

ALGORITMO GENÉTICO APLICADO À SELEÇÃO DE COLUNAS NO PROBLEMA DE ALOCAÇÃO DE TRIPULAÇÕES

Marco Antonio Moreira de Carvalho

Instituto Tecnológico de Aeronáutica - ITA. Divisão de Pós-Graduação em Engenharia Eletrônica e Computação
Av. Tivoli, número 183, apartamento 44, Vila Betânia, São José dos Campos, SP.
Bolsista CNPq
mamc@ita.br

André Gustavo dos Santos

Universidade Federal de Minas Gerais – UFMG.
Av. Antonio Carlos, 6627, Belo Horizonte, MG.
andre@dcc.ufmg.br

Resumo. Este trabalho apresenta o desenvolvimento de um algoritmo genético no problema de alocação de tripulações, especificamente no subproblema de seleção de colunas para cobrir os vários trechos de viagem, uma versão multiobjetivo do set covering problem (SCP). Apresentamos justificativas para se trabalhar com a versão multiobjetivo, dado o contexto do problema, encontrando um conjunto de soluções eficientes, um apoio para decisão de um agente externo. Mostramos as idéias incorporadas ao algoritmo genético clássico, e os resultados comprovam as melhorias obtidas em termos da qualidade da solução e do número de pontos na solução Pareto-Ótima. O problema mestre de geração de colunas também é descrito, na forma de um problema de caminho mínimo com restrições de recursos, bem como um algoritmo pseudo-polinomial para solucioná-lo na forma multiobjetivo. O conjunto dos algoritmos de geração e seleção de colunas se comporta como um método branch-and-price.

Palavras chave: Inteligência artificial, Metaheurísticas, Alocação de tripulações, Seleção de colunas, Algoritmo genético.

1. Introdução

O problema de alocação de tripulações é classificado como um problema NP-difícil (Souza et al., 2003) e consiste em distribuir tarefas aos tripulantes (pilotos, maquinistas, motoristas, aeromoças, cobradores, etc.) durante vários trechos de viagens a serem cobertos por empresas aéreas, ferroviárias ou rodoviárias, ou mesmo designar atividades pré-definidas (treinamentos, exames médicos e folgas, entre outros), gerando escalas (normalmente mensais) individuais para cada tripulante.

A distribuição deve ser feita de forma a otimizar a escala de cada tripulante, minimizando assim os gastos com tripulações, que, segundo Cabral et al. (2000) representam aproximadamente 20% dos gastos operacionais de uma empresa aérea, por exemplo. Devem ser respeitadas ainda as legislações federais, sindicais e normas das empresas, que regulam a relação entre empresas de transporte e tripulantes, de tal forma que todas as rotas oferecidas sejam cobertas pelos tripulantes.

Tipicamente, o problema é decomposto em duas fases: divisão das rotas a serem cobertas em tarefas, ou trechos, e geração das escalas dos tripulantes. A geração de escalas ainda pode ser dividida em duas: geração de colunas, técnica aplicada a problemas lineares de grandes dimensões, no caso de não se dispor de todas as colunas *a priori* (Pereira, 2002) e seleção de colunas. A primeira gera um conjunto de escalas possíveis para cada tripulante, e a segunda seleciona o melhor subconjunto de escalas, de acordo com algum objetivo. A seleção de colunas é considerada o problema mestre, e a geração de colunas é considerada um subproblema. O problema de alocação de tripulações é um problema de programação inteira.

2. Revisão da literatura

Alguns trabalhos encontrados na literatura tratam o problema de alocação de tripulações tanto em malhas aéreas (Schaefer, 2000), (Moudani et al., 2001), (Klabjan et al., 2001), (Ehrgott e Ryan, 2003), quanto em ferroviárias (Caprara et al., 1998) e rodoviárias (Fores et al., 1998). Em todos eles o problema de recobrimento de conjuntos (*set covering problem* - SCP) é usado como um subproblema. Como dito anteriormente, uma das etapas do processo de alocação de tripulações consiste em se resolver o SCP.

De acordo com Kohl e Karish (2004), na maioria das companhias européias, cada membro da tripulação indica previamente suas preferências por tarefas. Assim, fica claro que a alocação deve ser feita considerando no mínimo dois objetivos: minimizar custos operacionais da empresa e maximizar satisfação da tripulação em ter suas preferências integral ou parcialmente atendidas. Esse último também trará ganhos indiretos para empresa, pois uma tripulação mais satisfeita proporcionará melhores serviços aos clientes da companhia. Porém, podem ser objetivos conflitantes, não

havendo uma solução que atenda de forma ótima simultaneamente os dois critérios, e sim um conjunto de soluções ótimas, chamado conjunto Pareto-ótimo ou conjunto de soluções eficientes (Chankong e Haimes, 1983). Um algoritmo para esse problema deve então gerar todas as soluções eficientes já que a princípio, não há informação sobre como comparar as soluções. Elas serão posteriormente analisadas por algum agente de tomada de decisão, como o diretor da empresa, que escolherá a solução que melhor lhe satisfaz.

No mesmo trabalho, Kohl e Karish afirmam haver no mínimo quatro tipos de objetivos a se considerar: diminuir custos operacionais, aumentar satisfação da tripulação, diminuir riscos com eventuais atrasos ou ausência inesperada de tripulantes, e outras preferências particulares das empresas. Alguns trabalhos recentes começam a tratá-lo na forma multiobjetivo, como (Schaefer, 2000) e (Ehrgott e Ryan, 2002) onde se buscam soluções robustas, tratando a questão de custos operacionais por atraso e (Moudani et al., 2001), onde se considera, além dos custos, a satisfação da tripulação.

O SCP é tradicionalmente tratado na literatura em sua forma mono-objetivo. Os melhores algoritmos exatos para a versão mono-objetivo do SCP utilizam *branch-and-cut* combinado com outras técnicas (Beasley e Jornsten, 1992), (Balas, Ceria e Cornuéjols, 1996). O problema também já foi atacado por diversas técnicas heurísticas, e dentre elas citamos relaxação lagrangeana (Beasley, 1990), *simulated annealing* (Jacobs e Brusco, 1993) e algoritmos genéticos (Beasley e Chu, 1996). Entretanto, é encontrado em situações reais em sua versão multiobjetivo, por exemplo, quando usado como subproblema na solução do problema de alocação de tripulações. Resultados recentes e comparações de várias heurísticas para a versão multiobjetivo do SCP podem ser encontrados em Jaskiewicz (2004). Diante de tantos objetivos importantes, não podemos nos limitar à tradicional forma de resolver o problema apenas diminuindo os custos diretos com a alocação das tripulações, mas devemos tratar o problema como multiobjetivo, como de fato ele ocorre na prática. Aqui tratamos o problema na forma bi-objetivo, mas o algoritmo é facilmente extensível para mais de dois objetivos.

Métodos de geração de colunas têm sido largamente aplicados a problemas de programação inteira, e as abordagens variam de acordo com a disponibilidade de colunas no conjunto inicial. Em Revelle et al. (1970), encontra-se um exemplo de problema de localização de facilidades com colunas conhecidas *a priori*. Caprara, et al. (1999) e Makri e Klabjan (2001) trabalham com um conjunto inicial composto por algumas colunas, adicionando novas colunas ao longo do processo de resolução do problema. Outra abordagem consiste em inicialmente utilizar uma base artificial, sem colunas conhecidas *a priori*, como visto em Kohl (1995).

Entre os critérios de seleção de colunas destacam-se o critério de *Dantzig* (Farley, 1990), critério de *Bixby* (Bixby et al., 1992), critério de *Hu e Johnson* (Hu e Johnson, 1999), e o critério de *Gopalakrishnan* (Gopalakrishnan et al., 2001).

Neste trabalho as colunas não são conhecidas *a priori*, e foi utilizado o critério de *pricing* para seleção de colunas, no qual as colunas com menor custo reduzido são selecionadas.

3. Geração de colunas

Na geração de colunas a representação utilizada foi a de um grafo direcionado acíclico (Gamache, Soumis e Desrosiers, 1999), no qual os vértices representam dias e horários associados ao início ou fim de uma atividade, e as arestas representam atividades a serem realizadas (pré-designadas ou não), folgas semanais, folgas mensais ou simplesmente tempo inoperante (arestas de continuação). Cada aresta tem valores associados a cada recurso (tempo, custo operacional, etc.) consumido pela tarefa que representa.

Para cada tripulante, um grafo é construído inicialmente contendo todas as tarefas que possam ser desempenhadas pelo mesmo. Um caminho a partir do vértice de início do grafo (por exemplo, o início do mês) até o vértice final do grafo (fim do mês) é uma escala mensal para um tripulante. A viabilidade de um caminho é associada com a quantidade de recursos consumida por cada tarefa.

Claramente, o número de caminhos que podem ser gerados é uma função exponencial da quantidade de tarefas disponíveis para o empregado, e somente o grafo não é suficiente para garantir a viabilidade dos caminhos, pois existem restrições que envolvem todos os vértices do grafo, impedindo sua modelagem direta. Trata-se do problema de caminho mínimo com restrições de recursos (*shortest path problem with resources constraints*).

Para gerar um subconjunto de caminhos de qualidade foi utilizado o algoritmo de programação dinâmica pseudo-polinomial de fixação de *labels* (*label setting algorithm*) proposto por Desrosiers et al. (1995), que utiliza o conceito de *labels* para resolver o problema.

Gerado um subconjunto de colunas, estas são submetidas ao algoritmo genético, usando a modelagem do problema de recobrimento de conjuntos, o qual por sua vez, avalia este subconjunto e através de sua execução altera as escalas a fim de melhorá-las, gerando assim um novo subconjunto de escalas. Este novo subconjunto é submetido a um pacote de programação linear modelado como o problema de alocação de tripulações, o qual calcula o custo dual de cada escala, decidindo quais delas poderão ser incluídas em uma nova execução do *algoritmo de fixação de labels*.

O conjunto dos métodos funciona assim como o método *branch-and-price* (Barnhart, 1998), utilizado para testar um grande número de colunas implicitamente. Os algoritmos são executados até que não hajam escalas a serem adicionadas ao problema mestre.

3.1. O algoritmo de fixação de labels

O algoritmo associa a cada caminho do grafo um *label* contendo o consumo de cada recurso no vértice e um componente de custo, os quais são acumulados ao longo do caminho, somando-se os valores consumidos em cada aresta utilizada, o que possibilita a verificação da viabilidade de cada caminho e a comparação entre caminhos.

A verificação da viabilidade de cada caminho é feita comparando-se a quantidade de cada recurso consumido pelo caminho até o vértice atual com os limites superior e inferior para o consumo de cada recurso. Caso algum limite seja violado em qualquer altura do caminhamento, o caminho é considerado inviável e é descartado.

Um caminho é considerado eficiente se nenhum outro caminho o dominar, ou seja, se nenhum outro caminho tiver valores melhores para todos os recursos consumidos e para o componente de custo.

Caminhos não dominantes entre si formam o conjunto Pareto-Ótimo de escalas, no vértice final. Aqueles que tiverem custo negativo são retornados ao problema mestre, eles correspondem às novas variáveis com custo reduzido negativo no problema mestre.

4. Seleção de colunas

Neste trabalho foi considerada a versão multiobjetivo do problema, podendo, por exemplo, ter como objetivos o custo da escala de cada tripulante e a preferência do tripulante em relação a ela. Na fase de seleção de colunas não é escolhida apenas uma escala, ao invés disso, é selecionado um subconjunto de escalas, cabendo a escolha final a um agente de decisão externo que poderá privilegiar um objetivo. Isso se explica porque na versão multiobjetivo do problema tipicamente não existe uma solução que seja a melhor em todos os objetivos. A seleção de colunas é um problema de recobrimento de conjuntos (*set covering problem* - SCP), o qual aparece como o modelo mais comumente empregado para formulação de problemas de otimização de grande porte (Pereira, 2002).

A entrada do problema de recobrimento de conjuntos é uma matriz binária A de dimensões $m \times n$ e c , um vetor n -dimensional, onde c_j representa o custo da coluna j , assumindo $c_j > 0$. Podemos dizer que as colunas são instâncias e as linhas são recursos.

Dizemos que uma coluna j cobre a linha i se $a_{ij} = 1$ (onde a_{ij} representa o elemento da matriz A na linha i e na coluna j). Neste caso, a instância j utiliza o recurso i . Caso contrário $a_{ij} = 0$. O problema consiste em selecionar um subconjunto de colunas de forma que todas as linhas sejam cobertas com um custo mínimo.

Neste trabalho, as colunas representam as escalas geradas na fase de geração de colunas, sendo garantido que cada escala é viável de ser seguida por um tripulante, e as linhas são as tarefas, os trechos a serem cobertos. O objetivo é então distribuir as tarefas em escalas minimizando custos com as tripulações.

4.1. O algoritmo genético

Algoritmo genético é uma técnica de programação evolutiva utilizada em problemas de otimização complexos. A idéia por trás do algoritmo genético é emular o que existe na natureza: sobrevivência dos mais aptos. Operadores de *mutação* (*mutation*), *cruzamento* (*crossover*) e *seleção* (*selection*) são emulados. Estes operadores são aplicados a um conjunto de soluções em potencial (*população*) chamadas *cromossomos* ou *indivíduos*.

A seguir são apresentados os detalhes da implementação de algoritmo genético realizada.

4.1.1. Codificação

Os cromossomos (soluções em potencial) são representados como vetores binários de tamanho n (o número de colunas já geradas para o problema). Cada posição do vetor (correspondente a um bit) representa uma coluna. Se o valor correspondente for 1, a coluna foi selecionada e faz parte desta solução em potencial, caso contrário a coluna não foi escolhida e não faz parte da solução em potencial. A Figura 1 apresenta um exemplo da codificação utilizada.

1	0	1	1	0
---	---	---	---	---

Figura 1 - Exemplo de codificação, onde as colunas 1, 3 e 4 são escolhidas e fazem parte da solução.

4.1.2. População inicial

A população inicial é gerada de forma aleatória, sem a utilização de nenhum conhecimento específico do problema. Para cada cromossomo, cada um dos bits recebe o valor 0 ou 1, com a mesma probabilidade. Não foi utilizado nenhum conhecimento prévio do problema para geração da população inicial para evitar algum tipo de convergência prematura, pelo baixíssimo custo computacional do modo aleatório e pelos bons resultados obtidos em experimentos. Além disso, a solução inicial melhora rapidamente, podendo ser evitado gasto computacional ao tentar gerar uma população inicial direcionada.

4.1.3. Funções de avaliação

Para cada coluna selecionada no cromossomo, o valor correspondente a cada um dos objetivos é acumulado na estrutura do cromossomo, gerando o custo total de cada objetivo. Ao avaliar o cromossomo testa-se se as colunas selecionadas cobrem todas as linhas (tarefas), ou seja, se o cromossomo realmente representa uma solução em potencial. Além disso, verifica-se se existem colunas redundantes (colunas que cobrem somente linhas cobertas por outras colunas). Em ambos os casos, funções de reparo são utilizadas. Este procedimento é realizado na fase de avaliação da população para que a viabilidade seja garantida e evitar uma nova chamada da função de avaliação posteriormente.

4.1.4. Funções de reparo

As funções de reparo incluem colunas a fim de cobrir linhas não cobertas, e também retiram colunas redundantes. Para isso, ambas as funções de reparo verificam a razão entre os custos das colunas de acordo com os dois objetivos e o número de linhas cobertas, fazendo um ranking das colunas com base nas seguintes expressões:

$$(1) \quad c_j / \sum_{i=1}^n a_{ij}$$

$$(2) \quad d_j / \sum_{i=1}^n a_{ij}$$

$$(3) \quad (c_j + d_j) / \sum_{i=1}^n a_{ij}$$

Onde c_j representa o custo da coluna j considerando-se o primeiro objetivo, d_j representa o custo da coluna j considerando-se o segundo objetivo e o somatório representa a quantidade de linhas (enumeradas de 1 a n) do problema cobertas pela coluna j .

A Equação (1) dá preferência a colunas com custos baixos no primeiro objetivo que cobrem maior quantidade de linhas, a Eq. (2) faz o mesmo para o segundo objetivo, e a Eq. (3) faz uma combinação linear dos objetivos. Estas expressões são utilizadas em um esquema *round robin*: cada uma é usada durante um certo número de iterações. Este esquema permite explorar melhor o espaço de busca e evita ótimos locais, no caso de aglomeração da população em apenas uma região da fronteira Pareto-ótima.

Na Figura 2 é demonstrado o esquema *round robin*. Na parte inicial (delimitada pela região número 1), o primeiro objetivo é explorado, na parte central (delimitada pela região número 2), a combinação dos dois objetivos é explorada e na parte final (delimitada pela região número 3), o segundo objetivo é explorado. Os valores nos eixos representam os valores de cada objetivo, e são meramente ilustrativos.

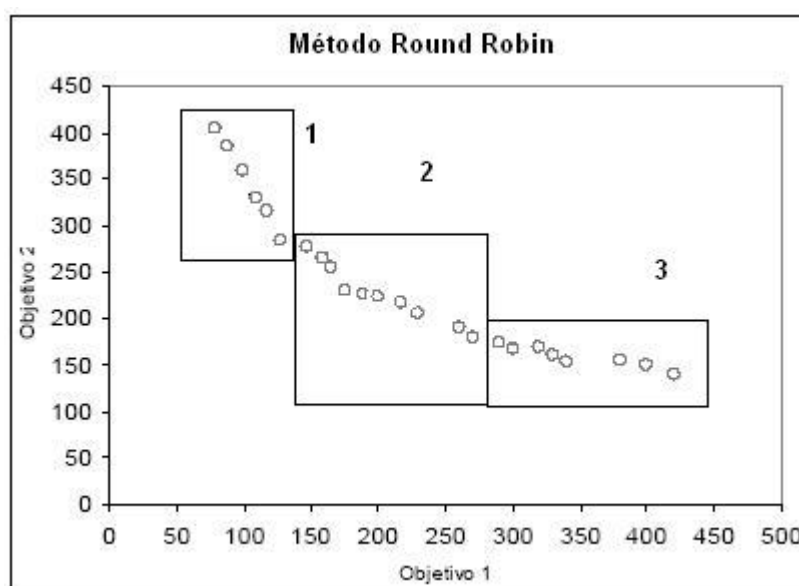


Figura 2. Exemplo de exploração do espaço de busca utilizando o esquema *round robin*.

4.1.5. Classificação em barreiras

Em versões multiobjetivo é comum se encontrar soluções melhores em certos objetivos e piores em outros. Formam-se então barreiras de soluções não dominantes entre si, mas que dominam outras barreiras. Esta classificação é utilizada para estabelecer um outro ranking das soluções (Srinivas e Deb, 1994) em barreiras, utilizado em conjunto com a função de avaliação para estabelecer a adaptação de cada cromossomo. A Figura 3 apresenta um exemplo desta classificação. Os valores nos eixos representam os valores de cada objetivo, e são meramente ilustrativos.

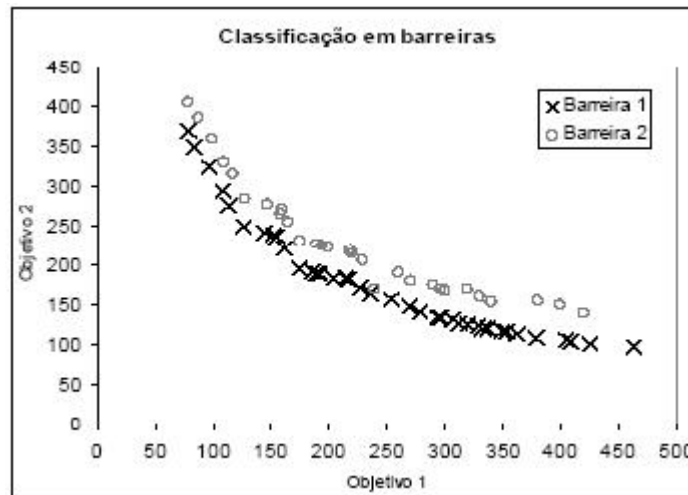


Figura 3. Exemplo de divisão da população em barreiras.

4.1.6. Processo de seleção

Uma “filtragem” na população corrente é realizada com objetivo de aproveitar os melhores cromossomos e permitir que estes tenham maior chance de se reproduzir e se manter através das iterações.

Para o processo de seleção é utilizado o método da roleta, o qual consiste em dois procedimentos, construir e girar a roleta. Para construir, distribui-se uma faixa da roleta para cada cromossomo com base em sua probabilidade cumulativa de seleção, calculada de acordo com a barreira em que o cromossomo se encontra. Depois se gira a roleta: gera-se um valor aleatório dentro da roleta e seleciona-se o cromossomo cuja faixa de probabilidade cumulativa de seleção contenha esse valor.

O método é repetido até que a população seja preenchida completamente. Cromossomos com alta probabilidade cumulativa podem e devem ser escolhidos mais de uma vez, aumentando assim suas chances de se perpetuarem. Outros cromossomos de menor probabilidade cumulativa e os operadores genéticos são encarregados de evitar ótimos locais e convergências prematuras.

4.1.7. Operadores genéticos

Operadores genéticos transformam a população corrente por meio de alterações no material genético que compõe os cromossomos. São aplicados à população de acordo com probabilidades ou mesmo aleatoriamente.

A implementação de *crossover* (ou cruzamento) utilizada combina o material genético de dois cromossomos selecionados anteriormente através do método de torneio binário, no qual são formadas duas filas com dois cromossomos selecionados aleatoriamente cada. O melhor cromossomo de cada fila é selecionado para participar do *crossover* de ponto, no qual um ponto aleatório dentro do cromossomo é selecionado. Cada cromossomo filho recebe o material genético de um dos pais até o ponto selecionado, e o material genético do outro pai deste ponto em diante.

Após a geração do par de filhos, estes são adicionados à população corrente, e poderão posteriormente substituir os cromossomos da população dominados por eles, ou também serem eliminados caso sejam dominados. No caso de não dominarem nenhum outro cromossomo e não serem dominados, os cromossomos são simplesmente adicionados à barreira que pertencerem. Este operador garante a herança de boas características nas próximas gerações.

O operador de mutação é simples, cada bit de cada cromossomo tem a mesma pequena probabilidade de ser submetido à mutação, que inverte o valor do bit submetido, alterando assim seu valor nas funções objetivo. A probabilidade de submissão dos genes ao operador de mutação é regulada por uma taxa previamente estabelecida. Este operador é responsável por introduzir e manter a diversidade genética da população.

4.1.8. Controle de populações

Ao fim de cada iteração (geração), a primeira barreira da população corrente é copiada para uma população externa que mantém as melhores soluções encontradas durante a execução. Caso algum cromossomo recém-chegado domine outro da população externa, haverá uma redefinição da barreira na população externa.

Após a aplicação dos operadores genéticos, avaliação da nova população e cópia da primeira barreira para a população externa, é realizado um corte na população interna com objetivo de conter o seu aumento, mantendo para a próxima iteração do algoritmo apenas as três primeiras barreiras ou 40% dos indivíduos da população corrente. Esse processo é também conhecido como dizimação (Michalewicz, 1996).

4.1.9. Parâmetros

Parâmetros são parte essencial de um algoritmo genético e têm forte influência sobre os resultados obtidos (Gudwin, Guerrero e Suárez, 1999). A cada nova entrada pode ser necessário um novo ajuste. Entretanto, após vários testes de ajuste para cada uma das entradas usadas neste trabalho, o mesmo valor dos parâmetros foi utilizado para todas, sendo eles: taxa de mutação: 10%; taxa de cruzamento: 80%; taxa de dizimação da população: 60%; número de gerações: 8000; quota do *round robin*: 100; tamanho inicial da população inicial: 100. Estes valores foram escolhidos por demonstrarem melhora significativa no comportamento do algoritmo genético, mantendo a diversidade da população e evitando a convergência prematura sem cair em ótimos locais. Outro fator levado em consideração ao comparar duas versões com parâmetros diferentes é a quantidade de valores ótimos alcançados e o número de gerações necessárias para alcançá-los.

5. Testes e resultados para seleção de colunas

São apresentados os resultados obtidos pela fase de seleção de colunas, bem como comparações entre versões dos algoritmos testados neste trabalho e comparações das versões finais com técnicas da literatura recente.

Todos os testes computacionais foram realizados em computadores com processadores AMD Duron 1.3 gigahertz, 128 megabytes de memória RAM, sistema operacional SuSE Linux e compilador GCC (*Gnu Compiler Collection*).

Foram realizados testes com instâncias encontradas na biblioteca de instâncias numéricas da MCDM (International Society on Multiple Criteria Decision Making, 2005), na seção *MultiObjective Set Covering Problem* (MOSCP).

5.1. Descrição das entradas

As entradas se dividem em conjuntos identificados por letras maiúsculas. A Tabela 1 descreve as características dos conjuntos.

Tabela 1 - Características dos conjuntos de entradas utilizadas.

Identificação/ Série	Linhas	Colunas	Densidade média da matriz de recobrimento	Objetivos
A	10 a 200	100 a 1000	19.20	aleatórios
B	10 a 200	100 a 1000	10.03	aleatórios e conflitantes
C	10 a 200	100 a 1000	10.06	Aleatórios, com padrões
D	10 a 200	100 a 1000	20.00	Aleatórios, com padrões e conflitantes

Na sub-seção seguinte são apresentados os resultados para instâncias das séries A e B, embora tenham sido testadas instâncias de todas as séries. Instâncias aleatórias e com objetivos conflitantes se encaixam perfeitamente no perfil do problema de alocação de tripulantes e por isso a seção seguinte se concentra nas mesmas.

5.2. Resultados

Os resultados da fase de geração de colunas são apresentados como conjuntos de pontos, onde cada ponto representa os valores de cada solução. Como se trata da versão bi-objetivo do problema de recobrimento de conjuntos, os resultados são apresentados em um gráfico de dispersão, onde o valor em cada eixo representa o valor de cada objetivo da solução.

Para a verificação de dominância, para cada ponto, traça-se uma reta imaginária paralela a cada eixo; os pontos que se encontrarem dentro da área imaginária do primeiro quadrante são considerados dominados pelo ponto localizado na origem das retas, pois possuem valores piores em ambos os objetivos. O problema aqui abordado é um problema de minimização, portanto quanto mais os pontos se aproximarem da origem do gráfico, melhor serão em comparação aos outros mais distantes.

5.3. Comparações

Foram realizados dois tipos de comparações: o primeiro compara versões do algoritmo utilizado neste trabalho, com objetivo de avaliar as principais melhorias implementadas, neste caso, o torneio binário e o esquema *round robin*; o segundo envolve sua comparação com os resultados de Jaskiewicz (2004), cujo trabalho descreve a implementação e comparação de dez metaheurísticas aplicadas ao problema de recobrimento de conjuntos bi-objetivo.

As métricas de comparação de qualidade dos conjuntos de pontos gerados utilizadas são: comparação direta ponto a ponto, onde se verifica a quantidade de pontos dominados e/ou acrescentados; e a medida de distância média do conjunto de pontos, proposta em Czyzak e Jaskiewicz (1998), onde o conjunto de referência é o conjunto gerado neste trabalho. A medida de distância assume o valor zero se em todos os objetivos o conjunto gerado Jaskiewicz (2004) alcança os valores do conjunto de referência, caso contrário, a medida assume o valor do máximo desvio de valores.

A Tabela 2 mostra o resultado da execução do algoritmo com e sem a utilização de torneio binário na fase de *crossover*, para algumas entradas.

Tabela 2 - Melhorias alcançadas utilizando Torneio Binário.

Arquivo	Número de pontos		Pontos dominados		Distância média entre as versões
	Sem torneio binário	Com torneio binário	Sem torneio binário	Com torneio binário	
2scp11A	38	36	4	0	0.00140
2scp82B	8	13	4	0	0.04993
2scp101A	11	12	9	0	0.16783

Em todos os casos a utilização de torneio binário gerou um conjunto melhor de pontos, sendo melhor quanto maior a entrada. Nota-se que para a entrada 2scp11A foram gerados menos pontos, porém de melhor qualidade, dominando 4 dos pontos gerados sem o torneio. A Figura 4 mostra os pontos gerados para a maior entrada, 2scp101A, tornando clara, visualmente, a melhoria causada por esta técnica. Os valores nos eixos representam os valores de cada objetivo, e são meramente ilustrativos.

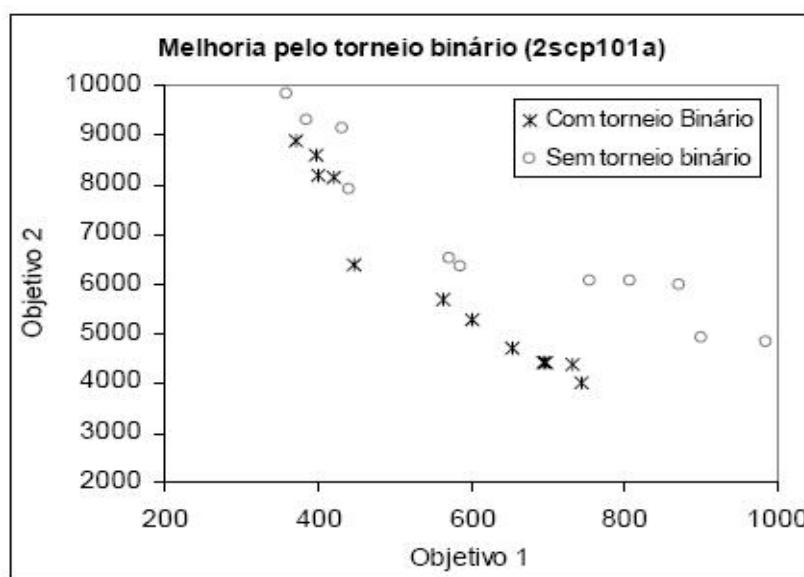


Figura 4. Melhoria alcançada utilizando o torneio binário em relação ao método aleatório.

A Tabela 3 mostra o resultado da execução do algoritmo com e sem a utilização do esquema *round robin* nas funções de reparo. Embora a utilização do *round robin* tenha deixado de gerar alguns pontos, o conjunto de pontos gerados é melhor, sempre dominando soluções geradas sem *round robin*. Na figura 5 são mostrados os pontos gerados para uma das entradas, os valores nos eixos representam os valores de cada objetivo, e são meramente ilustrativos. Pode ser notado na parte inferior do gráfico que dando prioridade algumas vezes a um objetivo, outras vezes a outro, conseguiu-se melhorar e expandir a fronteira de um dos objetivos, dominando 8 dos pontos encontrados sem *round robin*.

Tabela 3 - Melhorias alcançadas utilizando *round robin*.

Arquivo	Número de pontos		Pontos dominados		Distância média entre as versões
	Sem <i>round robin</i>	Com <i>round robin</i>	Sem <i>round robin</i>	Com <i>round robin</i>	
2scp11A	31	38	8	0	0.00322
2scp82B	16	13	7	1	0.01077
2scp101A	11	12	7	3	0.07491

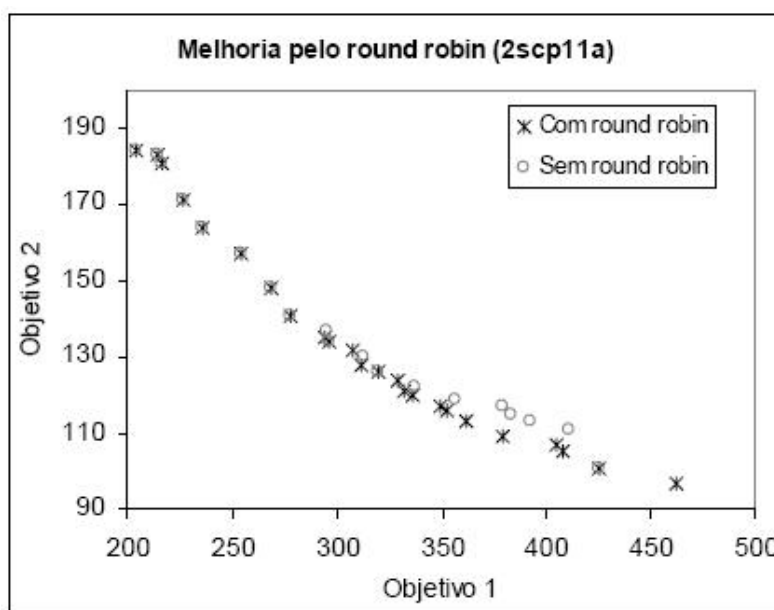


Figura 5. Melhoria alcançada utilizando o esquema *round robin* em relação ao método comum.

Comparou-se os resultados para três instâncias relatadas em (Jaszkiewicz, 2004) e disponibilizadas em sua página web. Em seu trabalho, Jaszkiewicz compara os métodos *Genetic Local Search*, *Simulated Annealing*, *Pareto Simulated Annealing*, *Nondominated Sorting Genetic Algorithm*, *Controlled Elitist Nondominated Sorting Genetic Algorithm*, *Strength Pareto Evolutionary Algorithm*, *Multiple Start Local Search with Random Weight Vectors*, e um novo método: *Pareto Memetic Algorithm*. Ele conclui que os melhores desempenhos são dos métodos *Multiple-Objective Genetic Local Search* (MOGLS) e *Pareto Memetic Algorithm* (PMA), sendo as duas versões do *Nondominated Sorting Genetic Algorithm* (NSGA) as que obtiveram pior desempenho.

O algoritmo genético desenvolvido neste trabalho foi testado e comparado individualmente com todas as técnicas utilizadas por Jaszkiewicz, mas são apresentadas somente as comparações com os resultados combinados dos métodos MOGLS e PMA, as de melhor desempenho em seu trabalho. Reuniram-se os resultados dos dois métodos descartando os pontos dominados, podendo ser vistos na Tab. 4.

Tabela 4 - Comparação com os resultados dos métodos MOGLS+PMA.

Arquivo	Comparação Direta	Distância Média
2scp82B	Todos os pontos dominados ou igualados	0,01451
2scp101A	25 pontos não foram igualados ou dominados	-0,00097
2scp102A	Todos os pontos dominados ou igualados	0,98544

Dentre as entradas testadas, a entrada 2scp101A teve um desempenho ligeiramente pior. Mas para as entradas de densidade maior e objetivos conflitantes, algoritmo genético implementado encontra resultados melhores. Pode ser

observado que para a entrada 2scp102A a distância média entre as barreiras foi quase 1,0 pela métrica utilizada. Isso pode ser visualizado na Fig. 6, onde se nota que na parte inicial do gráfico (preferência do primeiro objetivo), o algoritmo aqui proposto não apenas gerou mais pontos como também gerou pontos de melhor qualidade. A Figura 7 apresenta uma ampliação da parte inicial das barreiras da figura anterior. Em ambas as figuras, os valores nos eixos representam os valores de cada objetivo, e são meramente ilustrativos

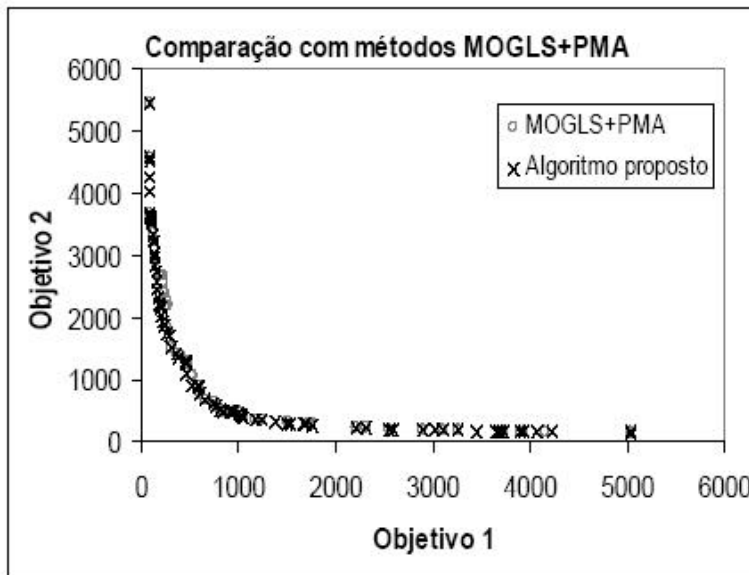


Figura 6 - Comparação entre as heurísticas para a entrada 2scp102A, mostrando todos os pontos gerados.

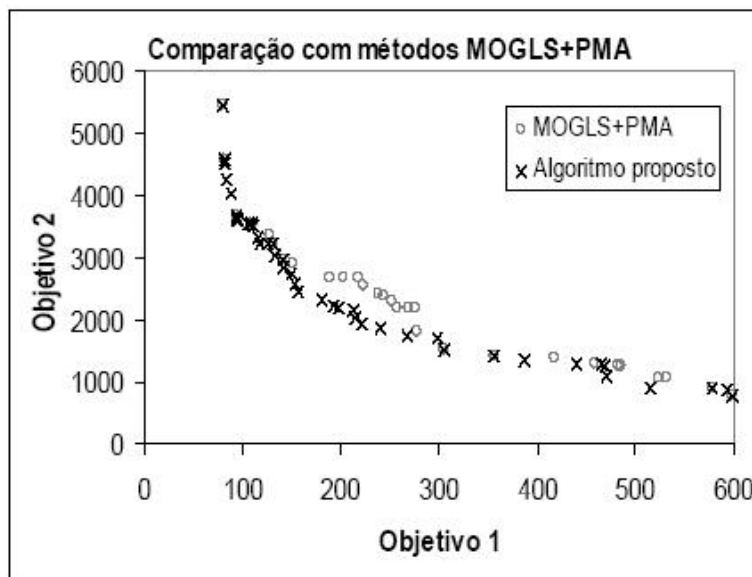


Figura 7 - Comparação entre as heurísticas para a entrada 2scp102A, mostrando apenas a parte inicial das barreiras.

7. Conclusões

Apresentaram-se alterações no algoritmo genético clássico aplicando-o ao problema de recobrimento multiobjetivo, a fim de usá-lo em problemas de alocação de tripulações, embora possa ser utilizado em vários outros problemas onde a seleção de colunas surge como um subproblema.

O desempenho do algoritmo também foi testado com outras heurísticas e resultados recentes da literatura, mostrando um desempenho superior para algumas instâncias. A incorporação de torneio binário e um esquema de reparo dos cromossomos utilizando uma técnica *round robin* de preferência de objetivos trouxe significantes melhorias ao algoritmo, tornando o NSGA um algoritmo competitivo.

Atualmente está sendo incorporada a fase de geração de colunas, uma etapa anterior ao subproblema aqui apresentado, que deverá alimentar o SCP com colunas viáveis, segundo as várias restrições do problema de alocação de tripulações.

8. Agradecimentos

Agradecemos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq, pelo apoio financeiro cedido ao desenvolvimento deste trabalho.

9. Referências

- Balas, E., Ceria, S., Cornuejols, G., Natraj, N., 1996, "Gomory cuts revisited", *Operations Research Letters* 19, 1—9
- Beasley, J. E., 1990, "A Lagrangean Heuristic for Set-Covering Problems", *Naval Research Logistics*, 37:151-164.
- Beasley, J. E., Chu, P. C., 1996, "A genetic algorithm for the set covering problem", *European Journal of Operational Research*, vol.94, 1996, pp392-404.
- Beasley, J. E. and Jornsten, K., 1992 "Enhancing an algorithm for set covering problems", *European Journal of Operational Research*, 58:293-300.
- Bixby, R., Gregory, J., Lustig, I., Marsten, R. E Shanno, D., 1992, "Very large scale linear programming: a case study in combining interior point and simplex methods" *Operations research*, 40. 885-897.
- Cabral, L. A. F., Freitas, M. J., Maculan, N. and Pontes, R. C. V., 2000, "An Heuristic Approach for Large Scale Crew Scheduling Problems at Rio-Sul Airlines", 40th International Symposium of the AGIFORS.
- Caprara, A., Fischetti, M., Toth, P., Vigo, D., Guida, P. L., 1998, "Algorithms for railway crew management", *Mathematical Programming* 79, 125-141.
- Caprara, A., Fischetti M., Toth, P., Guida, P. L., 1999, "Solution of Large-Scale railway Crew Planning Problems: the Italian Experience", N.H.M. Wilson (ed.) *Computer-Aided Transit Scheduling*, Lecture Notes in Economics and Mathematical Systems 471, Springer-Verlag, 1-18.
- Chankong, V. and Haimes, Y. Y., 1983, "Multiobjective Decision Making: Theory and Methodology", North-Holland (Elsevier), New York.
- Czyzak, P. and Jaszkievicz, A., 1998, "Pareto Simulated Annealing - A Metaheuristic Technique for Multiple Objective Combinatorial Optimization", *Journal of Multicriteria Decision Analysis*, 7 34-47.
- Desrosiers, J., Dumas, Y., Solomon, M. M., Soumis, F., 1995 "Time Constrained Routing and Scheduling, Network Routing," M.O. BALL et al. (eds), *Handbooks in Operations Research and Management Science* 8, 35-139.
- Ehrgott, M. and Ryan, D. M., 2003, "Constructing robust crew schedules with bicriteria optimization", Technical report, University of Auckland, To appear in *Journal of Multi-Criteria Decision Analysis*.
- Farley, A. A., 1990, "A note on bounding a class for linear programming problems, including stock problems" , *Operations Research*, 38 (5), 922-924.
- Fores, S., Proll, L. and Wren, A., 1996, "A column generation approach to bus driver scheduling", Bell, M H G (editor), *Transportation Networks: Recent Methodological Advances*, pp.195-208, Pergamon.
- Gamache, M., Soumis, F., Marquis, G., 1999, "A column generation approach for large-scale aircrew rostering problems", *Operations Research*, vol. 47.
- Gopalakrishnan, B., Johnson, E. Lee, E. e Pritchett, A., 2001, "A subproblem approach for solving the airline crew pairing problem", *INFORMS fall*, Miami Beach.
- Gudwin, R. R., Guerrero, J. A. S. and Suárez, L. L., 1999, "Análise da Importância de Parâmetros em um Algoritmo Genético por meio de sua Aplicação no aprendizado de uma Rede Neural", *Anais do XIX Congresso da SBC. II ENIA*.
- Hu, J., Johnson, E. L., 1999, "Computational results with a primal-dual subproblem simplex method. *Operations Research Letters*, 25, 149-157.
- International Society on Multiple Criteria Decision Making, 2005, <http://www.univvalenciennes.fr/ROAD/MCDM/ListMOSCP.html>.
- Jacobs, L. W. and Brusco, M. J., 1993, "A simulated annealing-based heuristic for the set-covering problem", working paper, Operations Management and Information Systems Department, Northern Illinois University, Dekalb, IL 60115, USA.
- Jaszkievicz, A., 2004, "A Comparative Study of Multiple-Objective Metaheuristics on the Bi-Objective Set Covering Problem and the Pareto Memetic Algorithm", *Annals of Operations Research* 131, 135-138. Kluwer Academic Publishers.
- Klabjan, D., A.J. Schaefer, E.L. Johnson, A.J. Kleywegt, and G.L. Nemhauser, 2001, "Robust Airline Crew Scheduling," *Proceedings of TRISTAN IV (refereed)*, pages 275-280.
- Kohl, N., 1995, "Exact methods for time constrained routing and related scheduling problems", PhD Thesis, Department of Mathematical Modeling, Technical University of Denmark, DK-2800 Lyngby, Dinamarca, xviii + 234p.

- Kohl, N. and Karish, S. E., 2004, "Airline Crew Rostering: Problem Types, Modeling and Optimization", *Annals of Operations Research*, 127, 223-257.
- Michalewicz, Z, 1996, "Genetic Algorithms + Data Structures = Evolution Programs", 3rd edition, Springer-Verlag, Berlin.
- Makri, A. and Klabjan, D., 2001, "Efficient column generation techniques for airline crew scheduling", Technical report, University of Illinois at Urbana-Champaign.
- Moudani, W., Consenza, C. A. N., Coligny, M. and Mora-Camino, F., 2001, "A Bi- Criterion Approach for the Airlines Crew Rostering Problem", *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, 486-500.
- Pereira, M. A., 2002, "Estratégias de seleção para o método de geração de colunas", Tese de Doutorado, INPE.
- ReVelle, C.S. and Swain, R. (1970) "Central Facilities Location" *Geographical Analysis* 2 30-42.
- Schaefer, A.J., 2000, "Airline Crew Scheduling under Uncertainty", Ph.D. dissertation, Georgia Institute of Technology.
- Souza, M. J. F., Cardoso, L. X. T., and Silva, G. P., 2003, "Programação de Tripulações de Ônibus: Uma Abordagem Heurística", XXXV Congresso da Sociedade Brasileira de Pesquisa Operacional SBPO.
- Srinivas, N. and Deb, K., 1994, "Multiple Objective Optimization using Nondominated Sorting in Genetic Algorithms", *Evolutionary Computation* 2(2). 221-248.