UM ALGORITMO EVOLUCIONÁRIO COMBINADO COM OTIMIZAÇÃO PARA RESOLVER O PROBLEMA DE PROJETO DE UMA SUPPLY CHAIN DE REMANUFATURA

Ernesto D.R. Santibanez-Gonzalez
Departamento de Computação, Universidade Federal de Ouro Preto, Campus Morro do Cruzeiro, Ouro Preto, MG, Brasil
santibanez.ernesto@gmail.com

Geraldo Robson Mateus
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brasil
mateus@dcc.ufmg.br

Henrique L. Pacca Luna
Instituto de Computação, Universidade Federal de Alagoas
Maceió, AL, Brasil
hpluna@gmail.com

RESUMO

O projeto de uma rede logística reversa é um problema complexo e ainda relativamente inexplorado. Nós propomos um esquema de otimização binário baseado em enxame de partículas (BPSO) para resolver um problema de projeto de rede de remanufatura NP-hard. O algoritmo combina uma pesquisa estocástica tradicional com um método de solução ótimo para resolver um problema de Programação Linear (LP) associado. Dividimos o enxame em dois grupos elementares: o primeiro grupo orienta a busca da melhor localização das instalações de remanufatura, enquanto o segundo grupo define os fluxos ótimos entre as instalações. O algoritmo foi codificado em GAMS e apresentamos resultados computacionais para 10 instâncias de rede geradas aleatoriamente com até 350 instalações de coleta, 100 locais candidatos à localização de instalações de reprocessamento e 40 instalações de remanufatura.

PALAVARAS CHAVE. *Supply chain* reversa, Programação inteira, Metaheuristicas.

ABSTRACT

As recognized by several authors, the design of a reverse logistics network is a complex problem and still relatively unexplored. We propose a binary particle swarm optimization (BPSO)-based scheme for solving a NP-hard remanufacturing network design problem. The algorithm combines a traditional stochastic search with an optimal solution method for solving to optimality a relaxed Linear Programming (LP) problem. We divide the swarm into two elementary groups. The first group guides the search for the best location of remanufacturing facilities, while the second group defines the optimal flows between the facilities. We solve to optimality a relaxed LP problem obtained from the original problem and then we project the solution into the swarm space. The algorithm was coded in GAMS and we report computational results for 10 network instances generated randomly with up to 350 sourcing facilities, 100 candidate sites for locating reprocessing facilities and 40 remanufacturing facilities.

KEYWORDS. Reverse supply chain. Integer programming. Metaheuristics.

1. Introdução

A gestão dos fluxos de retorno de produtos tem recebido atenção crescente na última década. A administração eficiente dos resíduos e fluxos de retorno de produtos é uma disciplina preocupada com o destino final dos produtos e seus componentes, e qual é seu impacto sobre a poluição do ar, água e terra, além dos custos de tratamento e eliminação em aterro. Existem regras específicas em algumas regiões como a União Europea, onde o *The European Waste Electrical and Electronic Equipment Directive* (WEEE) e *End of Life Vehicles Directive* (VFV) estão incentivando empresas na indústria automotiva para colaborar com outras empresas e organizações na cadeia de suprimentos para garantir que os produtos, no final de sua vida, possam ser desmontados e reutilizados, recondicionados, reciclados ou eliminados de forma segura.

Além de fortes restrições ambientais e legais, existem várias razões para que um número crescente de empresas estejam interessadas em se engajar em iniciativas sustentáveis, como a gestão dos fluxos reversos dentro de sua cadeia de suprimentos (Rogers & Tibben-Lembke, 1998; Fleischmann et al, 2000). Como apontado por alguns autores (Blackburn et al., 2004), grandes benefícios econômicos "podem ser obtidos mediante o redesenho da cadeia de suprimentos reversa de modo a que esta seja mais rápida e reduza os custosos atrasos ". Segundo os mesmos autores, as estratégias de projeto de cadeias de suprimentos reversas são relativamente inexploradas e subdesenvolvidas. Os produtos devolvidos são remanufaturados, se é conveniente desde o ponto de vista custo-benefício. Algumas empresas podem tratar todo o produto que retorna como defeituoso. Alguns produtos retornados podem ser novos e nunca usados, em seguida, estes produtos devem ser devolvidos para o fluxo direto. Também pode acontecer que alguns produtos não reutilizados ou recondicionados são vendidos como sucata ou para a reciclagem. Algumas vezes, os produtos remanufaturados são vendidos em mercados secundários para ter uma receita adicional, muitas vezes, a um segmento de marketing relutantes ou incapazes de adquirir um novo produto. Neste contexto, as atividades de remanufatura são reconhecidas como a principal opção de recuperação em termos de sua viabilidade e os benefícios que ela traz. Estudamos este problema como parte dos problemas de logística reversa. Neste trabalho é abordado o problema de projetar uma rede de remanufatura de cadeia de suprimentos e propõe-se um algoritmo de otimização baseado em enxame de partículas binária para resolvê-lo. O problema abordado é um problema NP-hard de otimização combinatória.

Nas últimas décadas, algoritmos evolucionários (EVO) têm sido amplamente utilizados como técnicas robustas para resolver uma série de problemas difíceis de otimização combinatória (CO). Um EVO é dirigido pela evolução de uma população de indivíduos na busca de uma solução ótima para os problemas de CO. *Particle Swarm Optimization* (PSO) é um algoritmo evolutivo que tem sido aplicado com sucesso em muitas áreas, e parece ser uma abordagem adequada para vários problemas de otimização. Como apontado por alguns autores, embora esta técnica seja usada exitosamente para resolver muitos problemas contínuos, em problemas discretos ou binários ainda existem algumas dificuldades (Engelbrecht, 2005). Nosso trabalho também aborda este aspecto.

Este trabalho faz duas contribuições principalmente. Em primeiro lugar, propõe e explora um algoritmo baseado em PSO binário, que combina um método de otimização dentro de um esquema de algoritmo PSO para resolver um problema de cadeia de suprimentos reversa. Em segundo lugar, é realizado um estudo computacional do método para exemplos de problemas de até 350 instalações de coleta, 100 sítios candidatos para localizar instalações de reprocessamento e 40 instalações de remanufatura (350x100x40). Apresentamos-nos conclusões respeito à qualidade da solução obtida e os tempos de computação. Para este tipo de problemas, estas são as maiores instâncias resolvidas até agora com algoritmos evolucionários.

No que resta, o artigo está organizado como segue. Na segunda seção, nós fornecemos uma revisão da literatura com uma breve introdução à cadeia de suprimentos sustentável e reversa e particularmente para o caso de remanufatura em conexão com a logística reversa. Na terceira seção, formulamos o modelo matemático para o problema. Na seção 4 é proposto o algoritmo que combina otimização com PSO binário para resolvê-lo. Na quinta seção,

apresentamos os resultados experimentais para redes geradas aleatoriamente. A última seção contém as nossas conclusões.

2. Revisão da Literatura

O aumento da preocupação ambiental e sustentável implica uma série de mudanças para as empresas desde o ponto de vista da estratégia e as atividades tácticas-operacionais, afetando suas pessoas e impactando os seus processos de negócios e tecnologias associadas. Neste âmbito, os modelos para resolver o problema de localização de instalações estão entre as decisões estratégicas importantes para as empresas e organizações que enfrentam questões sustentáveis. *Supply Chain Management* - SCM, e em particular Gestão Sustentável da Cadeia de Suprimentos - SSCM dá um bom arcabouço para abordar as questões sustentáveis. SSCM envolve (a) muitas organizações, (b) muitos processos de negócios entre e dentro destas organizações, e (c) com os objetivos sociais, ambientais e econômicos compartilhados por cada organização e da cadeia de suprimentos. Logística reversa é parte da SSCM e compreende uma série de atividades para o tratamento de produtos devolvidos, até que estes sejam devidamente valorizados ou simplesmente eliminados (Rogers & Tibben-Lembke, 1998). Essas atividades incluem coleta, limpeza, desmontagem, teste e classificação, armazenamento, transporte, e operações de recuperação. Em relação às operações de recuperação, podemos encontrar uma combinação de várias opções de recuperação, como a reutilização, reparação, recondicionamento, remanufatura e reciclagem (Dekker & Van Der Laan, 1999).

Existe outro tipo importante de problemas de projeto de redes logística, o caso particular de cadeias de suprimentos de circuito fechado ou *closed-loop supply chain*. Normalmente, o problema de projeto de rede logística é do tipo de cadeias de suprimentos *forward* ou cadeias de suprimentos reversa, como no caso mencionado anteriormente. Quando integramos as cadeias de suprimentos *forward* e reversa, obtemos um sistema fechado (*closed-loop*) e, em seguida, uma nova classe de problemas englobados pelo termo gestão (e projeto) da cadeia de suprimentos de circuito fechado.

Neste artigo, vamos nos concentrar no problema de rede de remanufatura e na cadeia de suprimentos reversa. Aqui, a remanufatura é definida como um dos métodos de recuperação pelo qual produtos (usados) ou componentes são recuperados para produzir uma unidade equivalente em qualidade e desempenho que o produto original novo e que, em seguida, podem ser revendidos como novos produtos ou peças. Como as atividades de remanufatura são freqüentemente implementadas pelo produtor inicial dessa rede é provável que seja um sistema de circuito fechado. Atividades de remanufatura são reconhecidas como a principal opção de recuperação em termos de sua viabilidade e os benefícios que traz para a empresa. Essa atividade fornece ás empresas uma maneira de controlar a eliminação dos seus produtos usados, para reduzir eficazmente os custos de produção e poupar matérias-primas.

2.1 Modelos de Localização para Projeto de *Supply Chain* Reversa e de Circuito Fechado

A localização de instalações é um dos problemas estratégicos que fazem parte de um processo de planejamento para a gestão e projeto da rede da cadeia de suprimentos.O problema de localização de instalações e alocação de clientes não é novidade para a comunidade de pesquisa operacional e aborda os aspectos fundamentais do projeto da cadeia de suprimentos (Daskin et al., 2005). Este problema é "um dos mais abrangentes problemas de decisões estratégicas que precisam ser otimizados para a operação de longo prazo eficiente de toda a cadeia" (Altiparmak et al., 2006). Como observado por Farahani e Hekmatfar (2009), algumas pequenas mudanças sobre os modelos clássicos de localização de instalações pode transformar esses problemas em problemas bastante difíceis de resolver.

Nos últimos anos, a modelagem matemática e os métodos de solução para a gestão eficaz dos fluxos de retorno (e / ou integrados com os fluxos *forward*) têm sido estudados no contexto da logística reversa, em cadeia de suprimentos de circuito fechado e cadeia sustentável de suprimentos.

Dekker et al. (2004) classificou a pesquisa sobre a logística reversa (e *supply chain* de circuito fechado) em três áreas funcionais: Inventário, Distribuição e Produção e, Âmbito da

Cadeia de Suprimentos. Decisões de distribuição tradicionais envolvem o projeto de rede e localização de instalações para a distribuição de produtos *forward* e reverso e para a coleta e reprocessamento de produtos devolvidos. Este trabalho se concentra em modelos quantitativos para o projeto de cadeias de suprimentos de circuito fechado, e diz respeito às decisões sobre a estrutura topológica da rede, o número de instalações para localizar, as capacidades das instalações, e a distribuição dos fluxos de produtos entre as instalações localizadas em cada camada da cadeia de suprimentos.

Para uma revisão da pesquisa sobre modelos quantitativos usados em logística reversa e cadeias de suprimentos de circuito fechado antes de 2000, pode ser consultado Dekker et al (2004). Rubio et al (2008) fez uma compilação de pesquisas publicadas sobre a logística reversa dentro do período de 1995-2005. Eles estudaram o tema com base na classificação proposta por Dekker et al. (2004).

Como argumentado por vários autores, as abordagens tradicionais para a solução de problemas de projeto de rede de cadeias de suprimentos de circuito fechado (ou reverso) normalmente é formular um problema de programação linear misto inteiro de grande porte (MILP) onde a localização potencial de instalações são modeladas como variáveis binárias e os fluxos de produtos entre instalações como variáveis (continuas) não-negativas. Tipicamente, estes problemas pertencem à classe de problemas NP-hard de otimização combinatória. A integração entre os fluxos *forward* e reverso em cadeias de suprimentos de circuito fechado introduz algumas modificações nos modelos tradicionais de localização de instalações, dando origem a algumas complexidades adicionais na solução destes modelos.

Por ordem cronológica, os modelos de logística reversa são discutidos recentemente, por exemplo, por Jayaraman et al. (2003), Yongsheng e Shouyang (2008), Salema et al.(2010) e Gomes et al. (2011). Quase todos os autores propuseram modelos MILP. As maiorias dos métodos de solução estão baseadas em pacotes (padrão) comerciais.

Modelos para o projeto de cadeias de suprimentos de circuito fechado são tratados nos trabalhos de Lu e Bostel (2007), Barbosa-Póvoa et al. (2007), Mutha e Pokharel (2009), Zarei et al (2010) e Amin e Zhang (2012). Modelos estocásticos, em combinação com funções multiobjetivo foram estudados por Pishvaee e Torabi (2010).

Quanto aos algoritmos evolucionários para resolver problemas relacionados têm sido estudados, por exemplo, os seguintes: *Simulated Annealing* (Lee e Dong, 2008), Algoritmo Genético (Min et al, 2006; Lee et al, 2009; Zarei et al, 2010.).; Algoritmo Memético (Pishvaee et al., 2010) e *Tabu Search* (Lee e Dong, 2008). Note-se que Lee et al., (2009) resolveram problemas de até 15 centros de retorno, 10 centros de desmontagem e 14 centros de processamento. Eles não relataram resultados computacionais em relação a tempo ou a qualidade do *gap* ótimo obtido. Lee e Dong (2008) resolveram problemas em redes de até 100x40x30 com um tempo de computação (s) máximo de 2.939 e um *gap* máximo (em comparação com um limite inferior) de 15%.

2.2 *Discrete Particle Swarm Optimization Algorithm*

PSO é uma metaheurística com base no comportamento social e comunicação de um bando de pássaros ou cardumes de peixes (Kennedy & Eberhart, 1995). *PSO* pode ser considerado como um algoritmo evolutivo, pois explora o espaço de soluções através de uma população (enxame) e da estrutura de vizinhança de cada individuo (partículas) e aproveita a informação geracional adquirida. Mas, tem algumas divergências com outros algoritmos evolutivos, por exemplo, ele não tem operadores evolutivos tais como cruzamento e mutação dos métodos genéticos. *PSO* tem a vantagem de que é mais simples de utilizar e com menos parâmetros para ajustar. Em *PSO*, as soluções potenciais (contidas em partículas), movimenta-se em um espaço de busca multidimensional com uma velocidade, que é constantemente atualizada por uma combinação entre a experiência própria da partícula, a experiência dos vizinhos da partícula e a experiência do enxame (*swarm*) inteiro. *PSO* têm sido aplicada com sucesso a uma ampla gama de problemas (Poli, 2008). Desde sua origem, *PSO* foi desenvolvida para resolver problemas de otimização contínua (Kennedy & Eberhart, 2001; Poli, 2008). Recentemente,

algumas experiências têm sido realizadas com uma modificação do algoritmo, chamada de *discrete PSO*, aplicadas a solução de problemas de otimização combinatória (por exemplo: Allahverdi & Al-Anzi, 2006; AlHajri et al, 2007.).

3. Modelo matemático para projetar uma rede de cadeia de suprimentos de remanufatura

Nesta seção apresentamos o modelo MILP para o problema de projetar uma rede sustentável de cadeia de suprimentos. Este problema pode ser classificado como um modelo de localização capacitado com demanda conhecida, com um produto único, estático e de três camadas. A rede de cadeia de suprimentos de remanufatura é composta por três tipos de empresas: instalações de coleta (sites de origem, por exemplo, uma loja de varejo), locais de reprocessamento e instalações de remanufatura. Ao nível dos clientes, há demanda de produtos e também produtos usados pronto para ser recuperados, por exemplo, telefones celulares. Supomos que os clientes devolvem produtos a sites de coleta como uma loja de varejo. Na segunda camada da rede de cadeia de suprimentos, existem centros de reprocessamento utilizados apenas no canal reverso e eles são responsáveis por atividades, tais como a limpeza, desmontagem, verificação e classificação dos produtos de retorno, antes que esses produtos (devolvidos) sejam enviados de volta para as instalações de remanufatura. Na terceira camada, as instalações de remanufatura recebem os produtos retornados e verificados provenientes de centros de reprocessamento e são responsáveis pelo processo de recondicionamento. Neste trabalho abordamos o fluxo reverso de produtos (retornados) provenientes de sites de coleta que vão para as instalações de remanufatura através de instalações de reprocessamento adequadamente localizados em locais pré-definidos. Em tal rede, o fluxo reverso é formado por produtos usados, que vão desde os clientes através de pontos de coleta até as instalações de remanufatura, enquanto o outro fluxo (*forward*) desde as instalações de remanufatura diretamente para ponto de vendas é composto por produtos "novos".

3.1 Modelo RSCP

Em nosso modelo é assumido que a demanda de produtos (novos) e as quantidades disponíveis de produtos de retorno nos clientes são conhecidos e deterministas.

Apresentamos os seguintes dados de entrada e conjuntos para nosso modelo:

I = o conjunto de instalações de origem pertencentes a primeira camada, indexado por i
J = o conjunto de instalações de remanufatura pertencentes a terceira camada, indexado por j
K = o conjunto de locais candidatos para localizar instalações de reprocessamento, pertencentes a camada intermédia, indexado por k
a_i = quantidade de produtos de retorno no local de origem $i \in I$
b_j = quantidade de produtos demandados no local remanufatura $j \in J$
f_k = custo fixo de abrir uma instalação de reprocessamento no local candidato $k \in K$
g_k = custo de gestão em uma instalação de reprocessamento aberta no local candidato $k \in K$
c_{ik} = é o custo unitário de entrega de produtos em $k \in K$ a partir de uma instalação de origem localizada em $i \in I$
d_{kj} = é o custo unitário de distribuição desde uma instalação em $k \in K$ para $j \in J$
m_k = capacidade de reprocessamento no local $k \in K$

Nós consideramos as seguintes variáveis de decisão:

w_k = 1 se localizar uma instalação de reprocessamento no local candidato $k \in K$, 0 caso contrário
x_{ik} = fluxo de retorno desde origem $i \in I$ para instalação de reprocessamento localizada em $k \in K$
y_{kj} = fluxo desde a instalação de remanufatura localizada em $k \in K$ para a instalação $j \in J$

Seguindo o modelo proposto por Li (2011), o problema de projeto de uma cadeia de suprimentos de remanufatura (*RSCP*) é definido por:

$$Minimize \sum_{k \in K} f_k w_k + \sum_{i \in I} \sum_{k \in K} c_{ik} x_{ik} + \sum_{k \in K} \sum_{i \in I} g_k x_{ik} + \sum_{k \in K} \sum_{j \in J} d_{kj} y_{kj} \quad (1)$$

$$\sum_{i \in I} x_{ik} \leq m_k w_k \qquad \forall k \in K \quad (2)$$

$$\sum_{k \in K} x_{ik} \leq a_i \qquad \forall\, i \in I \qquad (3)$$

$$\sum_{k \in K} y_{kj} \geq b_j \qquad \forall\, j \in J \qquad (4)$$

$$\sum_{i \in I} x_{ik} = \sum_{j \in J} y_{kj} \qquad \forall\, k \in K \qquad (5)$$

$$x_{ik}, y_{kj} \geq 0, \qquad \forall\, i \in I, \forall\, j \in J, \forall\, k \in K \qquad (6)$$

$$w_k \in \{0,1\} \qquad \forall\, k \in K \qquad (7)$$

A função objetivo (1) minimiza a soma dos custos de localização de instalações de reprocessamento, mais os custos de transporte desde as instalações de coleta para às instalações de reprocessamento e destas para às instalações de remanufatura. Restrição (2) garante que o fornecimento para às instalações $k \in K$ é realizado para uma instalação de reprocessamento já aberta e com capacidade para atender os produtos que recebe. Restrição (3) garante que todos os produtos de retorno desde a instalação $i \in I$ para todas às facilidades $k \in K$ é menor que a capacidade de oferta. Restrição (4) garante que a demanda na instalação de remanufatura $j \in J$ é satisfeita. Restrição (5) garante que todos os produtos de retorno que chegam a facilidade k também são entregues às instalações de remanufatura. Restrição (6) é uma restrição padrão de variáveis não-negativas. Restrição (7) é uma restrição padrão de variáveis binárias. Este modelo tem $O(n^2)$ variáveis não-negativas, onde $n = max\ \{|I|,|J|,|K|\}$ e $|K|$ variáveis binárias. O número de restrições é $O(n)$.

4. Esquema Evolucionário

Nosso algoritmo de solução do problema de projeto de rede de cadeia de suprimentos de remanufatura considera a integração de algoritmos evolutivos com algoritmos de otimização, aproveitando as características particulares do problema. Neste artigo usamos a variante discreta do algoritmo *PSO* como método de computação evolutiva para resolver problemas de otimização contínua. O *PSO* é um algoritmo de otimização baseado na teoria de enxame onde a ideia principal de um *PSO* clássico é modelar o bando de pássaros voando em torno de um pico em uma paisagem. Em *PSO* as aves são substituídas por seres artificiais chamados partículas e o pico na paisagem é o pico (máximo ou mínimo) de uma função objetivo (*fitness*). As partículas do enxame estão voando através do espaço de solução de busca com uma velocidade para a formação de (sub)bandos de forma a atingir os picos das funções de aptidão (seja minimizar ou maximizar uma função). Em um *PSO* contínuo, o estado de uma partícula individual i em um espaço de busca da solução D é caracterizado por dois fatores: a sua posição u e velocidade v. A posição u e velocidade v da i-partícula no espaço de solução d-dimensional de busca pode ser representado como segue:

$$v_{id}^{t+1} = c_1 v_{id}^{t} + c_2 r_1 (p_{id} - u_{id}^{t}) + c_3 r_2 (p_{ig} - u_{id}^{t}) \qquad (8)$$

$$u_{id}^{t+1} = u_{id}^{t} + v_{id}^{t+1} \qquad (9)$$

Onde c_1 é chamado de fator de ponderação de inércia, c_2 e c_3 são constantes chamadas coeficientes de aceleração, r_1 e r_2 são dois números independentes aleatórios uniformemente distribuídos no intervalo [0,1], p_{id} correspondente ao melhor valor objetivo atingido pela partícula i até o tempo t, p_{ig} representa a melhor partícula encontrada até agora no tempo t. Equação (8) é usada para o cálculo da nova velocidade de cada partícula i no tempo (t+1). Equação (9) é usada para a atualização da posição da partícula i no tempo (t+1). Cada $v_{id}^{t} \in [-v_{max}, v_{max}]$ e $u_{id}^{t} \in [-x_{max}, x_{max}]$, com v_{max} e x_{max} são definidas pelo usuário para controlar o excessivo número de partículas fora do espaço de solução. Partículas vão voar para uma nova posição de acordo com (9). O processo é repetido até que um critério de parada seja atingido.

4.1 *Binary PSO Algorithm*

A fim de resolver os problemas de otimização discreta, Kennedy e Eberhart (1997) proposeram um algoritmo *PSO* binário (*BPSO*). A diferença entre um algoritmo *BPSO* e algoritmo *PSO* tradicional é que a equação (9) foi substituída pela seguinte expressão:

$$u_{id}^{t+1} = \begin{cases} 1 \; if \; rand(\;\;) < S(v_{id}^{t+1}) \\ 0 \; if \; rand(\;\;) > S(v_{id}^{t+1}) \end{cases} \qquad (10)$$

Onde S(.) é a função sigmoide e *rand*(.) é um número aleatório uniformemente distribuído no intervalo [0,1]. Usamos este método neste artigo.

O algoritmo *PSO* típico codifica uma solução (viável) para o modelo *RSCP* como uma partícula, e, em seguida, aplica-se o algoritmo tradicional para resolver o problema. Seguimos uma abordagem diferente em nosso trabalho, aproveitando a estrutura do modelo *RSCP*.

Podemos definir valores 0 ou 1 para as variáveis $w_k, \forall\, k \in K$, vamos chamá-los w_k^*. Então podemos reescrever o modelo *RSCP* como segue:

(RSCPR)

$$Minimize \sum_{k \in K^*} f_k w_k^* + \sum_{i \in I} \sum_{k \in K} c_{ik} x_{ik} + \sum_{k \in K} \sum_{i \in I} g_k x_{ik} + \sum_{k \in K} \sum_{j \in J} d_{kj} y_{kj} \qquad (1a)$$

Sujeito a:

$$\sum_{i \in I} x_{ik} \leq u_k w_k^* \qquad \forall\, k \in K^* \qquad (2a)$$

$(3) - (6)$

Onde o conjunto K^* contém as instalações já abertas, e as variáveis w_k^* são agora coeficientes com valores conhecidos. A restrição (2a) continua a garantir que os produtos de retorno desde todas as instalações de coleta sejam transportados para uma instalação de reprocessamento que já esteja aberta. O restante das restrições permanece igual que o modelo *RSCP* original. Além disso, observe que este problema associado é um problema de programação linear. No nosso esquema evolutivo, usamos um algoritmo *BPSO* para orientar todo o processo de busca por uma solução ótima para o problema *RSCP*. No entanto, uma parte da partícula é obtida por resolução do problema de otimização *RSCPR*. Desta forma, usamos um algoritmo *BPSO* decomposto, com base em dois grupos de enxames, um deles orienta a busca global para uma solução ótima.

4.2 Representação da posição das partículas

Encontrar um bom método de codificação para o problema de otimização é o problema mais crítico na definição do algoritmo *PSO* (Kennedy & Eberhart, 2001). Neste documento, o espaço da partícula *d*-dimensional é dividido em dois conjuntos u_1 e u_2. O comprimento de cada partícula no primeiro conjunto (vector) corresponde ao número de sítios candidatos para localizar instalações de reprocessamento $|K|$. Enquanto a segunda parte, as partículas representam uma solução para o problema LP relaxado associado. No primeiro grupo de partículas, cada componente de cada vetor pode ter apenas 1 ou 0. Se o componente *k-ésimo* de u_1 é igual a 1, então uma instalação de reprocessamento no local candidato *k* deve ser aberta, 0 em caso contrário. O segundo grupo de partículas representa os fluxos de produtos de retorno entre a instalação de coleta dos retornos e as instalações de reprocessamento e entre as instalações de reprocessamento e as instalações de remanufatura. Conforme descrito anteriormente neste trabalho, esta parte é obtida resolvendo o problema de otimização *RSCPR*. Veja a seguir um exemplo de codificação da posição das partículas, para um problema de quatro (4) instalações de coleta, três (3) locais candidatos para localizar instalações de reprocessamento e uma (2) instalação de remanufatura. Para o exemplo da Figura 1, os produtos de retorno somam um total de 500, e o fluxo desses produtos é mostrado na Figura.

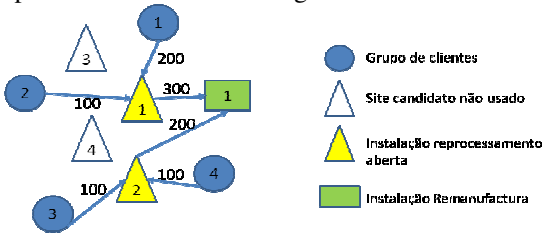

Figura 1: Exemplo de problema *RSCP* com fluxo de produtos de retorno entre instalações

Para esse problema exemplo, a posição para uma partícula deste bando pode ser a seguinte: (100;0;150;0;0;150;0;100;250;0;0;250). Essa posição para um problema com 4 instalações de coleta, 2 instalações de reprocessamento abertas e 2 instalações de remanufatura, indica que 100 unidades de produto de retorno vão desde a primeira instalação de coleta para a primeira instalação de reprocessamento aberta, 150 unidades de produto de retorno são transportadas desde a segunda instalação de coleta para a primeira instalação de reprocessamento aberta, 150 unidades são transportadas desde a terceira instalação de coleta para a segunda instalação de reprocessamento aberta e 100 unidades são transportadas desde a quarta instalação de coleta para a segunda instalação de reprocessamento aberta. Logo, desde a primeira instalação de reprocessamento aberta são transportadas 250 unidades para a primeira instalação de remanufatura e outras 250 unidades são transportadas desde a segunda instalação de reprocessamento para a segunda instalação de remanufatura.

4.3 Geração do bando inicial de partículas

O bando inicial de partículas, com a posição de cada partícula é gerado da seguinte forma:

- Para as partículas do primeiro bando, a posição é gerada em forma aleatória usando uma distribuição uniforme no intervalo [0,1]. Cada posição de partícula é representada por um vetor $|K|$-dimensional de 0-1, sendo $|K|$ o número de locais candidatos para localizar uma instalação de reprocessamento. Depois de gerar esse vetor $|K|$-dimensional, é usado (10) para obter um vetor binário. É claro que este procedimento pode originar soluções inviáveis. Por exemplo, para o problema da Figura 1 anterior, poderíamos ter a seguinte posição de partícula: (1;0,0;0,0;0,0). A posição dessa partícula indica que só uma instalação de reprocessamento deve ser aberta no local 1. Se a capacidade dessa instalação for inferior a 500, essa posição é "inviável". Por esse motivo, nos usamos um procedimento de viabilização similar ao usado para o caso do algoritmo genético, para dar uma posição viável a partícula, segundo explicado mais adiante neste artigo.
- Para as partículas do segundo bando, a posição é gerada a partir da solução do problema associado de PL que considera a posição de cada partícula do primeiro bando.

4.4 O procedimento de viabilidade

O algoritmo BPSO padrão pode gerar algumas partículas do primeiro grupo (enxame) representando soluções inviáveis, o que pode dar origem a posições inviáveis das partículas do segundo bando. Para resolver este problema, para cada partícula do primeiro bando aplicamos um procedimento simples para projetar a posição de cada partícula no espaço de soluções viáveis de nosso problema. Este procedimento consiste em duas etapas. Na primeira etapa vamos verificar se o número de instalações já abertas são suficientes para satisfazer toda a demanda. No segundo passo, para aquelas partículas que não satisfazem a procura, obtemos uma solução viável para RSCP com base na partícula inicial. Este procedimento é bastante simples, começa por abrir instalações até que a demanda seja satisfeita. O enxame é composto de apenas soluções viáveis. A aptidão (*fitness*) de uma partícula é calculada resolvendo simultaneamente o problema RSCPR e usando a função objetivo (1a).

4.2 Outras considerações

Nós inicializamos nosso algoritmo com um tamanho de enxame de 20 partículas. O número máximo de iterações foi também ajustado para 10. Utilizou-se esta configuração de parâmetros porque o teste inicial proporcionou bons resultados para a complexidade do problema estudado neste artigo. O coeficiente de aceleração c_2 é definido com valor 2 e c_3 com valor 1. O coeficiente de inércia começa com um valor de 0,9 e diminui até chegar a 0,4 dependendo do número de iterações realizadas. O enxame inicial é composto de apenas soluções viáveis. Cada partícula do primeiro grupo é gerada aleatoriamente usando uma distribuição uniforme no intervalo [0,1]. Em seguida é aplicado o procedimento binário e depois disso é aplicado o procedimento de viabilidade. Em relação ao segundo grupo de partículas, para cada partícula resolvemos o problema de otimização LP descrito anteriormente neste artigo. O vetor velocidade

v_{id}^t é gerado aleatoriamente no intervalo [-4,4], como no método contínuo. Em cada iteração, o algoritmo utiliza o método descrito anteriormente e actualiza o vector de posição u_{id}^{t+1}de acordo com (9). O vetor velocidade v_{id}^t é atualizado a cada iteração de acordo com (8). A função de *fitness* é como (1). Em cada iteração vamos atualizar a melhor posição de uma partícula e a melhor posição global do enxame. Observe que não exploramos estruturas de vizinhança, ou seja, a forma como a informação é distribuída entre os seus membros.

5. Experimentação computacional

Nesta seção, vamos discutir e comparar os resultados computacionais obtidos pelo algoritmo evolutivo proposto. A nossa proposta é analisar o desempenho do algoritmo proposto em termos de tempo computacional para obter a solução e da qualidade do limite superior (solução) obtido. Implementamos o algoritmo em GAMS e usamos CPLEX como uma sub-rotina (chamado de dentro GAMS) para resolver o problema de otimização *RSCPR*. Todos os experimentos foram desenvolvidos em um PC com 4 GB RAM e 2.3GHz. Na literatura, não há conjuntos de dados disponíveis para o nosso problema. Geramos aleatoriamente 10 problemas testes seguindo metodologias similares utilizadas para conhecidos problemas de cadeias de suprimentos relacionados com o nosso problema (por exemplo: Lu & Bostel, 2007). Estes problemas de teste são dados que correspondem a redes de até 350 locais de origem, 100 locais candidatos à localização de instalações de reprocessamento e 40 instalações de remanufatura. O conjunto de dados para os problemas de teste são apresentados na Tabela 1, e que estão disponíveis com os autores. Todos os custos de transporte foram gerados aleatoriamente usando uma distribuição uniforme com parâmetros [1,40]. Os custos de gestão foram criados em 30 para todas as instâncias de problemas, exceto para o N ° 1. Instâncias 7 e 8 são as mesmas que os casos 4 e 5, respectivamente, exceto que o valor dos custos fixos foram obtidos multiplicando por 10 os custos fixos das instancias 4 e 5. Instâncias 9 e 10 são as mesmas que os casos 7 e 8, respectivamente, mas a capacidade de cada local foi aumentada em 50%. Produtos disponíveis para retorno (a_j), capacidade de instalações de reprocessamento (m_k) e a capacidade das instalações remanufatura (b_l) são mostrados na Tabela 1.

Tabela 1: Conjunto de dados

#	Instâncias Problemas	Locais Origem	Locais Candidatos Reprocessamento	Instalações Remanufatura	Custos Fixos	a_j	m_k	b_l
1	40x20x15	40	20	15	300	150	400	400
2	100x40x20	100	40	20	500	150	400	750
3	150x40x20	150	40	20	1000	200	800	1500
4	200x80x20	200	80	20	1000	300	800	3000
5	300x80x40	300	80	40	2000	200	800	1500
6	350x100x40	350	100	40	2000	200	800	1750
7	200x80x20	200	80	20	10000	300	800	3000
8	300x80x40	300	80	40	20000	200	800	1500
9	200x80x20	200	80	20	10000	300	1200	3000
10	300x80x40	300	80	40	20000	200	1200	1500

Realizamos 5 ensaios para cada instâncias de teste. Tabela 2 e 3 mostram os resultados obtidos utilizando o algoritmo. Para cada instância de problema e ensaio, as tabelas mostram a solução fornecida pela fase inicial do algoritmo aleatório (z_{ran}), a solução fornecida pelo algoritmo BPSO2 (z_p), os tempos computacionais (em segundos) e o gap (100[z^* - z_p]/z^*), obtido comparando o z_p com o valor ótimo inteiro da função objetivo (z^*), obtido por GAMS. Todos os valores foram arredondados para um *gap* de duas casas decimais. A tabela também mostra o valor mínimo, máximo e médio para cada coluna e cada instância de teste. Observe que, para todas as instâncias de teste, a diferença média máxima é de 1,58% (instancia # 9) e *gap* médio mínimo é de 0,04% (instancia # 2). Quanto ao tempo de computação, note que todas as instâncias de teste foram resolvidas em menos de 58 segundos.

5. Conclusões

Neste trabalho, propomos uma heurística de otimização baseado no algoritmo binário de *PSO* para resolver um problema de projeto de rede de uma cadeia de suprimentos reversa. Este é um problema NP-hard e foi formulado como um problema de programação linear inteira mista 0-1. O algoritmo guia a busca por uma solução ótima para o problema *RSCP* combinando uma solução de pesquisa aleatória (*PSO*) com as melhores soluções geradas através da resolução de um problema de otimização associado. O algoritmo gera dois grupos de enxames. Primeiro é gerado as partículas que representam se a instalação for aberta ou fechada. A segunda parte da partícula é obtida a partir da solução de um problema de otimização *RSCPR* associada ao problema *RSCP* original. A heurística proposta foi codificada em GAMS e testado com 10 casos de teste gerados aleatoriamente. Resultados computacionais sobre *gap* e os tempos de computação são promissores.

Tabela 2: Valor solução fase aleatória (z_{ran}), valor solução *BPSO* (z_P), tempo computacional (secs) e *gap (%)*

# Instância	*Tentativa*	z_{ran}	z_P	*secs*	*gap(%)*
1	1	148500	146900	13,40	0,00
	2	149400	148550	13,46	1,12
	3	149200	149000	13,87	1,43
	4	149400	147650	13,51	0,51
	5	150050	147900	13,71	0,68
Mínimo		148500,0	146900,0	13,40	0,00
Máximo		150050,0	149000,0	13,87	1,43
Média		149310,0	148000,0	13,59	0,75
2	1	521200	520250	15,92	0,03
	2	521350	520300	15,98	0,04
	3	520300	520300	15,73	0,04
	4	520300	520300	15,79	0,04
	5	520600	520300	15,96	0,04
Mínimo		520300,0	520250,0	15,73	0,03
Máximo		521350,0	520300,0	15,98	0,04
Média		520750,0	520290,0	15,88	0,04
3	1	1065400	1065200	20,10	0,23
	2	1066600	1065400	19,83	0,24
	3	1065400	1065400	19,01	0,24
	4	1066600	1065800	19,88	0,28
	5	1065800	1065400	19,93	0,24
Mínimo		1065400,0	1065200,0	19,01	0,23
Máximo		1066600,0	1065800,0	20,10	0,28
Média		1065960,0	1065440,0	19,75	0,25
4	1	2091400	2090100	34,85	0,36
	2	2090700	2089000	32,89	0,30
	3	2090500	2090500	32,40	0,37
	4	2090500	2089100	33,38	0,31
	5	2090600	2090300	33,72	0,36
Mínimo		2090500,0	2089000,0	32,40	0,30
Máximo		2091400,0	2090500,0	34,85	0,37
Média		2090740,0	2089800,0	33,45	0,34
5	1	2130800	2130800	56,17	0,38
	2	2132100	2130800	56,15	0,38
	3	2131400	2130800	58,32	0,38
	4	2132200	2131100	56,29	0,40
	5	2132400	2131100	54,80	0,40
Mínimo		2130800,0	2130800,0	54,80	0,38
Máximo		2132400,0	2131100,0	58,32	0,40
Média		2131780,0	2130920,0	56,35	0,39
6	1	2471200	2467350	54,26	0,35
	2	2470900	2468750	54,32	0,41
	3	2471900	2469550	53,82	0,44
	4	2471850	2468500	54,34	0,40
	5	2468750	2467850	54,70	0,37
Mínimo		2468750,0	2467350,0	53,82	0,35
Máximo		2471900,0	2469550,0	54,70	0,44
Média		2470920,0	2468400,0	54,29	0,40

Tabela 3: Valor solução fase aleatória (z_{ran}), valor solução *BPSO* (z_P), tempo computacional (secs) e *gap(%),instancias 7-10*

# Instância	*Tentativa*	z_{ran}	z_P	*secs*	*gap(%)*
7	1	2765500	2765500	40,01	0,28
	2	2768700	2765600	36,20	0,29
	3	2766700	2765600	35,81	0,29
	4	2765600	2764800	35,75	0,26
	5	2765500	2764100	35,78	0,23
Mínimo		2765500,0	2764100,0	35,75	0,23
Máximo		2768700,0	2765600,0	40,01	0,29
Média		2766400,0	2765120,0	36,71	0,27
8	1	3481700	3480800	55,85	0,23
	2	3483000	3481400	54,33	0,25
	3	3481300	3480700	54,68	0,23
	4	3482600	3481400	55,05	0,25
	5	3482900	3480200	56,52	0,22
Mínimo		3481300,0	3480200,0	54,33	0,22
Máximo		3483000,0	3481400,0	56,52	0,25
Média		3482300,0	3480900,0	55,28	0,24
9	1	2727700	2521400	35,46	1,69
	2	2525300	2513600	36,06	1,38
	3	2531000	2517200	36,55	1,52
	4	2533100	2519900	36,48	1,63
	5	2535500	2521100	36,79	1,68
Mínimo		2525300,0	2513600,0	35,46	1,38
Máximo		2727700,0	2521400,0	36,79	1,69
Média		2570520,0	2518640,0	36,27	1,58
10	1	2996600	2992000	38,97	1,09
	2	2999900	2995600	46,73	1,21
	3	2994500	2990900	48,01	1,05
	4	3003400	2999600	47,72	1,35
	5	3000000	2994000	46,95	1,16
Mínimo		2994500,0	2990900,0	38,97	1,05
Máximo		3003400,0	2999600,0	48,01	1,35
Média		2998880,0	2994420,0	45,68	1,17

Referências

AlHajri, M.F., AlRashidi, M.R., e El-Hawary, M.E. (2007), A Novel Discrete Particle Swarm Optimization Algorithm for Optimal Capacitor Placement and Sizing, *Proceedings Canadian Conference on Electrical and Computer Engineering - CCECE 2007*, 1, 286 – 1289.

Allahverdi, A., e Al-Anzi, F.S. (2006), A PSO and a Tabu search heuristics for the assembly scheduling problem of the two-stage distributed database application, *Computers & Operations Research*, 33, 1056–10.

Altiparmak, F., Gen, M., Lin, L., e Paksoy, T. (2006), A genetic algorithm approach for multiobjective optimization of supply chain networks, *Computers & Industrial Engineering*, 51, 197–216.

Amin, S.H., e Guoqing Zhang, G. (2012), A proposed mathematical model for closed-loop network configuration based on product life cycle, *Int J Adv Manuf Technol*, 58, 791 – 801.

Barbosa-Povoa, A.P., Gomes Salema, M.I., e Novais, A.Q., Design and Planning of Closed Loop Supply Chains, em L. Papageorgiou & M. Georgiadis (Eds.), *Supply Chain Optimization*, Wiley, NY, 187-218, 2007.

Blackburn. J.D., Guide, V.D.G., Souza, C., e Van Wassenhove, L.N. (2000), Reverse Supply Chains for Commercial Returns, *California Management Review*, 46, 6-22.

Daskin, M.S., Snyder, L.V., e Berger, R.T., Facility Location in Supply Chain Design, em A. Langevin and D. Riopel (Eds.), *Logistics Systems: Design and Optimization*, 39-65, Springer, New York, 2005.

Dekker, R., Fleischmann, M., Inderfurth, K., e van Wasenhove, L. (Eds.), *Reverse logistics: quantitative models for close-loop supply chains*, Springer-Verlag, Berlin, 2004.

Dekker, R., e Van Der Laan, E.A. (1999), Gestion des stocks pour la fabrication et la refabrication simultanées: synthèse de resultants récents, *Logistique & Management*, 7, 59–64.

Engelbrecht, A.P., *Fundamentals of Computational Swarm Intelligence*, Wiley, New York, 2005.

Farahani, R.Z. e Hekmatfar, M. (Eds.), *Facility location: concepts, models, algorithms and case studies*, Physica-Verlag HD, Berlin, 2009.

Fleischmann, M., Krikke, H.R., Dekker, R., e Flapper, S.D.P. (2000), A characterisation of logistics networks for product recovery, *Omega*, 28, 6, 653-666.

Geoffrion, A.M., e Graves, G.W. (1974), Multicommodity Distribution System Design by Benders Decomposition, *Managament Science*, 20, 5, 822-844.

Gomes, M.L., Barbosa-Povoa, A.P., e Novais, A.Q. (2011), Modelling a recovery network for WEEE: A case study in Portugal, *Waste Management*, 31, 1645–1660.

Jayaraman, V., Patterson, R.A., e Rolland, E. (2003), The design of reverse distribution networks: Models and solution procedures, *European Journal of Operational Research*, 150, 128–149.

Kennedy, J., e Eberhart, R. C. (1995), Particle Swarm Optimization, *Proceedings of the IEEE International Conference on Neural Networks*, 1942-1948.

Kennedy, J., e Eberhart, R. C. (1997), A discrete binary version of the particle swarm algorithm, *Proceedings of the IEEE International Conference on Computational Cybernetics and Simulation*, 4104 – 4108.

Kennedy, J., e Eberhart, R.C., Swarm Intelligence, Morgan Kaufmann Publishers. San Francisco, 2001.

Lee, D.-H., e Dong, M. (2008), A heuristic approach to logistics network design for end-of-lease computer products recovery, *Transportation Research Part E*, 44, 455-474.

Lee, J-E., Gen, M., e Rhee, K-G. (2009), Network model and optimization of reverse logistics by hybrid genetic algorithm, *Computers & Industrial Engineering*, 56, 951–964

Lu, Z., e Bostel, N. (2007), A facility location model for logistics systems including reverse flows: The case of remanufacturing activities, *Computers & Operations Research*, 34, 2, 299–323.

Min, H., Ko, H.J., e Ko, C.S. (2006), A genetic algorithm approach to developing the multi-echelon reverse logistics network for product returns, *Omega*, 34, 56–69.

Mutha, A., e Pokharel, S. (2009), Strategic network design for reverse logistics and remanufacturing using new and old product modules, *Computers and Industrial Engineering*, 56, 334–346.

Pishvaee, M.S., e Torabi, S.A. (2010), A possibilistic programming approach for closed-loop supply chain network design under uncertainty, *Fuzzy Sets and Systems*, 161, 2668 – 2683.

Pishvaee, M.S., Farahani, R.Z., e Dullaert , W. (2010), A memetic algorithm for bi-objective integrated forward/reverse logistics network design, *Computers & Operations Research*, 37, 1100–1112.

Poli, R. (2008), Analysis of the Publications on the Applications of Particle Swarm Optimisation, *Journal of Artificial Evolution and Applications*, Article ID 685175, 10 pages. doi:10.1155/2008/685175.

Rogers, D.S., e Tibben-Lembke, R.S., *Going backwards: reverse logistics trends and practices*, Reverse Logistics Executive Council, Reno, 1998.

Rubio, S., Chamorro, A., e Miranda, F. (2008), Characteristics of the research on reverse logistics (1995-2005), *Int J Prod Res*., 46, 4, 1099-1120.

Salema, M.I.G., Barbosa-Povoa, A.P., e Novais, A.Q. (2010), Simultaneous design and planning of supply chains with reverse flows: a generic modelling framework, *European Journal of Operational Research*, 203, 2, 336–349.

Zarei, M., Mansour, S., Kashan, A.H., e Karimi, B. (2010), Designing a Reverse Logistics Network for End-of-Life Vehicles Recovery, *Mathematical Problems in Engineering*, Article ID 649028, 1-16.

Zhou, Y-S. e Wang, S-Y. (2008), Generic Model of Reverse Logistics Network Design, *J Transpn Sys Eng & IT*, 8, 3, 71-78.