs é de 10,6 km.


```
{
   "routes" : [
      {
         "bounds" : {
            "northeast" : {
               "lat" : -29.64447000000001,
               "lng" : -51.117640
            },
            "southwest" : {
               "lat" : -29.694080,
               "lng" : -51.17376000000001
            }
         },
         "copyrights" : "Dados cartográficos ©2012 Google, MapLink",
         "legs" : [
            {
               "distance" : {
                  "text" : "10,6 km",
                  "value" : 10648
               },
               "duration" : {
                  "text" : "19 minutos",
                  "value" : 1116
               },
               "end_address" : "R. Pres. Lucena, 3454 - Estância Velha -
RS, 93600-000, Brasil",
               "end_location" : {
                  "lat" : -29.64447000000001,
                  "lng" : -51.17169000000001
               },
               "start_address" : "R. Guia Lopes, 4201 - Mauá, Novo
Hamburgo - RS, 93410-340, Brasil",
               "start_location" : {
                  "lat" : -29.694080,
                  "lng" : -51.117640
               },
            }
         ],
      }
   ],
   "status" : "OK"
}
```


Figura 5.8 – JSON obtido através da *Google Directions API*.
Fonte: Do próprio autor.

5.4 Mutação

Como foi visto no capítulo anterior, a finalidade do operador genético mutação é alterar uma característica da solução resultante do processo de recombinação, a fim de aumentar a diversidade da população.

No protótipo proposto foi definido que a mutação será feita alterando-se o material genético de dois pontos do cromossomo a uma taxa de mutação de 5% da população. A taxa de mutação indica qual a possibilidade de uma dada solução candidata (cromossomo) ter seu material genético alterado em prol da diversidade genética. Isto garante que o espaço de busca fique estagnado e que todos os pontos do espaço de busca sejam verificados. Uma taxa de mutação muito alta implica que as soluções candidatas se tornem essencialmente aleatórias, já que devido a grande quantidade de mutações, elas não são resultantes do processo evolucionário e sim do acaso.

A figura a seguir demonstra o método responsável pela mutação.

```
// Aplica Mutação
    public void AplicaMutacao(){
// Cria os locais de entrega
        ArrayList<Integer> Numeros = new ArrayList<Integer>();
// Processa 100 números
        for (int i = 1; i <= 100; i++) {
            Numeros.add(i);
        }
// Varre as soluções candidatas
        for (int i=0; i < threadQTDE_POP; i++){
// Varre os locais de entrega
            for (int j = 0; j < threadQTDE_LOC; j++) {
// Embaralhamos os números
                Collections.shuffle(Numeros);
// Se essa rota caiu na regra dos 0,5% nas quais devemos aplicar a mutação
                if (Numeros.get(0) < 6) {
// Altera as locais de entrega 5 e 2
                    localAux = threadSolucoesCandidatas.get(i).getRota().get(2);
                    localAu1 = threadSolucoesCandidatas.get(i).getRota().get(5);
                    threadSolucoesCandidatas.get(i).getRota().set(2, localAu1);
                    threadSolucoesCandidatas.get(i).getRota().set(5, localAux);
                }
            }
        }
// Exibe a literal
        mf1.add("Aplicando mutação \n \n");
    }
```

Figura 5.9 – Método da mutação.
Fonte: Do próprio autor.

Para se indicar que uma determinada solução candidata pertence aos 5% da população na qual será aplicada a mutação, utilizou-se a *API Java Collections*. Esta API

fornece todos os meios necessários para que, de forma simples, possa-se agrupar outros objetos em um único.

No caso do operador genético mutação, onde se está interessado em verificar se a solução candidata pertence aos 5% da população na qual se irá aplicar a mutação, primeiramente monta-se a lista completa de elementos possíveis, neste caso de 1 a 100. Após reordena-se a coleção de forma aleatória utilizando-se o método *Collections.shuffle()*. Por fim, pega-se o primeiro elemento dessa coleção e verifica-se se este se enquadra no intervalo de 1 a 5. Caso se enquadre, aplica-se a mutação invertendo os locais de entrega 5 e 2.

5.5 Analisando os resultados

Uma vez que se tenha importado os endereços dos locais de entrega que se deseja utilizar na simulação, será habilitado o botão de "iniciar". No acionamento desse botão, todos os movimentos do algoritmo genético serão exibidos no *log* da janela principal. Com esse *log* será possível acompanhar todos os passos das suscetíveis gerações de populações de soluções candidatas em busca da solução vencedora, ou mais apta ao problema.

Como visto nos capítulos que trataram da base teórica do algoritmo genético, o primeiro passo do algoritmo genético é gerar uma população inicial de soluções candidatas. A figura a seguir demonstra a primeira parte do *log* que apresenta a população inicial gerada.

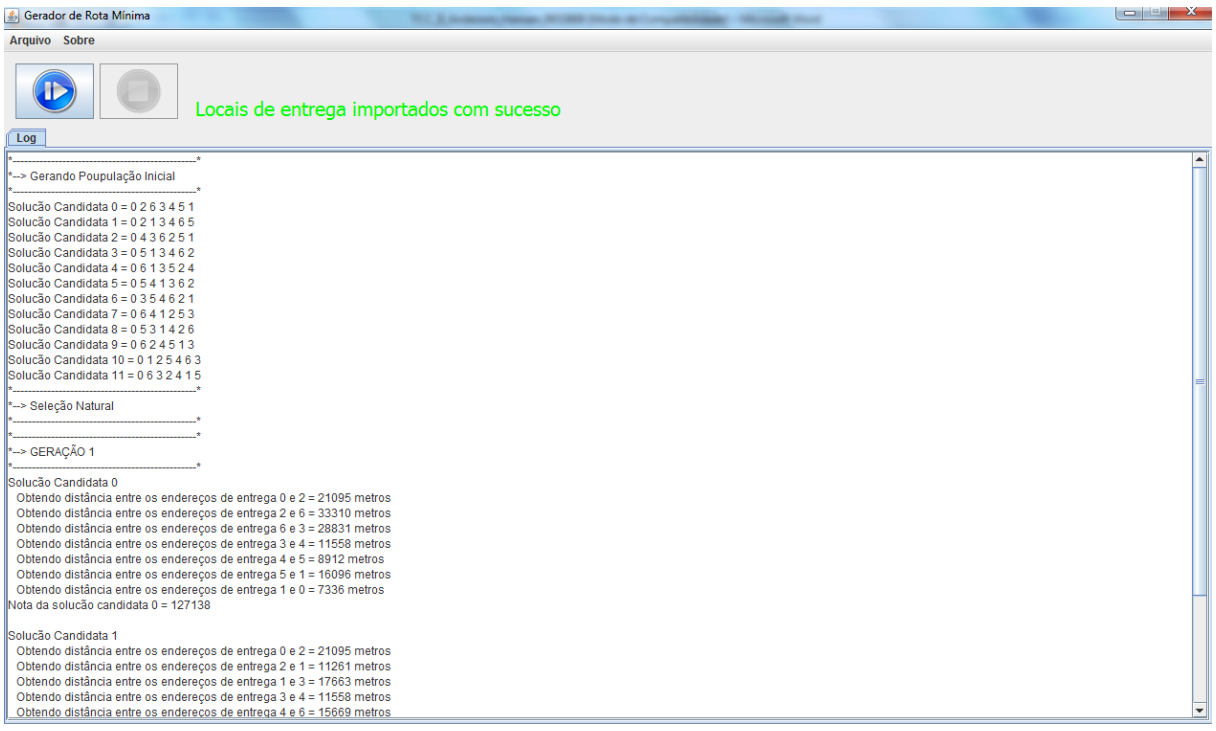

Figura 5.10 – Demonstração da geração da população inicial.
Fonte: Do próprio autor.

Na figura supracitada, foi demonstrada uma população inicial gerada para uma amplitude de busca de 12 indivíduos por população e cada indivíduo possuindo 7 locais de entrega. A sequencia numérica demonstrada, por exemplo, após a literal “Solução candidata 0”, representa a ordem de visitação desta solução. No exemplo da solução candidata 0, demonstra-se que esta solução sairá do local de entrega com ID 0 e passará pelos locais com IDs 2, 6, 3, 4, 5 e 1, para após voltar para o local de partida.

Após a população inicial ser gerada, no *log* será demonstrado o início da seleção natural através da literal “*--> Seleção Natural”. Na figura 5.10 pode ser verificado que após iniciar a seleção natural, o algoritmo genético iniciou a avaliação das soluções candidatas da primeira geração. É demonstrado nessa figura, por exemplo, que na solução candidata 0 da geração 1, o percurso entre os locais de entrega com ID 0 e 2 é de 21.095 metros.

Para que a obtenção da distância entre dois locais de entrega seja otimizada, o protótipo desenvolvido conta com uma função de *cache* das distâncias. Toda vez que for utilizada a *Google Directions API* para obter a distância entre dois pontos, será gravado em um *cache* o ponto de origem, o de destino e a distância entre eles. Desta forma, se uma determinada solução candidata possuir um trajeto em comum com a solução candidata que fez a requisição inicial, esta irá buscar a distância entre os pontos no *cache*, e não através de uma nova requisição. Isso evitará o desperdício de processamento. Outro ponto que culminou na implantação do *cache* foi o fato da *Google Directions API* possuir uma limitação diária de requisições que atualmente está em 2.500 requisições diárias. Para se implementar o *cache* foi utilizado a classe *HashMap*. Esta classe permite a inserção de chaves e valores, bem como fornece todas as operações opcionais de uma interface *map.*

Entre as requisições HTTP feitas para se obter a distância entre dois pontos, foi inserido um *delay* de 1 segundo, pois a *Google Directions API* não foi projetada para responder tantas requisições para um mesmo IP (*Internet Protocol*) em um intervalo tão curto de tempo. Quando a API detecta uma quantidade grande de requisições vinda de um único IP em um intervalo muito curto de tempo, esta interpreta as requisições como uma tentativa de congestionar o servidor (*Flood*) e bloqueia o serviço para o IP solicitante.

Ao final de cada geração será exibida no *log* a média da distância das soluções candidatas. Também serão exibidas literais indicando que o algoritmo genético aplicou a função de aptidão em todas as soluções daquela geração e passou pelos operadores genéticos de recombinação e mutação;

Quando o algoritmo genético processar tantas gerações quanto à quantidade que foi informada no campo “Quantidade de Gerações” na janela de configurações, o algoritmo irá interromper o processamento e indicar a solução candidata mais apta encontrada até então nas sucessivas gerações que simularam o processo biológico da reprodução sexuada e da transmissão de genes.

A figura a seguir demonstra o final de um exemplo de simulação, na qual foi indicado que o melhor caminho para a entrega das mercadorias é através dos locais de entrega cujos ID são 0, 4, 6, 5, 3, 2 e 1.

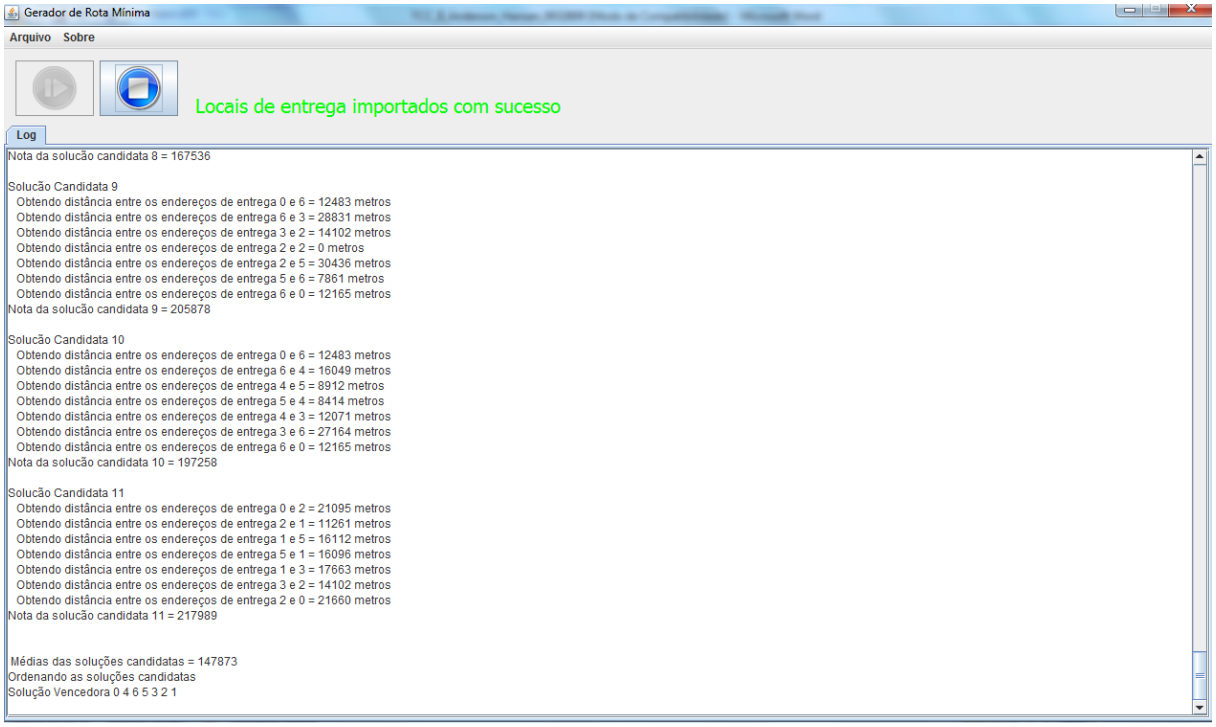

Figura 5.11 – Demonstração da geração vencedora.
Fonte: Do próprio autor.

6 ESTUDO DE CASO

Para avaliar os resultados obtidos, bem como a viabilidade comercial da proposta, será realizada uma simulação importando um arquivo TXT contendo seis endereços de entrega. Este arquivo TXT foi criado manualmente como forma de abstrair as informações que viriam do SIGER caso houvesse uma integração *on-line* entre o protótipo e o ERP.

Na simulação avaliada será considerada a Universidade Feevale como ponto de partida para as entregas, e obrigatoriamente a rota que será sugerida no final da execução do protótipo terá que ter passado por todos os seis endereços de entrega antes de retornar ao ponto de partida.

Não será utilizada na simulação a versão paga da *Google Maps APIs* que possibilita o acesso a 100.000 solicitações por dia ao serviço *web*. Por este motivo, optou-se por utilizar somente seis endereços de entrega na simulação, já que uma quantidade maior de endereços provavelmente implicaria na superação da quantidade de requisições da versão não paga.

Como locais de entregas serão utilizados os endereços das prefeituras das cidades de Estância Velha, Portão, São Leopoldo, Novo Hamburgo, Campo Bom e Sapiranga. Os endereços foram retirados dos *sites* das respectivas prefeituras e estarão no arquivo TXT que será importado. A sua ordem neste é irrelevante para o objetivo final.

A tabela a seguir apresenta as cidades com os respectivos endereços de suas prefeituras.

Tabela 6.1 – Endereços utilizados na simulação.

Prefeitura	Endereço
Estância Velha	Rua Guia Lopes, 4201, Canudos, Novo Hamburgo – RS
Portão	Rua Nove de Outubro, 229 - Portão - RS - 93180-000
São Leopoldo	Pc.Tiradentes, 119 - Centro, São Leopoldo - RS - 93010-020

Novo Hamburgo	Rua Guia Lopes, 4201, Canudos, Novo Hamburgo – RS
Campo Bom	Av. Independência - 800, Centro, Campo Bom - RS - 93700-000
Sapiranga	Av. João Correa, 793 - Centro - Sapiranga - RS - 93.800-000

Na janela de configuração do protótipo, onde se informa a amplitude desejada na simulação, será colocado no campo “Quantidade de Locais de Entrega” o valor 7. Este representa os seis endereços de entrega mais o ponto de partida. A quantidade de soluções candidatas que cada geração possuirá será informada no campo “Quantidade de Indivíduos da População” e será 24. Já no campo “Quantidade de Gerações” será informada que a simulação possuirá 10 gerações.

Com a amplitude da simulação configurada, os endereços da tabela 6.1 serão importados para o protótipo de modo que a simulação esteja pronta para iniciar. A figura abaixo demonstra os endereços importados.

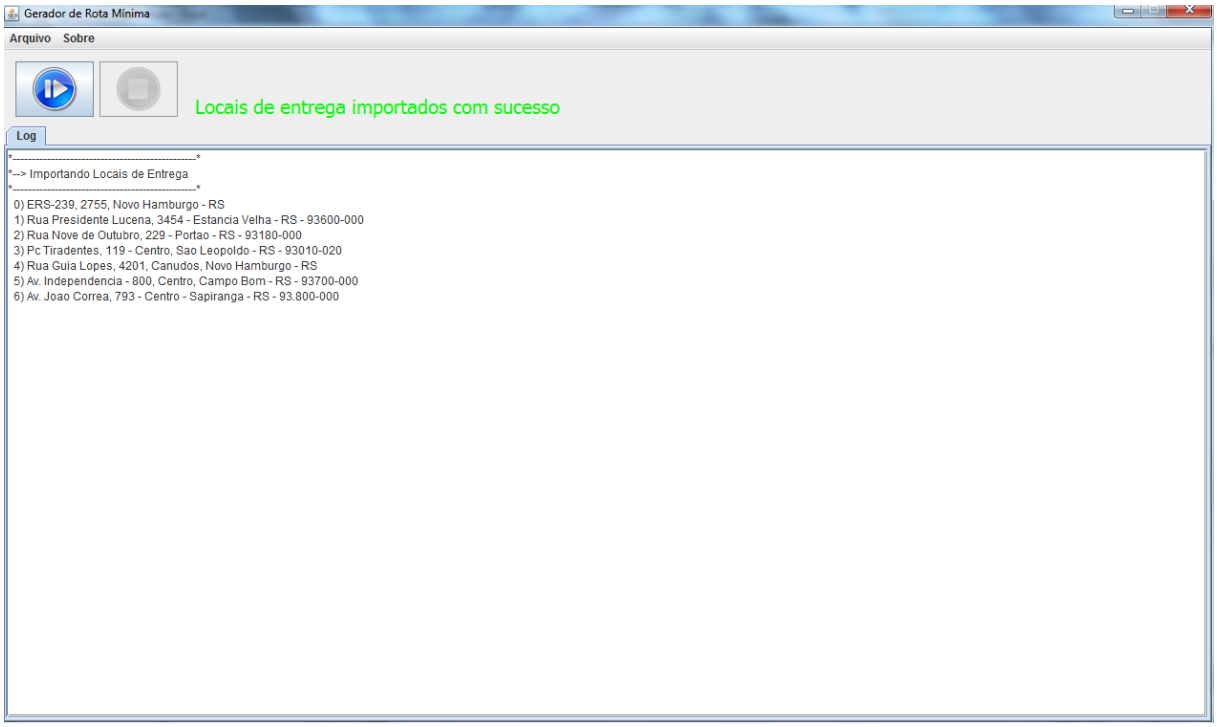

Figura 6.1 – Endereços importados que serão utilizados na simulação.
Fonte: Do próprio autor.

6.1 Resultados apresentados

Ao se iniciar a simulação, o protótipo gerou uma população inicial de soluções candidatas cuja média da distância entre todas as soluções foi de 114.019 metros. A figura a seguir demonstra a população inicial de soluções candidatas geradas na simulação estudada.

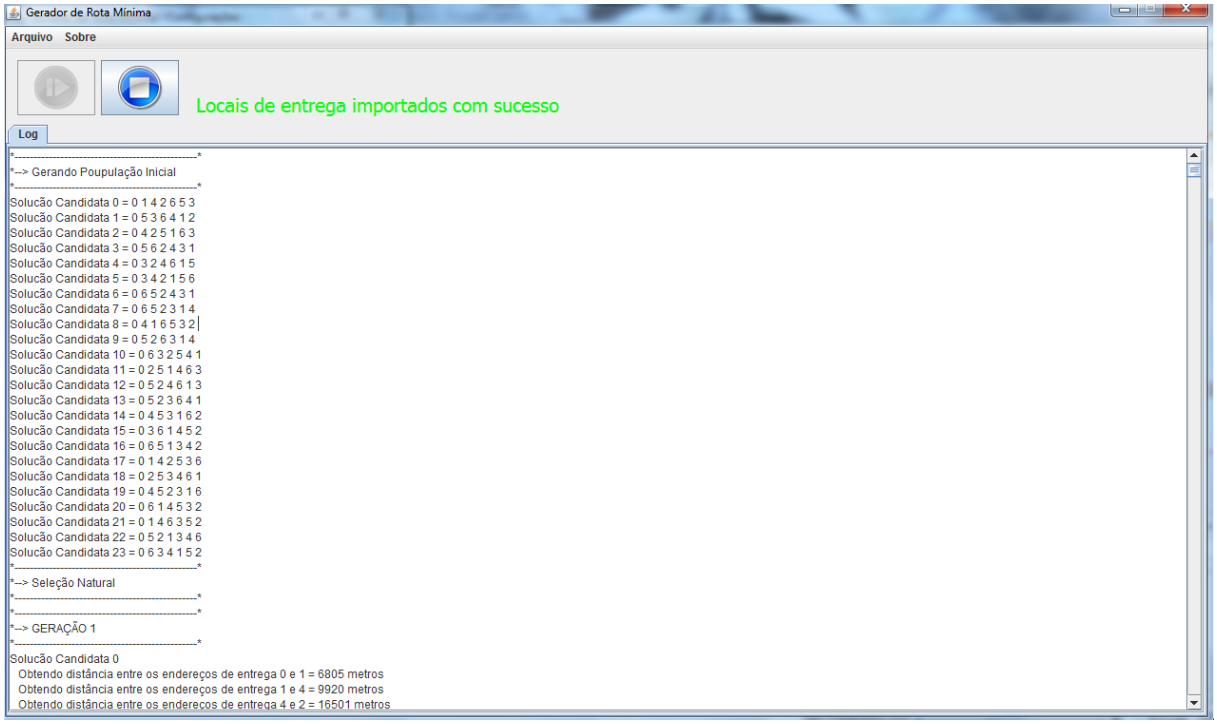

Figura 6.2 – População inicial da simulação.
Fonte: Do próprio autor.

Com a população inicial criada, o protótipo aplicou a função de aptidão em todas as 24 soluções candidatas iniciais, de modo que cada uma delas possuísse o seu valor de aptidão. Após calcular a função de aptidão de todas as soluções candidatas, o protótipo ordenou estas na ordem decrescente, de modo que a solução com menor valor (menor distância percorrida) seja considerada a melhor desta geração.

O próximo passo do protótipo foi descartar as 12 últimas soluções e recombinar as primeiras 12. Para a recombinação das soluções, pegou-se um par de soluções, e misturou-se suas características (neste caso a ordem de visitação), de modo que fossem geradas duas novas soluções contendo as características do par combinado. Deste modo, juntando-se as 12 primeiras soluções (pais), mais as 12 soluções geradas a partir das 12 primeiras (filhos), tem-se uma população de 24 soluções novamente para que seja simulada outra geração.

A solução ótima para o problema de roteamento proposto na simulação é que as entregas sejam feitas na seguinte ordem de ID: 0, 1, 2, 3, 4, 5, 6, 0. Esta ordem de visitação

resulta em um uma distância total de 73.809 metros e implicaria que, saindo da Universidade Feevale, as entregas sejam feitas passando-se pelas prefeituras de Estância Velha, Portão, São Leopoldo, Novo Hamburgo, Campo Bom e Sapiranga para após retornar para a Universidade Feevale. A tabela abaixo demonstra a distância entre cada local de entrega da solução ótima para o problema proposto.

Tabela 6.2 – Distância entre os locais de entrega da solução ótima.

Ponto de origem	Ponto de Destino	Distância (Metros)
0	1	6.805
1	2	10.844
2	3	15.664
3	4	11.558
4	5	8.912
5	6	7.861
6	0	12.165
	Total	73.809

Ao final da décima geração, o protótipo foi interrompido conforme as configurações feitas antes de se iniciar a simulação. A solução encontrada pelo protótipo para o problema de roteamento estudado indica que a ordem das entregas deve passar pelos seguintes IDs respectivamente: 0, 4, 1, 2, 3, 5, 6, 0. Esta solução possui uma distância total de 86.924 metros[1].

Apesar da solução encontrada pelo protótipo não ser a solução ótima demonstrada na tabela 6.2, ela pode ser considerada boa em virtude da proximidade com a solução ótima e da diferença entre a sua distância total e a distância total da pior solução encontrada após o final da décima geração.

[1] O LOG completo da simulação encontra-se em https://www.dropbox.com/sh/w8le74t4coa167h/y1SRggcfy3

O tempo que o protótipo levou para gerar a solução boa apresentada foi de um minuto e dez segundos. Este tempo não representa um padrão que possa ser considerado para todas as simulações, pois sabe-se que a cada simulação, o protótipo gera ao acaso uma população inicial de soluções candidatas diferente. Logo, mesmo uma configuração unívoca pode gerar um tempo decorrido total diferente em cada simulação.

O gráfico a seguir demonstra um comparativo da distância entre a solução ótima para resolver o problema estudado, a solução boa encontrada pelo protótipo ao final da execução do mesmo, e a pior solução da última geração encontrada no final da execução do protótipo. Na análise do gráfico deve-se levar em consideração que quanto menor a distância total da solução, mais apta para solucionar o problema ela é considerada.

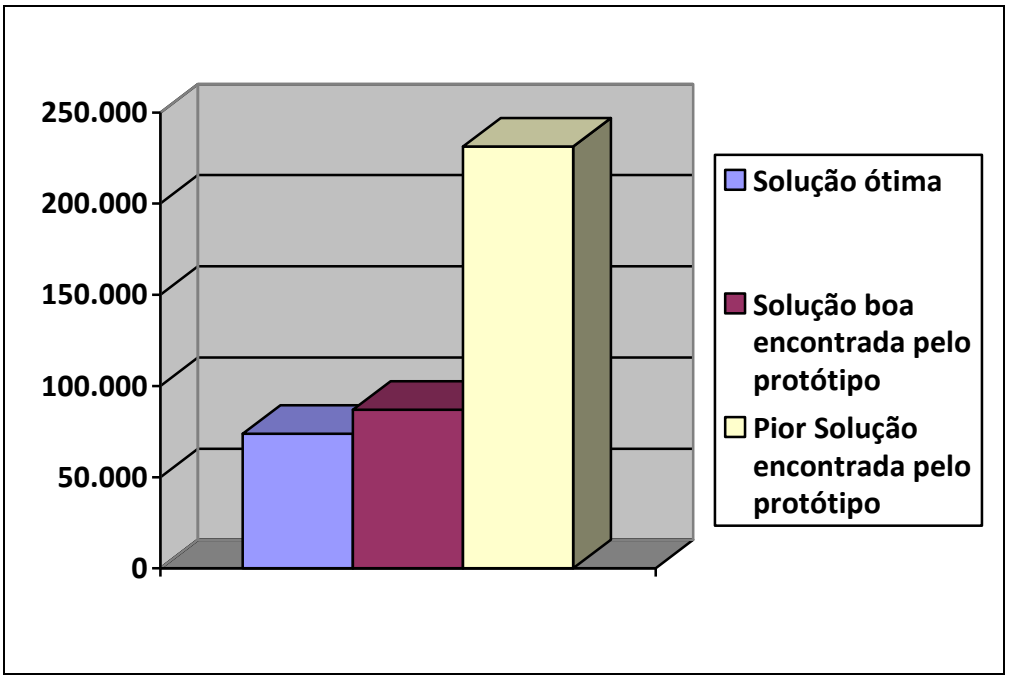

Figura 6.3 – Comparativo entre as soluções do problema proposto em metros.
Fonte: Do próprio autor.

CONCLUSÃO

Os sistemas de ERP são decisivos quando o assunto é tomada de decisão dentro das empresas. Para o sucesso de um ERP, este deve demonstrar resultados precisos e coerentes. Dentre os muitos aspectos que permeiam a gestão, o controle de cargas é um dos fatores cruciais para alcançar o objetivo de todas as empresas: o lucro.

Neste trabalho demonstrou-se como o processo de escolha da rota de distribuição das mercadorias pode ser automatizado. Em conjunto com a ferramenta de controle de cargas já existente no ERP SIGER, foi proposta uma associação com o paradigma meta-heurístico algoritmo genético. Esta associação serve para que, através de uma população inicial de rotas candidatas, se chegue a uma ou mais rotas que minimizariam a distância percorrida pelo veículo de entrega e, consequentemente, desonerariam os custos do processo de transporte.

Para que as chances de sucesso do algoritmo genético sejam maximizadas, é necessário que haja precisão nos parâmetros que são passados para a função de aptidão deste, que é a responsável pelo julgamento da eficiência de cada rota candidata.

Antes do surgimento das APIs de georreferenciamento disponibilizadas pelo Google em 2009, a utilização prática de meta-heurística para solucionar problemas de otimização de rotas de distribuição era prejudicada pela falta de precisão no cálculo real da distância entre dois pontos na superfície terrestre considerando a malha rodoviária.

A utilização de métodos exatos (que analisam todas as possibilidades possíveis antes de indicar a melhor solução) para resolver problemas de otimização combinatória se sobressai a qualquer outra técnica de resolução desde que o espaço de busca seja pequeno. Conforme o espaço de busca aumenta, a eficiência deste frente às técnicas como as de heurística, por exemplo, diminui.

No estudo de caso que foi analisado, a aplicação de um método exato para encontrar a rota mínima de distribuição seria a mais eficiente pelo espaço de busca do estudo possuir somente 6 endereços. Um problema como este, contendo 6 endereços de entrega, possui 720 possibilidades de rotas diferentes, no entanto, por exemplo, se forem acrescidos mais 4 endereços, as possibilidades irão a 3.628.800. Acrescentando mais um endereço, as possibilidades chegam a 39.916.800.

Conforme os dados supracitados, para o estudo de caso que foi analisado, o método exato seria o ideal, no entanto, como já mencionado nos capítulos anteriores, um problema de

ordem real demandaria a localização de uma rota mínima contendo uma quantidade de locais de entrega maior do que a quantidade que foi estudada, o que forçaria a utilização de outros métodos mais eficientes para estes casos.

A utilização de métodos exatos para espaços de buscas muito grandes só seria viável se o algoritmo possuísse a sua disposição uma capacidade de processamento muito grande, como é o caso da funcionalidade de cálculo de rotas disponibilizada pelo Google, que consegue um bom resultado utilizando-se da imensa capacidade de processamento de seus servidores.

Como para os clientes do ERP a capacidade de processamento é limitada, a utilização da heurística dos algoritmos genéticos se mostrou indispensáveis. Mesmo que a rota apresentada pelo algoritmo genético não seja a mesma que seria localizada através de métodos exatos, esta se mostrou muito próxima da solução ótima e sua utilização já traria uma vantagem competitiva, diminuindo o custo do transporte.

O ideal para a aplicação da proposta no ERP SIGER seria uma solução hibrida que testasse o tamanho do espaço de busca antes de iniciar o processamento. Desta forma, dependendo do tamanho do espaço de busca, se optaria pelo método exato ou pela utilização de heurística.

Para que fosse viável a aplicação da proposta no ERP SIGER, seria necessário que a versão da *Google Directions API* utilizada na função de aptidão para localizar a distância entre os locais de entrega fosse a versão *Premier*. Isto por que durante o desenvolvimento do protótipo e da execução do estudo de caso, constatou-se que a quantidade de 2.000 requisições diárias da versão não *Premier* se mostrou um fator limitador do estudo e, consequentemente, será também um limitador durante a utilização oficial pelos clientes do ERP SIGER.

Com a versão *Premier* seria possível utilizar mais vezes a funcionalidade e também aumentar a quantidade de gerações informada na configuração da amplitude do problema proposto. Mesmo não se tendo plena certeza de que o aumento da quantidade de gerações irá impactar na melhora da solução encontrada, sabe-se que a teoria geral dos algoritmos genéticos indica uma inclinação para uma melhora da solução conforme as sucessíveis gerações vão sendo processadas.

Um dos possíveis problemas encontrados na utilização da *Google Directions API* diz respeito a precissão da distância localizada. Como este serviço utiliza dados do *Google Maps*,

sua credibilidade é abalada, já que muitas vezes ao se colocar um endereço na página do *Google Maps*, o ponto exibido no mapa não condiz com o local que se espera. No entanto essa distorção, quando ocorre, é pequena e não é possível determinar se o problema é no processamento do endereço ou se apenas trata-se de um erro de exibição do local no mapa.

A geração da rota mínima de distribuição utilizando o protótipo não é instantânea e demanda um tempo considerável para a obtenção do resultado. Isto pode ser prejudicial já que o objetivo almejado no futuro é a integração com o ERP SIGER e sabe-se que qualquer demora nas rotinas deste implica em reclamações e cobranças por resultados mais rápidos pelos clientes.

Como trabalhos futuros pode-se aprimorar o protótipo, implementando-se tanto um método exato quanto o algoritmo genético. Pode-se, desta forma, estudar o comportamento destes dois métodos a fim de se delinear quando o método exato começa a ficar menos vantajoso do que a utilização da heurística dos algoritmos genéticos.

Pode-se também fazer uma pesquisa sobre outros paradigmas de obtenção da distância entre dois locais no globo e compara-lo com os já apresentados neste trabalho. Da mesma forma, pode-se realizar um comparativo entre as diferentes técnicas de recombinação das soluções candidatas para que se possa indicar se existe alguma técnica que se sobressaia sobre as demais na resolução de problemas de roteamento. Igualmente, pode-se aplicar diferentes técnicas de mutação a fim de se realizar um estudo comparativo.

Uma implementação que poderia ser realizada em um trabalho futuro diz respeito a quando o protótipo interrompe as sucessivas gerações. Atualmente se configura a quantidade de gerações que a simulação irá possuir, no entanto, outra forma que poderia ser aplicada seria indicar um percentual de eficiência e o protótipo só seria interrompido quando a solução encontrada atingisse esse percentual, comparando-se com as soluções candidatas da população inicial.

O casamento entre a robustez do controle de cargas encontrado atualmente no ERP SIGER, a eficácia dos algoritmos genéticos em encontrar soluções para problemas de otimização, e a precisão disponibilizada pela *Google Directions API*, deve proporcionar um serviço de entrega mais preciso aos clientes do ERP SIGER. Esta precisão será revertida em menores custos de transporte e, consequentemente, em maior lucro para os clientes do ERP SIGER.

REFERÊNCIAS BIBLIOGRÁFICAS

AFONSO, R. Computer World. **FGV: Totvs domina pequenas empresas e SAP tem 50% das grandes.** 2011. Disponível em: < http://computerworld.uol.com.br/negocios/2011/04/19/fgv-totvs-domina-pequenas-empresas-e-sap-tem-50-das-grandes/>. Acesso em: 04 mar. 2012.

ARTERO, A. O. **Inteligência Artificial – Teoria e Prática**. São Paulo: Editora Livraria da Física, 2009. 230p.

BARBOZA, E, L. **Gestão estratégica da informação e a inteligência competitiva: Estudo exploratório para a implantação no conselho de consumidores da ENERSUL – CONCEN**. 2008, 71 f. Trabalho de Conclusão de Curso (Graduação em Biblioteconomia) - Curso de Biblioteconomia, Instituto de Ensino Superior da FUNLEC – IESF , Campo Grande, 2008. Disponível em <www.bibliotecariosms.files.wordpress.com/2009/07/elder-lopes-barboza-gestao-estrategica-da-informacao-e-a-inteligencia-competitiva.pdf>. Acesso em: 08 mar. 2012.

BEAL, A. **Gestão estratégica da informação: como transformar a informação e a tecnologia da informação em fatores de crescimento e de alto desempenho nas organizações**. São Paulo: Atlas, 2004. 137p.

BIOLOGIA, Só. **A teoria sintética da evolução**. Disponível em: <http://www.sobiologia.com.br/conteudos/Evolucao/evolucao19.php>. Acesso em: 14 abr. 2012.

BOAVENTURA NETTO, P. O.; JURKIEWICZ, S. **Grafos: introdução e prática**. São Paulo: Blucher, 2009. 162p.

CASTRO, L. N. **Computação Natural – Uma jornada Ilustrada**. São Paulo: Editora Livraria da Física, 2010. 266p.

CIA de Talentos. **Programa Consultores Trainees SAP 2012**. São Paulo: São Paulo. Disponível em: <http://www.ciadetalentos.com.br/sap/>. Acesso em: 05 mar. 2012.

CORRÊA, H.; GIANESI, I. G. N.; CAON, M. **Planejamento, programação e controle da produção: MRP II/ERP- conceitos, uso e implantação**. 4 ed. São Paulo: Atlas, 2006. 452p.

CUNHA, C. B. **Aspectos práticos da aplicação de modelos de roteirização de veículos a problemas reais**. Revista Transportes da ANPET – Associação Nacional de Pesquisa e Ensino em Transportes, v. 8, n. 2, p. 51-74, nov. 2000.

DEITEL, H. M; DEITEL, P. J. **Java: Como Programar**. 6 ed. São Paulo: Pearson Prentice Hall, 2005. 1110p.

DOLCE, O; POMPEO, J. N. **Fundamentos de matemática elementar: geometria plana.** 7 ed. São Paulo: Atual, 1993. 451p.

ENCYDIA - **Fórmula de Haversine**. In: ENCYCLOPEDIA ENCYDIA. Disponível em: < http://pt.encydia.com/es/F%C3%B3rmula_do_Haversine >. Acesso em: 05 abr. 2012.

FILITTO, D. **Algoritmos Genéticos: Uma visão explanatória**. Revista Multidisciplinar da UNIESP Saber Acadêmico, n. 6, p. 136-146, dez. 2008.

FITZ, P. R. **Geoprocessamento sem complicação**. São Paulo: Oficina de Textos, 2008. 160p.

FRIEDMANN, R. M. P. **Fundamentos de orientação, cartografia e navegação terrestre**. 3 ed. Curitiba: Editora UTFPR, 2009. 365p.

GOLDBARG, M. C.; LUNA, H. P. L. **Otimização combinatória e programação linear: modelos e algoritmos**. 2 ed. Rio de Janeiro: Elsevier, 2005. 518p.

GOOGLE. **The Google Directions API**. Disponível em: < https://developers.google.com/maps/documentation/directions/?hl=pt-BR>. Acesso em: 29 mar. 2012.

HAVERSINE - **Fórmula de Haversine**. In: WIKIPÉDIA: a enciclopédia livre. Disponível em: < http://pt.wikipedia.org/wiki/F%C3%B3rmula_de_Haversine >. Acesso em: 05 abr. 2012.

HERMEL, C. V. **Proposta de otimização de rotas rodoviárias utilizando algoritmos genéticos: Um estudo de caso.** 2008, 67 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Curso de Ciência da Computação, Instituto de Ciências Exatas e Tecnológicas, Centro Universitário Feevale, Novo Hamburgo, 2008. Disponível em < http://ged.feevale.br/bibvirtual/Monografia/MonografiaCristianoHermel.pdf>. Acesso em: 20 mar. 2012.

LEE, L. et al. **Vehicle capacity planning system: A case study on vehicle routing problem with time windows**. IEEE Transactions on Systems, Man and Cybernetics, Part A, v. 33(2), 2003.

LEHMANN, C. H. **Geometria analítica**. 9 ed. São Paulo: Globo, 1998. 457p.

LINDEN, R. **Algoritmos Genéticos - Uma importante ferramenta da Inteligência. Computacional**. 2 ed. Rio de Janeiro: Brasport, 2008. 428p.

MELO, A. C. da S.; FILHO, V. J. M. F. **Sistemas de Roteirização e Programação de Veículos.** Revista Pesquisa Operacional, v. 21, n. 2, p. 223-232, jul. 2001.

MITCHELL, M. **An Introduction to Genetic Algorithms**. Cambridge, Massachusetts: The MIT Press, 2001. 208p.

MORESI, E, A, D. **Delineando o valor do sistema de informação de uma organização**. Revista Ciência da Informação, v. 29, n. 1, p. 14-24, jan./abr. 2000.

PARREIRAS, R. O. **Algoritmos Evolucionários e Técnicas de Tomada de Decisão em Análise Multicritério**. 2006. 165 f. Tese (Doutorado em Engenharia Elétrica) - Escola de Engenharia da UFMG, Universidade Federal de Minas Gerais, Belo Horizonte, 2006.

POLI, R.; LANGDON, W.B.; MCPHEE, N.F. **A Field Guide to Genetic Programming**. San Francisco, California: Editado pelos autores, 2008. 233p.

RECH Informática. **Sistema Integrado Gestão Empresarial Rech (SIGER)**. Rio Grande do Sul: Novo Hamburgo. Disponível em: <http://www.rech.com.br>. Acesso em: 09 mar. 2012.

ROCHA, C, H, B. **Geoprocessamento: tecnologia transdisciplinar**. 2 ed. Juiz de Fora: Ed. Do Autor, 2002. 220p.

RUSSELL, S.; NORVIG, P. **Inteligência Artificial**. 2. ed. Rio de Janeiro: Elsevier, 2004. 1021 p.

SAPa. **A História da SAP**. Disponível em <http://www.sap.com/brazil/about/historico/index.epx>. Acesso em 05 mar. 2012.

SAPb. **Enterprise Resource Planning — ERP**. Disponível em: <http://www.sap.com/brazil/solutions/business-suite/erp/index.epx>. Acesso em: 06 mar. 2012.

SAPc. **Sobre a SAP**. Disponível em: <http://www.sap.com/brazil/about/index.epx>. Acesso em: 05 mar. 2012.

SAPd - **Sistema de Gestão Empresarial**. In: WIKIPÉDIA: a enciclopédia livre. Disponível em: < http://pt.wikipedia.org/wiki/SAP_-_Sistema_de_Gest%C3%A3o_Empresarial>. Acesso em: 05 mar. 2012.

SILVA, G. L. **Uma nova abordagem para o problema de roteirização de veículos com restrições operacionais**. 2010. xi, 81 f. Tese (Doutorado em Transportes) - Universidade de Brasília, Brasília, 2010.

SOUZA, F.; MELHADO, S. **A importância do sistema de informação para a gestão das empresas de projeto**. Disponível em: < www.cesec.ufpr.br/workshop2007/Artigo-17.pdf >. Acesso em: 04 mar. 2012.

SOUZA, C. A.; SACCOL, A. Z. **Sistemas ERP no Brasil: Teoria e Casos**. São Paulo: Atlas, 2003. 368p.

TIMLIN, M.; PULLEYBLANK, W. **Procedence constrained routing and helicopter schedulling: heuristic design**. Canada, 1990.

TOTVSa. **ERP**. Disponível em: <http://www.totvs.com/software/erp>. Acesso em: 06 mar. 2012.

TOTVSb. **Nossa História**. Disponível em: <http://www.totvs.com/sobre-a-totvs/nossa-historia>. Acesso em: 12 mar. 2012.

TOTVSc. **Quem Somos**. Disponível em: <http://www.totvs.com/sobre-a-totvs/quem-somos>. Acesso em: 06 mar. 2012.

TURBAN, E.; RAINER JR, R. K.; POTTER, R. E. **Introdução a Sistemas de Informação: Uma abordagem Gerencial**. 4 ed. Rio de Janeiro: Elsevier, 2007. 364p.

TURBAN, E.; RAINER JR, R. K.; POTTER, R. E. **Administração de Tecnologia da Informação: Teoria e Prática.** Rio de Janeiro: Campus, 2003. 598p.

WHITBY, B. **I. A - Inteligência Artificial - Um guia para iniciantes**. São Paulo: Madras, 2003. 160p.

WU, L. **O problema de distribuição periódica de veículos.** São Paulo, 2007. 109 p. Dissertação (Mestrado em Engenharia de Transportes) - Escola Politécnica da Universidade de São Paulo, São Paulo, 2007.