ABORDAGEM MULTIOBJETIVA PARA ANÁLISE DE COMUNALIDADE EM SISTEMAS ESPACIAIS

Anderson Cattelan Zigiotto
Instituto de Aeronáutica e Espaço
Praça Mal. Eduardo Gomes, 50, CEP 12228-904, São José dos Campos, SP
zigiottoacz@iae.cta.br

Roberto d'Amore
Instituto Tecnológico de Aeronáutica
Praça Mal. Eduardo Gomes, 50, CEP 12228-900, São José dos Campos, SP
damore@ita.br

RESUMO

Sistemas espaciais tipicamente possuem baixo volume de produção e longo ciclo de projeto, tornando o custo de pesquisa e desenvolvimento excepcionalmente grande. Uma alternativa para diminuir estes custos é o uso de comunalidade, que consiste em substituir itens diferentes por um item comum, poupando esforços de P&D. No entanto, isto resulta em excesso de funcionalidade. O projetista deve escolher a melhor solução de compromisso entre economia e excesso de funcionalidade. Trabalhos existentes utilizam algoritmos de busca local para minimizar uma combinação destes dois objetivos conflitantes. No entanto, em sistemas que estão em projeto os pesos relativos não são conhecidos, sendo necessária uma abordagem multiobjetiva. Neste trabalho utilizamos um Algoritmo Genético Multiobjetivo para buscar as melhores alternativas de comunalidade de acordo com o modelo proposto. O algoritmo é capaz de encontrar a Frente de Pareto fornecendo ao projetista um conjunto de alternativas de projeto.

PALAVARAS CHAVE. Sistemas espaciais, análise de comunalidade, otimização multiobjetivo.

ABSTRACT

Space systems typically have reduced production volume and long design cycles, thus making research and development costs exceptionally large. An alternative to lower those costs is the use of commonality, which consists in replacing different items by a common one, hence saving R&D efforts. It results in excess of functionality, though. The project designer must choose the best tradeoff between costs saving and excess functionality. Existing work use algorithms based on local decisions to minimize a combination of those two conflicting objectives. However, relative weights are not known for systems that are still at the design phase, consequently a multiobjective approach is needed. In this work, we use a Multiobjective Genetic Algorithm to search for the best commonality alternatives according to the proposed model. The algorithm is able to find the Pareto Front and supply the designer with a set containing project alternatives.

KEYWORDS. Space systems, commonality analysis, multiobjective optimization.

1. Introdução

Sistemas e equipamentos para uso em aplicações espaciais geralmente possuem características bem diferentes dos produtos de consumo, notadamente no que diz respeito ao volume de produção e ciclo de projeto. Para estes, as empresas buscam reduzir ao máximo seu custo unitário através de um volume alto de produção com poucas variantes de produtos – se possível pertencentes à mesma "família" – e ciclo curto do projeto ao mercado e por fim ao descarte. Para os sistemas espaciais, no entanto, geralmente ocorre o oposto. Os equipamentos são especializados para executar determinada função, o volume de produção é baixo, até unitário, e o ciclo de desenvolvimento é extenso.

Essa particularidade decorre principalmente da baixa quantidade de missões espaciais desenvolvidas. Assim, um equipamento pode ser produzido para uma única missão, como no caso de sondas e satélites científicos, até no máximo algumas dezenas de missões, como para uso em veículos lançadores, de sondagem e cargas úteis, bem como em frotas de satélites de comunicação ou navegação. Desta maneira, a produção destes equipamentos não é favorecida por economia de escala ou pelo aprimoramento do processo de produção.

Situação semelhante ocorre com o tempo para desenvolvimento. Devido à alta confiabilidade exigida dos sistemas espaciais, o ciclo de desenvolvimento dos sistemas e equipamentos espaciais é muito extenso, pois inclui especificação criteriosa, diversas revisões críticas e ensaios funcionais e ambientais.

Para sistemas espaciais os custos de pesquisa e desenvolvimento são excepcionalmente grandes em comparação com os outros fatores. Uma solução que pode ser utilizada para diminuir os custos do desenvolvimento de um sistema é utilizar comunalidade, ou seja, tornar comuns itens diferentes. Para um sistema, pode-se substituir equipamentos especializados por um único equipamento com capacidade para realizar as funções dos substituídos. No nível dos equipamentos, componentes com valores semelhantes podem ser agrupados, diminuindo assim a variedade de partes e simplificando o controle de estoque.

Dependendo do número de itens o projetista, ou engenheiro de sistemas, pode necessitar de um método de auxílio à decisão sobre quais itens devem ser tornados comuns. Thomas (1991) propôs tratar a análise de comunalidade como um problema de particionamento. Neste modelo os itens pertencentes a uma mesma partição são agrupados em um item comum. Como exemplo de aplicação foi realizada a análise de comunalidade em equipamentos da Estação Espacial Internacional – EEI – tais como conectores, tanques d'água e motores elétricos. Para encontrar a partição que representa a solução com menor custo foi utilizado o método de Ward (1963).

Métodos de agrupamento, embora simples e rápidos, possuem algumas desvantagens. Eles se baseiam apenas em buscas locais e as decisões de agrupamento são irreversíveis, o que pode impedir o algoritmo de explorar outras regiões do espaço de soluções. Além disso, o método tal como apresentado funciona apenas com objetivo único, enquanto que a análise da comunalidade apresenta dois objetivos conflitantes.

Quando dois ou mais itens são agrupados em outro item que os substituirá, há perdas e ganhos para o sistema. Os ganhos geralmente são contabilizados principalmente nos menores custos de pesquisa e desenvolvimento, mas também com economia de produção e de operação. As perdas decorrem do excesso de funcionalidade do item comum em relação aos substituídos, que são especializados. Desta maneira, a busca por soluções de comunalidade apresentam dois objetivos distintos e conflitantes: minimizar o custo para o sistema o minimizar o excesso de funcionalidade. Além disso, há uma diferença de grandezas envolvidas nos dois objetivos. Enquanto que o primeiro pode ser descrito em unidade monetária, o segundo trata de aspectos técnicos e geralmente é descrito em unidades como potência ou volume.

Thomas (1989) utiliza uma combinação linear para reunir estes dois objetivos. Os pesos são tais que transformam o excesso de funcionalidade em unidade monetária, ou seja, representam o custo de cada quilograma ou Watt em excesso. Assim, com apenas um objetivo a ser minimizado, pode-se utilizar uma gama extensa de algoritmos de otimização. Esta

abordagem, entretanto, possui uma dificuldade crucial: como definir este vetor de pesos. Isto porque os custos monetários do excesso de funcionalidade nem sempre são conhecidos ou passíveis de cálculo e um erro na escolha dos pesos pode resultar em soluções que não são, na prática, as melhores.

Uma abordagem que separe de fato os objetivos evita a necessidade de se arbitrar os pesos da combinação linear. Desta maneira, um algoritmo ou heurística para otimização multiobjetivo pode ser utilizado. Na seção 2 é apresentado o problema de comunalidade e seu modelo. Na seção 3 é definido o Algoritmo Genético Multiobjetivo a ser utilizado para resolver um problema exemplo na seção 4. A seção 5 apresenta as conclusões.

2. Modelo para análise de comunalidade

Tornar itens comuns em um sistema reduz custos de pesquisa e desenvolvimento, pois substitui diferentes itens especializados por um único item comum. Custos de produção também podem ser reduzidos através da aquisição de quantidades maiores de insumos e do aumento na produtividade devido à curva de aprendizado. A diminuição da variedade de itens e, por conseguinte, de procedimentos e manuais, reduz os custos de operação. No entanto, quando dois itens A e B são agrupados e substituídos por um item comum AB, este deve satisfazer os requisitos funcionais dos dois itens especializados. Considerando A e B diferentes, AB inevitavelmente terá funções, dimensões ou componentes em excesso em relação ao item A e em relação ao item B. Este excesso de funcionalidade resulta em menor desempenho, existência de componentes não utilizados, aumento de massa, volume, consumo de energia, e taxa de falhas.

É tarefa do projetista do sistema considerar estas vantagens e desvantagens e escolher os componentes ou equipamentos para os quais é proveitoso utilizar comunalidade. No caso de sistemas mais simples e com poucos equipamentos, essa decisão pode ser feita manualmente. Por exemplo, em uma análise com apenas dois itens, A e B, existem duas alternativas, ou partições, possíveis: (a) desenvolver e produzir os dois itens, A e B, resultando na solução sem comunalidade [{A},{B}]; (b) desenvolver e produzir um único item AB, o qual realiza as funções dos dois itens substituídos, resultando na solução de comunalidade total [{AB}]. Se o número de itens a ser analisado for igual a três, A, B e C, existem cinco soluções:

i. item A, item B e item C, partição [{A},{B},{C}];
ii. item AB e item C, partição [{AB},{C}];
iii. item AC e item B, partição [{AC},{B}];
iv. item BC e item A, partição [{A},{BC}];
v. item ABC, partição [{ABC}].

O número de maneiras possíveis de se particionar um conjunto com n elementos em k subconjuntos distintos e não vazios é conhecido como Número de Stirling do segundo tipo, dado por:

$$S(n,k) = \left\{ {n \atop k} \right\} = \frac{1}{k!}\sum_{j=0}^{k}(-1)^j \binom{k}{j}(k-j)^n \quad (1)$$

Considerando que a análise da comunalidade pode resultar em um número qualquer de partições, até n, a quantidade de soluções possíveis é igual à soma para todos os valores de k dos Números de Stirling do segundo tipo, também chamada de Número de Bell,

$$B_n = \sum_{k=0}^{n} \left\{ {n \atop k} \right\} \quad (2)$$

Para 25 objetos o Número de Bell é aproximadamente $4{,}6.10^{18}$ e para 40 objetos $1{,}6.10^{35}$. O problema é, portanto, combinatório e não é viável utilizar enumeração, ou busca exaustiva para resolvê-lo.

Neste trabalho a análise de comunalidade será tratada como um problema de particionamento. As soluções do problema são partições ou agrupamentos

$$C = \{C_1, \ldots, C_K\}, \ (K \leq \mathrm{N}) \quad (3)$$

sendo N o número de itens e K o número de agrupamentos, do conjunto de dados

$$X = \{\underline{x}_1, \ldots, \underline{x}_j, \ldots, \underline{x}_N\} \quad (4)$$

em que

$$\underline{x}_j = \{x_{j,1}, x_{j,2}, \ldots, x_{j,d}\}^T \in \Re^d$$

sendo d a dimensão do espaço dos dados de entrada e cada elemento x_{ji} chamado atributo, característica ou variável. São condições para C formar um conjunto de partições de X

$$C_i \neq \varnothing,\ i = 1, \ldots, K \quad (5)$$

$$C_1 \cup C_2 \cup \ldots \cup C_K = X \quad (6)$$

$$C_i \cap C_j = \varnothing,\ i,j = 1, \ldots, K \text{ e } i \neq j\ . \quad (7)$$

Pode-se definir que o item projetado para substituir dois ou mais itens em um agrupamento k seja representado de maneira análoga, por um vetor de agrupamento

$$\underline{\mu}_k = \{\mu_{k,1}, \mu_{k,2}, \ldots, \mu_{k,d}\}^T\ . \quad (8)$$

Os atributos deste vetor agrupamento são definidos através de um operador definido na forma

$$\underline{\beta} = \{\beta_1, \beta_2, \ldots, \beta_d\}^T\ . \quad (9)$$

Por exemplo, para dois itens A e B, o vetor agrupamento é dado por

$$\underline{\mu}_{AB} = \{\mu_{AB,1}, \mu_{AB,2}, \ldots, \mu_{AB,d}\} \quad (10)$$

em que cada elemento é determinado por

$$\mu_{AB,i} = x_{A,i}\, \beta_i\, x_{B,i}\ ,\ i = 1, 2, \ldots, d \quad (11)$$

ou, em notação vetorial,

$$\underline{\mu}_{AB} = \underline{x}_A\, \underline{\beta}\, \underline{x}_B\ . \quad (12)$$

Para atributos ou requisitos relativos a capacidade, tais como potência, o operador β_i tipicamente será uma função máximo. A quantidade de cada item será definida em um vetor específico

$$Q = \{q_1, \ldots, q_n\}. \quad (13)$$

Estas definições podem ser exemplificadas a partir do problema apresentado por Thomas (1991) que aplica a análise de comunalidade aos tanques d'água da EEI. A Tabela 1 lista os requisitos, quantidade e custos de P&D e produção para esses equipamentos.

Tabela 1 - Tanques d'água da EEI (Thomas, 1991).

Tanque	Quantidade	C_{PeD} (1000 $)	C_{Prod} (1000 $)	Peso (lb)	Volume (ft^3)
1	2	92,82	102,51	3,23	16,70
2	6	366,68	810,76	27,87	135,67
3	4	67,83	64,60	1,98	9,61
4	2	355,77	775,18	26,58	129,41
5	4	71,86	70,60	2,16	10,53
6	1	49,37	40,20	1,20	5,83
7	1	46,17	36,12	1,08	5,25
8	4	464,31	1152,11	40,36	196,50
9	3	378,24	844,66	29,26	142,42

No exemplo existem 4 atributos: $x_{j,1}$ contém o custo de pesquisa e desenvolvimento; $x_{j,2}$ o custo de produção unitário; $x_{j,3}$ é o peso em libras e $x_{j,4}$ é o volume em pés cúbicos do j-ésimo item. Se, por exemplo, decidirmos que os tanques 2 e 4 serão agrupados, então teremos

$$\underline{x}_2 = \{366{,}68\ ,\ 810{,}76\ ,\ 27{,}87\ ,\ 135{,}67\ \}^T \quad (14)$$

e

$$\underline{x}_4 = \{355{,}77\ ,\ 775{,}18\ ,\ 26{,}58\ ,\ 129{,}41\}^T\ . \quad (15)$$

Admitindo-se que um tanque maior substitui um tanque menor, as funções de agrupamento são do tipo "máximo" e

$$\underline{\beta} = \{\max(), \max(), \max(), \max()\} \quad (16)$$

logo

$$\underline{\mu}_{[2,4]} = \{\max(x_{2,1}, x_{4,1}), \max(x_{2,2}, x_{4,2}), \max(x_{2,3}, x_{4,3}), \max(x_{2,4}, x_{4,4})\}^T \quad (17)$$

$$\underline{\mu}_{[2,4]} = \{366{,}68\ ,\ 810{,}76\ ,\ 27{,}87\ ,\ 135{,}67\}^T. \quad (18)$$

O ganho α ou ganho de comunalidade, obtido com o agrupamento dos *N* itens, é dado por (19). Ele é um número não-positivo por convenção, já que consiste em redução de custos, e é composto por duas parcelas.

$$\alpha = \sum_{k=1}^{K} \alpha_k = \alpha' + \alpha'' \quad (19)$$

A primeira parcela, mostrada em (20), consiste na economia dos esforços de pesquisa e desenvolvimento que seriam empregados nos itens substituídos.

$$\alpha' = \sum_{k=1}^{K} \left[\mu_{k,1} - \left(\sum_{i \in k} x_{i,1} \right) \right] \quad (20)$$

A segunda parcela, dada em (21), deverá ser considerada somente se houver uma curva de aprendizado para produção. Ela consiste na redução dos custos de produção devido ao aumento no número de itens idênticos formados pelos agrupamentos.

$$\alpha'' = \sum_{k=1}^{K} \left[\mu_{k,2} . \left(QC_k - \sum_{j=1}^{QC_k} j^{log_2 LC} \right) - \sum_{i \in k} x_{i,2} . \left(q_i - \sum_{j=1}^{q_i} j^{log_2 LC} \right) \right] \quad (21)$$

em que

$QC_k = \sum_{i \in k} q_i$, é a quantidade de itens no agrupamento *k* e

$0 < LC < 1$, é a constante de aprendizagem.

O cálculo do aumento de custo σ devido ao excesso de funcionalidade em todos os agrupamentos é um número não-negativo, descrito em (22)

$$\sigma = \sum_{k=1}^{K} \sigma_k = \sum_{k=1}^{K} \sum_{i \in k} q_i \left\{ \sum_{j=1}^{d} w_j \left| x_{i,j} - \mu_{k,j} \right| \right\} \quad (22)$$

em que w_j são os pesos relativos de cada atributo no cálculo do excesso de funcionalidade.

O agrupamento de itens resulta no aumento, em termos absolutos, da economia α bem como do excesso de funcionalidade σ. Nos trabalhos existentes os pesos w_j são tais que estas duas parcelas são dadas em unidades monetárias podendo, assim, ser reunidas em um único objetivo. Desta maneira, a melhor solução de comunalidade é a partição *S* que minimiza a função objetivo

$$z(S) = \sigma + \alpha \quad (23)$$

Esta abordagem, entretanto, exige que os custos monetários do excesso de funcionalidade sejam conhecidos e confiáveis. Em sistemas espaciais, saber o custo real de cada unidade de massa, volume ou consumo de energia é um desafio até para lançadores com grande histórico de missões e com bom sistema de gerenciamento de custos. No caso de lançadores e sistemas que ainda estejam em estudo, ou que possuam poucos vôos, os valores dos pesos seriam decorrentes de uma estimativa pouco confiável. Desta maneira, optamos por utilizar um método multiobjetivo, de maneira a não depender da correta atribuição dos pesos da combinação linear que forma a função objetivo.

3. Algoritmo genético multiobjetivo para análise de comunalidade.

Para buscar a partição que representa a melhor solução de comunalidade, é possível utilizar algoritmos de agrupamento de diversos tipos (Xu e Wunsch II, 2005) (Jain *et al.*, 1999).

Hruschka *et al.* (2009) descreve alguns algoritmos evolutivos de agrupamento. Dentre estes, destacam-se os algoritmos genéticos – AGs – conforme mostra Naldi *et al.* (2008), que se podem ser adaptados a diversos tipos de problema, embora não sejam algoritmos exatos, ou seja, não garantam que a melhor solução será encontrada.

Os AGs são métodos de busca e otimização inspirados no processo de evolução dos organismos vivos. Cada indivíduo de uma população corresponde a uma solução do problema, sendo codificada em uma estrutura de dados chamada cromossomo.

O algoritmo começa com uma população inicial formada por certo número de indivíduos, ou cromossomos. Esta população é avaliada e a cada indivíduo é atribuído um valor de *aptidão*, o qual representa a qualidade desta solução. Os indivíduos mais aptos possuem maior probabilidade de serem escolhidos para gerar descendentes através de um operador de cruzamento. A fim de garantir a diversidade da população e permitir que outras regiões do espaço de soluções sejam visitadas, é aplicado aos descendentes um operador de *mutação*. Este operador altera aleatoriamente um parâmetro do indivíduo e é aplicado com uma probabilidade baixa. A nova geração passa novamente pelos processos de avaliação, seleção, cruzamento e mutação até que determinada condição de parada seja atingida.

No caso de algoritmos com mais de um objetivo, o conceito de um indivíduo ser melhor que o outro não é baseado em um valor simples de aptidão, mas sim utilizando-se o conceito de dominância. Um indivíduo da população domina outro se for igual em todas as funções objetivos e melhor em pelo menos uma delas. Um conjunto de soluções não dominadas é chamado de Frente de Pareto e algoritmos de otimização multiobjetivo devem buscar o conjunto de soluções não dominadas e bem distribuídas que mais se aproxima desta frente.

Para um problema qualquer ser resolvido por AG deve-se definir, além destes operadores, a representação dos cromossomos. Em seu trabalho pioneiro no uso de AGs para problemas de agrupamento, Falkenauer (1994) afirma que as representações e operadores normalmente utilizados não são a melhor escolha para problemas de agrupamento, pois atuam nos alelos do cromossomo e ignoram a existência dos grupos.

Logo, é necessário definir uma representação adequada bem como operadores de cruzamento e mutação que lidem com grupos. Em seu Algoritmo Genético para Agrupamento – CGA – ele propõe uma codificação baseada em um vetor de inteiros, onde o índice representa o item e o valor representa o agrupamento ao qual este item pertence. A representação ainda contém uma segunda parte com uma lista dos agrupamentos encontrados na primeira parte, onde o operador cruzamento é realizado. Por exemplo, o cromossomo [1,2,3,4,1,1,1,1,1] : [1,2,3,4] representa a partição [{1,5,6,7,8,9},{2},{3},{4}]. Esta representação possui um problema: uma partição, ou solução, pode ser representada por cromossomos diferentes. No exemplo, a mesma partição pode ser representada por [2,3,4,1,2,2,2,2,2] : [2,1,4,3], ou por qualquer permutação de {1,2,3,4}. Em um algoritmo genético isto é chamado de degeneração da população, embora alguns autores usem o termo redundância. Uma representação baseada em lista encadeada também resultaria no mesmo problema.

Para garantir que cada solução seja representada por um único cromossomo, Tucker *et al.* (2005) e *Du et al.* (2006) utilizam representação baseada em *restricted grouwth functions* – RGF, embora o último a chame *restricted growth string* – RGS. Uma RGF é uma função

$$f : [n] \rightarrow [n] \tag{24}$$

tal que

$f(1) = 1$ e

$f(i+1) \leq \max\{f(1), \ldots, f(i)\} + 1$

Assim, por exemplo, [1,1,2,3,2,4] é uma RGF, mas [4,4,3,2,3,1] não, existindo uma relação unívoca entre uma partição e sua representação. Outros autores também utilizam representações que evitam a degeneração, tais como Korkmaz (2010) que adiciona restrições à representação em lista encadeada com esta finalidade.

Neste artigo utilizaremos a representação RGF para os cromossomos. Todas as funções, ou operadores, devem retornar funções RGF. Se, em alguma operação, as condições dadas em (24) não forem satisfeitas, é necessário realizar a retificação do cromossomo.

Vários trabalhos na literatura apresentam algoritmos genéticos de agrupamento para resolver problemas específicos, geralmente comparando-os com outros algoritmos ou heurísticas. Muitos operadores também são específicos de cada problema. Neste trabalho iremos utilizar e comparar os resultados obtidos pelos cruzamentos RGFGA de Tucker *et al.* (2005) e LLE de Korkmaz (2010).

O operador mutação é praticamente um consenso na literatura e consiste em, aleatoriamente, realizar uma das seguintes funções: (a) unir dois grupos quaisquer; (b) dividir um grupo; (c) mover um elemento de um grupo para outro.

O algoritmo deverá encontrar a partição S que minimiza duas funções objetivos: $f_1(S)$ igual ao ganho de comunalidade, não positivo, α dado em (19); e $f_2(S)$ igual ao excesso de funcionalidade σ descrito em (22). Para esta última, um vetor com os pesos relativos w_j de cada atributo deverá ser definido.

Seria possível ainda utilizar uma função objetivo em separado para cada um dos atributos existentes em σ, evitando a necessidade de definir estes pesos. No entanto, isto traria algumas complicações. Estudos indicam que os algoritmos multiobjetivo perdem sua efetividade quando o número de objetivos é maior que dois ou três (Ishibuchi *et al.* 2008). Assim, uma abordagem específica tornou-se necessária com a criação de uma nova área de estudos para algoritmos com vários objetivos. Ainda que existam trabalhos relatando bons resultados nessa área, uma grande quantidade de objetivos também demanda populações maiores para melhor preencher o conjunto de soluções não dominadas. Desta forma, o número de soluções retornadas pelo algoritmo pode ser muito grande para ser analisado pelo projetista, inviabilizando a proposta deste trabalho. Desta maneira utilizaremos apenas os dois objetivos descritos.

Neste trabalho utilizaremos o algoritmo NSGA-II proposto por Deb *et al.* (2002), o qual é um dos algoritmos genéticos multiobjetivo existentes mais estudados e utilizados. O NSGA-II ordena as soluções através do conceito de dominância, utilizando ainda um operador para calcular a distribuição das soluções na frente de Pareto. A seleção dos indivíduos escolhidos para cruzamento se dá através de torneio, em que o indivíduo dominante é selecionado.

Para gerar a população inicial é utilizada uma função que gera cromossomos aleatoriamente, respeitando as regras para RGF. O tamanho da população adotado é igual a quatro vezes o número itens, ou seu número par superior. O critério de parada escolhido é o de número máximo de gerações, no caso igual a dez vezes o número de itens, ou o número par superior.

4. Exemplo de aplicação

O algoritmo definido na seção 3 foi utilizado no problema de comunalidade de motores elétricos da EEI apresentado por Thomas (1991). Na Tabela 2 são listadas as especificações do problema, incluindo quantidades, de motores elétricos necessários para a EEI, bem como seus custos de pesquisa e desenvolvimento e produção e demais atributos.

Os objetivos são: minimizar, ou maximizar em valor absoluto, $f_1(S)$, igual à economia obtida ao se agrupar motores; e minimizar $f_2(S)$ que é o custo do excesso de funcionalidade. É utilizado o mesmo vetor de pesos do exemplo W = [0; 1; 0,5; 2,8; 10]. Admitindo-se que um motor maior substitui um menor, a função de agrupamento β é do tipo máximo para todos os atributos. Foi utilizada taxa de aprendizado de 85% para cálculo dos custos de produção.

A Fig. 1 mostra o resultado obtido com o algoritmo NSGA-II, executado em Matlab, utilizando a codificação e os operadores descritos na seção 3. O operador de cruzamento é RGFGA com taxa de 70% e a taxa de mutação é de 30%. No exemplo o tamanho da população é igual a 112 indivíduos. A porcentagem da população escolhida para formar a frente de Pareto é de 50%, de forma que o vetor de soluções possui até 56 elementos. Após 280 gerações a execução do algoritmo é interrompida. Além da totalidade da população inicial, são mostradas as frentes de Pareto com seus tempos de execução para as gerações 10, 25, 100 e 280. O algoritmo foi executado em um processador i3 550 de 3.2 GHz sem uso de paralelismo.

Tabela 2 - Dados do exemplo com 27 motores da EEI (Thomas, 1991)

Motor	Quantidade	C_{PeD} (1000 $)	C_{Prod} (1000 $)	Potência (W)	Peso (lb)	Volume (ft^3)
1	12	18,32	8,71	3,5	0,350	0,004
2	4	24,83	11,80	6,6	0,554	0,006
3	22	26,80	12,73	6,5	0,012	0,001
4	4	23,72	11,27	6,8	0,517	0,006
5	2	28,41	13,5	7,8	0,679	0,008
6	8	26,17	12,44	10,0	0,600	0,007
7	2	39,56	18,8	20,0	1,120	0,013
8	3	46,02	21,87	32,0	1,408	0,016
9	9	44,63	21,21	32,0	1,344	0,015
10	4	66,36	31,53	68,0	2,448	0,028
11	1	118,49	56,31	280,0	5,880	0,068
12	7	84,06	39,95	250,0	3,500	0,041
13	4	94,84	45,07	300,0	4,200	0,049
14	2	87,29	41,48	285,0	3,705	0,043
15	5	105,02	49,91	350,0	4,900	0,057
16	4	118,29	56,21	510,0	5,865	0,068
17	2	114,86	54,58	510,0	5,610	0,065
18	4	54,14	25,73	200,0	1,800	0,021
19	4	35,72	16,98	98,0	0,960	0,011
20	8	31,66	15,05	80,0	0,800	0,009
21	4	33,44	15,89	79,0	0,869	0,010
22	8	31,77	15,10	67,0	0,804	0,009
23	22	33,16	15,76	66,0	0,858	0,010
24	22	18,36	8,72	13,0	0,351	0,004
25	2	20,02	9,51	12,5	0,400	0,005
26	22	20,84	9,90	12,5	0,425	0,005
27	4	45,85	21,79	200,0	1,400	0,016

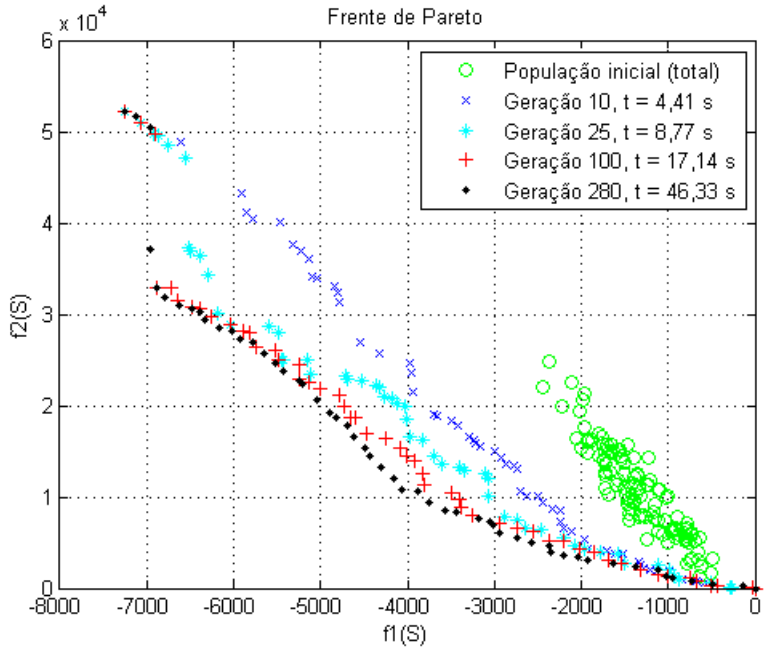

Figura 1 - Resultado do AG multiobjetivo para o problema com 27 motores da EEI.

É possível notar ainda na Fig. 1 que, embora os cromossomos sejam RGFs aleatórias no espaço de atributos, no espaço de objetivos os elementos da população inicial não se encontram uniformemente distribuídos. Ainda assim, o algoritmo foi capaz de encontrar um conjunto de soluções bem distribuídas sobre a frente de Pareto.

Na Fig. 2 são comparados os resultados obtidos por dois operadores de cruzamento, RGFGA e LLE.

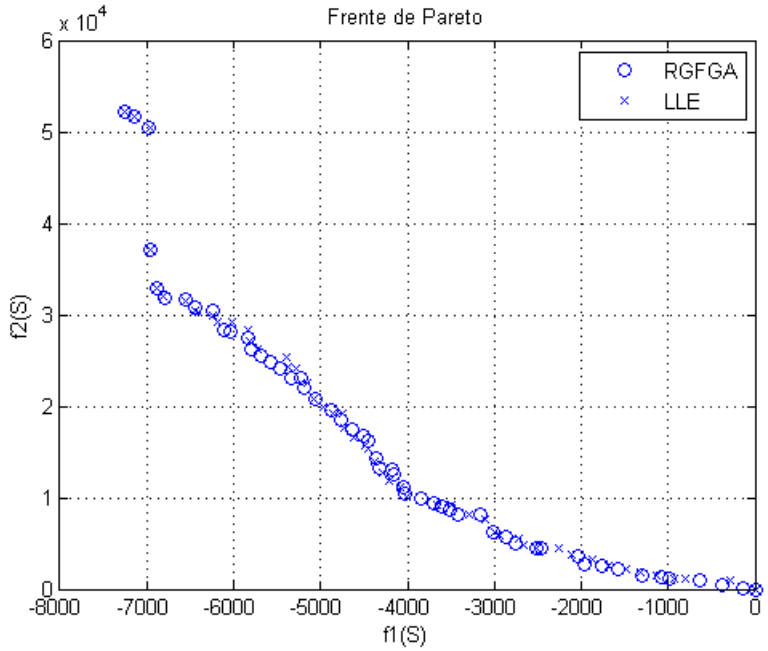

Figura 2 - Comparativo entre cruzamento RGFGA e LLE para o problema dos 27 motores.

A Fig. 2 mostra que os resultados para os dois operadores são semelhantes. Em uma avaliação qualitativa pode-se dizer que o desempenho do cruzamento RGFGA foi ligeiramente superior ao do LLE para este caso.

Há algumas soluções coincidentes nos dois extremos: comunalidade total e nenhuma. É esperado que estas soluções sejam não dominadas por conterem os menores valores possíveis dos objetivos. A solução que apresenta comunalidade total, ou seja, onde um único tipo de motor substitui todos os outros, possui a maior economia possível. No outro lado, a solução inicial com motores únicos e comunalidade zero, possui o menor valor possível de excesso de funcionalidade, também nulo. O algoritmo mostrou eficiência ao encontrar estes dois pontos para ambos operadores de cruzamento.

5. Conclusões

Neste artigo foi apresentado um algoritmo genético multiobjetivo que resultou em um conjunto de soluções satisfatórias para o problema da comunalidade de motores elétricos em sistemas espaciais. Ele se mostrou efetivo em obter a frente de Pareto para dois operadores de cruzamento diferentes. Para o algoritmo foram definidas também a representação e a mutação adequadas ao tipo de problema. As soluções encontradas devem, então, passar pela análise do especialista, ou engenheiro de sistemas, para que e melhor seja escolhida. O uso de comunalidade permite diminuir o custo de um sistema espacial sem recorrer à diminuição na qualidade de componentes.

Embora o modelo do problema apresente objetivos conflitantes, os trabalhos existentes buscavam minimizar uma combinação linear destes. Esta combinação exige que os pesos relativos a cada objetivo sejam conhecidos e que eles convertam todos os fatores para a mesma unidade. Para sistemas com pouco histórico de utilização ou que estejam em fase de projeto, estes pesos são desconhecidos ou são estimados com pouco grau de certeza. Ao efetuar uma

abordagem realmente multiobjetiva, evitamos os erros que possam ser ocasionados pela escolha errada destes números.

Outra contribuição consiste na utilização de uma metaheurística de otimização global, ao invés de algoritmos que trabalham somente com buscas locais como o método de agrupamento hierárquico.

Referências

Deb, K., Pratap, A., Agarwal, S. e Meyarivan, T. (2002), A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, 6 (2), 182-197.

Du, J., Alhajj, R. e Barker, K. (2006), Genetic algorithms based approach to database vertical partition, *Journal of Intelligent Information Systems*, 26, 167-183.

Falkenauer, E. (1994), New Representation and Operators for GAs Applied to Grouping Problems, *Evolutionary Computation*, 2 (2), 123-144.

Hruschka, E. R., Campello R. J. G. B., Freitas, A. A. e de Carvalho A. C. P. L. F. (2009), A survey of evolutionary algorithms for clustering, *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews*, 39 (2),133-155.

Ishibuchi, H., Tsukamoto, N. e Nojima, Y. (2008), Evolutionary many-objective optimization: A short review, *Proc. of 2008 IEEE Congress on Evolutionary Computation*, 2419-2426.

Jain, A. K., Murty, M. N. e Flynn, P. J. (1999), Data clustering: a review, *ACM Computing Surveys*, 31,264-323.

Korkmaz, E. E. (2010), Multi-Objective Genetic Algorithms for Grouping Problems, *Applied Intelligence*, 33 (2), 179-192.

Naldi, M. C., de Carvalho, A. C. P. L. F., Campello, R. J. G. B. e Hruschka, E. R., Genetic Clustering for Data Mining, em *Soft Computing for Knowledge Discovery and Data Mining*, 113-132, Springer US, 2008.

Thomas, L. D. (1991), Commonality Analysis Using Clustering Methods, *Operations Research*, 39, 677- 680.

Thomas, L. D., A methodology for commonality analysis, with applications to selected Space Station systems, *NASA Technical Memorandum 100364*, 1989.

Tucker , A., Crampton, J. e Swift, S. (2005), RGFGA: An Efficient Representation and Crossover for Grouping Genetic Algorithms, *Evolutionary Computation*, 13, 477-499.

Ward, J. H. (1963), Hierarchical Grouping to Optimize an Objective Function, *Journal of the American Statistical Association*, 58, 236-244.

Xu, R. e Wunsch II, D. (2005), Survey of clustering algorithms, *IEEE Transactions on Neural Networks*, 16,645-678.