Uma Matheurística Baseada em PSO para o Problema de Custo de Disponibilidade de Recursos com Múltiplos Modos

Lettiery DLamare Portela Procópio
Universidade Federal do Rio Grande do Norte - UFRN
Campus Universitário Lagoa Nova, 59078-970, Natal, RN
lettiery@ppgsc.ufrn.br

Lucídio dos Anjos Formiga Cabral
Universidade Federal da Paraíba - UFPB
Rua dos Escoteiros, 58058-600, João Pessoa, PB
lucidio@ci.ufpb.br

Marco Cesar Goldbarg
Universidade Federal do Rio Grande do Norte - UFRN
Campus Universitário Lagoa Nova, 59078-970, Natal, RN
gold@dimap.ufrn.br

RESUMO

Este artigo descreve a construção de uma Matheurística baseada na Otimização por Enxame de Partículas (PSO) afim de solucionar o Problema de Custo de Disponibilidade de Recursos com Múltiplos Modos. Inspirado na necessidade de balancear a utilização de recursos renováveis com o tempo total (makespan), através do escalonamento das atividades com seus diversos modos de execução presente no projeto. Testes realizados em instâncias da literatura comprovam a eficácia da utilização da programação matemática adaptada com a Meta-heurística PSO.

PALAVRAS CHAVE. Disponibilidade de Recursos, PSO, Matheurística.

ABSTRACT

This paper discribes the construction of Matheuristic based on Particle Swarm Optimization in order to solve the Availability of Cost Problem Resources with Multi-Modes. Inspired by the need to balance the use of renawable resources with the total time (makespan), by scheduling the activities with its various implementations executions modes present in the project. Tests on literature instances demonstrate the effectiveness in te use of mathematical programming adapted to the Particle Swarm Optimization.

KEYWORDS. Resource Availability, PSO, Matheuristic.

1. Introdução

O processo de tomada de decisão é uma etapa crítica para diversos tipos de projetos reais e já é pressuposta por alguns autores como uma ciência própria, a Ciência da Gestão (*Management Science*) Taylor [2004]; Calazans [2012]. Certamente, o ato de refletir e planejar os passos de um determinado projeto impacta fortemente na sua execução, destinando-o para o sucesso ou fracasso em casos de omissão desta etapa. Métodos científicos apresentam-se como ferramentas para avaliar as possíveis alternativas.

Por isso, o planejamento de projetos necessita da coordenação de atividades relacionadas para atingir objetivos pré-definidos. Entre os objetivos mais comuns estão o tempo de conclusão e o custo total do projeto. Um recente exemplo de aplicação pode ser encontrado no trabalho de [Yamashita e Morabito, 2012], o qual fala sobre o Problema de Custos de Disponibilidade de Recursos com Múltiplos Modos (PCDRMM), que tem o objetivo de otimizar o custo da alocação de recursos nas atividades considerando o tempo final do projeto.

Problemas como o PCDRMM permite ganhos concretos nos negócios empresariais através da otimização de processos. Suas soluções são medidas por duas grandezas distintas: a quantidade de recursos que serão utilizados e o tempo final do projeto. Métodos exatos se apresentam como uma abordagem naturalmente desejada, onde soluções ótimas forneceriam cenários atrativos para a indústria, mas a necessidade de minimização do tempo total do projeto, e a alocação das atividades com diversos modos distintos, caracterizam um problema de otimização combinatória pertencente a classe *NP-Hard* (como desmostrado por Möhring [1984]; Blazewicz et al. [1983]). Essas propriedades nos remetem a escolha de métodos heurísticos ou heurísticos com hibridização exata, para permitir menor tempo de processamento computacional e resultados de boa qualidade.

A aplicação desse problema pode ser encontrada em vários cenários reais como na construção civil onde máquinas, matéria prima e equipes de trabalhadores são recursos escassos e o tempo final do projeto necessita ser otimizado. Assim como o compartilhamento de recursos na computação em nuvem, a produção industrial e até projetos de desenvolvimento de softwares podem ser solucionados utilizando o PCDRMM.

Assim, este trabalho é motivado pela busca de soluções de boa qualidade para o Problema de Custo de Disponibilidade de Recursos com Múltiplos Modos através de exemplos da literatura que obtiveram sucesso com heurísticas para resolver problemas onde a utilização de recursos é cara para o projeto como um todo ou que ainda existe uma escassez dos recursos utilizados.

Este trabalho é dividido em cinco seções: (1) Introdução e explicações inicias sobre o problema, (2) Descrição e características do problema, (3) Trabalhos Relacionados e revisão da literatura do PCDRMM, (4) Descrição da Solução proposta, (5) Resultados e Conclusões do trabalho.

2. Descrição do Problema

O PCDRMM pode ser definido como um problema que balanceia a minimização da utilização dos recursos renováveis e o tempo de duração do projeto, contudo, suas atividades podem assumir diversos modos de execução que variam o tempo e a quantidade de recursos utilizados. Elas ainda possuem uma relação de precedência temporal que determina o tempo mínimo de início.

O custo de utilização dos recursos é contabilizado por todo o horizonte do projeto, e os recursos podem ser reutilizados por outras atividades sem adicionar quaisquer gastos à função objetivo. Este problema equilibra a utilização dos recursos de forma a não penalizar o tempo final do projeto, por isso sua aplicabilidade a projetos reais se apresenta de forma imediata ou de fácil adaptação.

Ressalte-se que a principal tarefa desse problema é organizar as atividades a fim de minimizar o tempo final do projeto conhecido como *makespan*, e o custo atribuído à disponibilização dos recursos para as atividades. Uma das primeiras características que torna o problema mais aplicável a projetos reais é a possibilidade de uma atividade ser realizada de várias maneiras distintas.

Um exemplo de dados de entrada para o problema podem ser observados na Figura 1 e na Tabela 1. Para mapear a precedência das atividades descritas na Figura 1, temos um conjunto $H = \{(s,1),(s,2),(s,4),(1,3),(2,3),(3,t),(4,t)\}$, no qual a existência do par ordenado $(1,3)$ implica que a atividade 1 precede a atividade 3. Vale ressaltar que as atividades s e t são virtuais para demarcam o início e o fim do projeto.

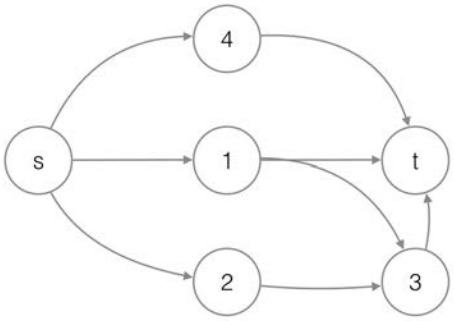

Figure 1: Exemplos de precedências

As variáveis d_{ji} que demarcam a duração da atividade j no modo i, e r_{jik} que quantifica a necessidade da atividade j no modo i pelo recurso do tipo k, são demonstradas na Tabela 1.

Atividade j		**Modo de execução** i	**Duração** d_{ji}	**Recurso 1** r_{ji1}	**Recurso 2** r_{ji2}	**Recurso 3** r_{ji3}
s	M_s **= 1**	1	0	0	0	0
1	M_1 **= 1**	1	3	2	2	1
2	M_2 **= 1**	1	4	1	2	1
3	M_3 **= 2**	1	2	2	2	3
		2	5	1	0	1
4	M_4 **= 1**	1	2	0	2	0
t	M_t **= 1**	1	0	0	0	0

Table 1: Exemplo de dados de entrada

A Tabela 1 exemplifica os valores referentes às atividades j que executam nos possíveis modos i, e duram d_{ji} unidades de tempo, também requerem $r_{ji1\ldots3}$ quantidades de recursos. A propriedade de Múltiplos Modos torna-se visível na atividade 3 $(j = 3)$ que pode executar em dois modos diferentes. O primeiro modo $(i = 1)$ gastará 2 unidades de tempo, 2 recursos do tipo 1 e 2, e 3 recursos do tipo 3, no seu segundo modo $(i = 2)$ necessitará de 5 unidades temporárias, 1 recurso do tipo 1 e 3, mas não necessitará de recursos do tipo 2. A Tabela 1 também demonstra a não influência das atividades virtuais s e t para o projeto, pois, não necessitam de unidade de tempo nem quantidade de recurso.

O custo dessa instância do problema pode ser dado pela função $C_k(a_k) = c_1a_1 + c_2a_2 + c_3a_3$, onde c_1, c_2 e c_3 são os custos de utilizações dos recursos do tipo $k = \{1,2,3\}$, e para esse exemplo assumem os valores de 2, 1 e 3 respectivamente. Já a quantidade $a_k = \{a_1, a_2, a_2\}$ representa o total de recursos do tipo k utilizados ao longo do projeto.

Ao utilizar da atividade 1 $(j = 1)$ é acrescentado ao custo do projeto o valor da função $C_k(a_k) = 2 * 2 + 1 * 2 + 3 * 1$, ou seja, $C_k(a_k) = 9$. Esse custo só será acrescentado totalmente ao projeto se ao executar a atividade 1 ainda não tiver sido disponibilizado nenhum recurso para o projeto, ou se os recursos estiverem sendo utilizados por outra atividade, caso contrário, eles serão reutilizados pela atividade 1 e acrescentadas somente as novas disponibilizações dos novos recursos alocados.

Uma solução viável para esse problema é a definição do tempo de início de todas as atividades. Porém, uma boa solução almeja a minimização tanto do *makespan* quanto da quantidade de recursos utilizados no projeto. É possível ver na Figura 2 uma solução de boa qualidade para os dados citados acima.

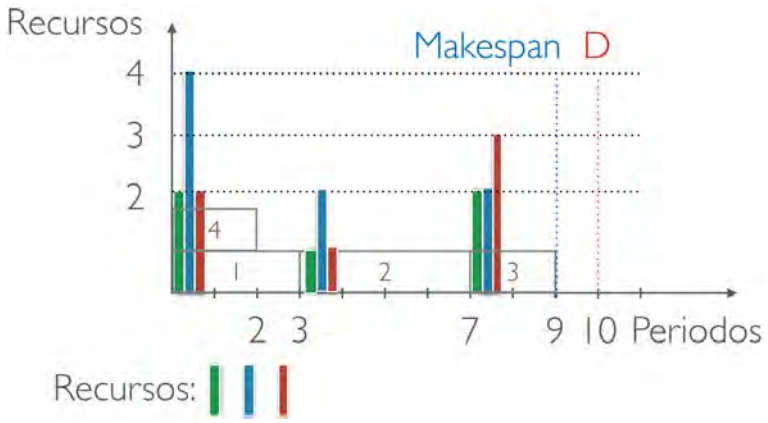

Figure 2: Exemplos de solução

Na Figura 2 a disponibilidade dos recursos é demarcada pelas barras verdes, azuis e vermelhas que diferem respectivamente os tipos 1, 2 e 3. As quatro atividades reais são mostradas com retângulos enumerados segundo a Tabela 1, assim, podemos ver no eixo horizontal do gráfico a marcação temporária do projeto e no eixo vertical são enumerados as disponibilidades dos recursos. A linha pontilhada vermelha demarca a data $D = 10$ de entrega do projeto e a linha pontilhada azul refere-se ao *makespan* $= 9$ da solução.

O custo final de utilização dos recursos são demarcados pelos maiores valores que os 3 tipos de recursos assumiram: 2, 4 e 3 respectivamente, onde os recursos verde e azul (tipo 1 e tipo 2) assumiram os seus maiores valores no primeiro período com a solicitação das atividades 1 e 4, já o recurso 3 assume seu maior valor para atender a atividade 3 que executa no seu modo 1 ($i = 1$). Para contabilizar o custo total do projeto utilizamos a função $C_k(a_k) = 2a_1 + 1a_2 + 3a_3$, onde os valores que multiplicam a_k são os pesos dos respectivos tipos de recursos, $C_k(a_k) = 2{*}2{+}1{*}4{+}3{*}3 = 17$.

A formulação matemática proposta por Yamashita e Morabito [2012] para o PCDRMM considera o problema como um projeto com n atividades, das quais a primeira e a última atividade (1 e n) são equivalentes às atividades virtuais s e t , representando o início e o fim do projeto. O par ordenado (h, j), quando existe no conjunto H, representa uma relação de precedência da atividade h com j, ou seja, se $(h, j) \in H$, então h precede j. M_j é a quantidade de modos de execução da atividade j. Quando j é executada no modo i requer d_{ji} unidades de tempo para finalizar e r_{jik} quantidades do recurso do tipo $k(k = 1, \ldots, m)$. O custo da disponibilidade da quantidade de recurso a_k do tipo k é dado por uma função linear não decrescente $C_k(a_k)$, é adotado que a variável a_k assume valor constante em relação ao tempo. A data final de entrega do projeto é dado por D. Yamashita, baseada em Talbot [1982], apresenta um modelo com as seguintes variáveis de decisão:

- $x_{jit} = \begin{cases} 1, & \text{se a atividade } j = 1, ..., n \text{ é executada no modo } i = 1, ..., M_j \text{ e termina} \\ & \text{no instante } t = 1, ..., D. \\ 0, & \text{caso contrário.} \end{cases}$
- a_k = quantidade de recursos do tipo k disponível ao longo do projeto, $k = 1, \ldots, m$.

Os dados de entrada do problema são:

- $C_k(a_k)$ = função de custo associada à disponibilidade a_k do recurso do tipo k.

- D = data de entrega do projeto.
- H = conjunto de relações de precedência.
- d_{ji} = duração da atividade j quando executada no modo i.
- r_{jik} = quantidade de recursos do tipo k que a atividade j utiliza ao ser executada no modo i.
- LF_j = instante de término mais tarde que a atividade $j \leq D$ pode ser finalizada, considerando as relações de precedência entre as atividades. Esse valor é obtido pelo alocação regressiva das atividades, iniciando da atividade t alocada no instante D.
- EF_j = instante de término mais cedo que a atividade j pode ser completada, também considera as relações de precedência entre as atividades. O EF_j é obtido pela alocação progressiva das atividades, tendo inicio na atividade s.

objetivo:

$$Minimizar \sum_{k=1}^{m} C_k(a_k) \tag{1}$$

sujeito a:

$$\sum_{i=1}^{M_j} \sum_{t=EF_j}^{LF_j} x_{jit} = 1,\ j = 1, ..., n \tag{2}$$

$$\sum_{i=1}^{M_h} \sum_{t=EF_h}^{LF_h} t \cdot x_{hit} \leq \sum_{i=1}^{M_j} \sum_{t=EF_j}^{LF_j} (t - d_{ji}) \cdot x_{jit}\ j = 1, ..., n,\ \forall h \mid (h, j) \in H \tag{3}$$

$$\sum_{j=1}^{n} \sum_{i=1}^{M_j} \sum_{b=1}^{t+d_{ij}-1} r_{jik} x_{jib} \leq a_k\ k = 1, ..., m\ t = 0, \ldots, D \tag{4}$$

$$x_{jit} \in \{0, 1\}\ \ j = 1, ..., n\ t = 1, ..., D\ i = 1, \ldots, M_j \tag{5}$$

$$a_k \geq 0,\ a_k \in Z\ k = 1, ..., m \tag{6}$$

Na formulação anterior, Yamashita e Morabito [2012] descreve o problema com a função objetivo (1) que minimiza o custo pela disponibilização dos recursos através da função $C_k(a_k)$ que quantifica o custo de utilização da quantidade a_k do recurso do tipo k. A restrição (2) garante que toda atividade só será executada exclusivamente uma vez em um único modo dentro da sua janela de tempo (LF_j, EF_j) permitida. A precedência das atividades é respeitada pela restrição (3). (4) resguarda que durante o tempo de execução das atividades, a quantidade a_k do recurso do tipo k não será excedida e garante que os recursos são do tipo renováveis. As restrições (5) e (6) definem a variável de decisão binária x_{jit} e que a quantidade a_k seja inteira e positiva.

Para entender a complexidade da resolução do problema por métodos exatos, podemos calcular o número de soluções possíveis em uma instâncias simples com $j = 10$ atividades, onde cada j pode ser executada em $M_j = 3$ modos distintos, totalizando 10! x $3^{10} \approx 2,14$ x 10^{11} soluções (viáveis e inviáveis).

3. Trabalhos Relacionados

Os trabalhos da literatura que abordam o problema oferecem várias soluções como resposta para o problema, pois é atribuída a um decisor (gestor do projeto) a responsabilidade da escolha que respeitam a data final de entrega do projeto. Uma avaliação entre o custo e o tempo é feita para escolher entre perder em tempo e ganhar nos custos do projeto, ou adiantar o prazo e acrescentar um custo. Essa relação de "perde-ganha" é denominada *trade-off* e está muito presente na logística.

Chen et al. [2012] afirma que o trabalho pioneiro abordando o PCDRMM foi o de Hsu e Kim [2005]. Hsu e Kim [2005] tratam de uma adaptação do Problema de Custo de Disponibilidade de Recursos (PCDR) acrescentando a característica dos múltiplos modos nas atividades, e nomeada de *Multi-Mode Resource Investment Problem (MMRIP)*.

Além disso, é proposta uma regra de decisão que considera simultaneamente restrições de data de vencimento (*due date*). O uso de recursos foi proposto para selecionar e agendar tarefas. O presente trabalho também utiliza do *trade-off* como ponto de parada do algoritmo.

Chen et al. [2012] ainda propôs um método heurístico de enumeração baseada em árvore de precedência que seleciona a ordem em que as atividades serão executadas, logo em seguida são realizados três passos para alcançar os objetivos: a seleção das atividades; atribuição de seus modos; e a decisão de seus tempos de início. A motivação apresentada é o aluguel de recursos computacionais na nuvem para processamento, uma vez que essa alocação é paga apenas uma vez, caracterizando os recursos renováveis.

O terceiro trabalho Yamashita e Morabito [2012] foi o primeiro a nomear o *Problema de Custo de Disponibilidade de Recursos com Múltiplos Modos*, e a propor um algoritmo exato que utiliza *Brach-and-Bound* para solucionar e comparar seus resultados com implementação da formulação matemática através do solver *CPLEX* da *IBM*. Sua comparação mostrou que o algoritmo proposto supera pelo tempo computacional a implementação do modelo.

Qi et al. [2015] utilizam uma heurística de Otimização por Enxame de Partículas (*Particle Swarm Optimization*) para comparar seus resultados com soluções exatas implementada no software Lingo, porém seus testes mostram que a adição de uma etapa de busca local baseada em *Scatter Search* retarda seu algoritmo.

Por fim, uma adaptação do PCDRMM é encontrada em Nadjafi [2014] com o nome de *Multi-mode Resource Availability Cost Problem with Recruitment and Release dates for resources* (MRACP-RR), onde o custo de disponibilização dos recursos é contabilizado apenas pelo período entre a primeira utilização do recurso e a última. É utilizado a heurística *Simulated Annealing* para construção de suas soluções. Como não existe registro anterior desse problema adaptado, são realizados testes computacionais com o modelo matemático, e seu algoritmo demostra ser eficaz para instâncias com muitas atividades.

Nestes trabalhos destacam-se caracteristicas marcantes como Yamashita e Morabito [2012], que apresenta-se como o único a propor uma heurística que garante soluções ótimas. A pesquisa desenvolvida em Chen et al. [2012] relata bons resultados em instâncias distintas, porém com um conjunto limite de até 20 atividades, enquanto Nadjafi [2014] mostrou a qualidade do seu algorítmo na qualidade da soluções geradas e no tempo de processamento em instância de até 90 atividades, além de ser uma adaptação que contabilizar o custo de forma diferente. É destacável também em Nadjafi [2014] a preocupação de liberar o prazo de entrega do projeto para uma decisão com mais abrangências nos valores de tempo e custo (decisão de *trade-off*).

4. Particle Swarm Optimization com a Matheurística

Motivado pelos promissores resultados em [Qi et al., 2015], a Meta-heurística *Particle Swarm Optimization* (PSO) apresentada por [Kennedy e Eberhart, 1995] com base no comportamento de pássaros e outros animais que andam em grupo, foi possível mapear um padrão para a movimentação desses indivíduos em busca do objetivo do bando.

No PSO cada indivíduo (representado por uma solução) de uma população (enxame), quantifica sua experiência local (vizinhos locais), e ideais do enxame (melhores globais), assim, dar-se o aprendizado individual de cada partícula de um enxame. O fluxograma da Figura 3 resume a adaptação dos passos para o processo evolucionário do algoritmo proposto.

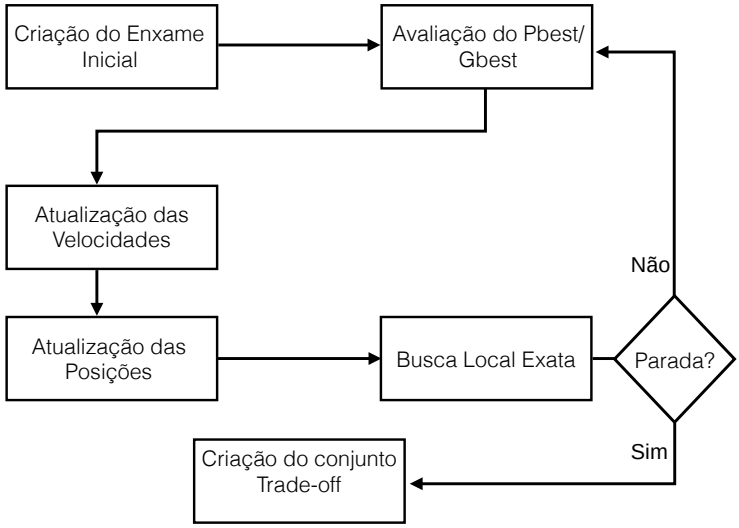

Figure 3: Fluxograma do PSO proposto

A criação do enxame inicial influencia diretamente no número de interações necessárias para alcançar o objetivo do algoritmo e na diversificação das regiões do espaço de busca. Obter enxames iniciais que permitam uma boa eficiência na busca e implicitamente representem regiões de busca diversificadas são características almejadas por três algoritmos de criação de soluções iniciais:

- Melhor *makespan*: são selecionados os modos das atividades com a menor duração do tempo de execução, desconsiderando a utilização dos recursos. A escolha do tempo de início das atividades é exatamente o instante de término da sua última predecessora $EF_j - min(d_{ji})$.
- Modos aleatório: onde os modos das atividades são selecionados de maneira aleatória e são alocadas de acordo com sua ordem de prioridade (precedência).
- Melhor modo por custo: que tem como a escolha do modo m_{ij} da atividade j é realizada verificando o custo de utilização dos recursos do modo i, isso possibilita uma tentativa de redução do custo do projeto.

Uma partícula do PSO é um escalonamento que respeita todas as restrições da Formulação Matemática, o qual pode ser avaliado tanto pelo seu tempo *makespan*, quanto pelo seu custo de utilização dos recursos $C_k(a_k)$. A minimização desses valores são as metas a serem alcançadas, porém, a função objetivo da modelagem matemática do problema, tenta minimizar apenas o custo, deixando o tempo como uma restrição complementar do problema.

Para melhor quantificar a localização das partículas no algoritmo, foi inserido o valor do *makespan* na função de avaliação da solução $f(s) = p_t * t + p_c * c$, onde p_t é a proporção do tempo final da solução t, e p_c é a proporção do custo por disponibilidade c. A utilização de $p_t = p_c$ balanceia a necessidade do custo e o tempo da solução e serve também de parâmetros de ajuste do algoritmo de acordo com a necessidade do decisor. Para melhorar a avaliação dos indivíduos os valores de t e c são normalizados, uma vez que são grandezas distintas. A função de avaliação ainda atribui uma penalidade à solução que não cumprir o requisito da data final do projeto D.

A melhor experiência conhecida de uma partícula é seu $pbest$, e o melhor escalonamento conhecido de todo o enxame (melhor global) é valor de referencia do $gbest$. Assim a fase de avaliação do $gbest$ e $pbest$, quantifica os escalonamentos para verificar se foi gerado um novo $gbest$ ou $pbest$ na iteração atual.

A representação da solução do PCDRMM no PSO é baseada em [Qi et al., 2015], que mapeia uma partícula em dois vetores: $t_j = (t_1, t_2, ..., t_n)$, demarca o tempo de inicio t_j de cada atividade j, e $m_j = (m_1, m_2, ..., m_n)$, enumera o modo m_j de todas as n atividades. Essas duas informações necessárias para representar um escalonamento são concatenados em apenas um vetor $x = t_j \cup m_j$, com $x_i = (q_1, q_2, ..., q_n, q_{n+1}, q_{n+2}, ..., q_{2n})$, onde os primeiros n números $(q_1, q_2, ..., q_n)$ representam os tempos de início das atividades e, logo em seguida $(q_{n+1}, q_{n+2}, ..., q_{2n})$ os modos são apresentados. Como uma solução do PCDRMM não pode ser representada no domínio contínuo e o PSO gera soluções com essas características, adotamos o maior número inteiro $\lceil x \rceil$ para as variáveis contínuas.

As posições das partículas e velocidades calculadas com base na própria experiência e pela observação da sua vizinhança. Elas se movem no espaço de busca de acordo com as equações a seguir:

$$v_{i(e+1)} = w.v_{ie} + c_1.r_1(pbest_{ie} - x_{ie}) + c_2.r_2(gbest_e - x_{ie}) \tag{7}$$

$$x_{i(e+1)} = x_{ie} + v_{i(e+1)} \tag{8}$$

$$e = 1, ..., E \tag{9}$$

$$i = 1, ..., m \tag{10}$$

A equação (7) representa a atualização da velocidade para a nova época $(e + 1)$, onde temos v_{ie} como a velocidade da partícula i na época e, w representa a inércia da direção que estava sendo seguida, r_1 e $r_2 \in [0, 1]$ são constantes aleatórias, c_1 e $c2$ são os coeficientes que representam a aceleração cognitiva que levam os indivíduos em direção ao $pbest$ e ao $gbest$ respectivamente. A equação (8) atualizar a posição do indivíduo i na época x_{ie}, essas operações são realizadas por todas as m partículas do enxame A.

O pseudo-código descrito a seguir esclarece os passos do Algoritmo em Enxame de Partículas proposto para a solução do problema:

Algoritmo 1: Pseudo-código do PSO

Dados: Conjunto de partículas: A, Melhores locais: $pbest$, Melhor global: $gbest$, Épocas: E
início
  $A \leftarrow IniciaEnxameDeParticulas()$
  para todo $e \leftarrow 1$ *até* E **faça**
    $Atualizar(pbest, gbest, A)$
    para todo $a \in A$ **faça**
      $AtualizarVelocidades(a)$
      $a \leftarrow MoverParticula(a, e)$
    fim
    $B \leftarrow Selecao(A)$
    $BuscaLocalExata(B)$
    $Atualizar(pbest, gbest, B)$
  fim
  $CriacaoTradeOff(pbest)$
fim

No algoritmo existe uma etapa de busca local exata no processo, onde são selecionadas 10% das soluções da população para serem alteradas, sendo que 5% são constituídos pelos melhores indivíduos conhecidos, e os outros 5% são eleitos de forma aleatória, esse processo apenas modifica a solução se for possível a minimização nos valores de tempo ou custo.

Essa busca local é feita utilizando o modelo de programação matemática de Yamashita e Morabito [2012] adaptado como descrito nas equações (11), (12) e (13) e implementado pelo *framework CPLEX* da *IBM*. São aplicados algumas alterações no modelo original para possibilitar a otimização de apenas parte da solução em um tempo computacional aceitável.

A adaptação feita nessa etapa adiciona ao modelo original do PCDRMM o sub-conjunto de atividades fixas F que não serão alteradas pelo modelo, o qual é formado pelas atividades de um escalonamento viável $F = \{\forall j \in s' \mid t'_j - d'_{ji} \notin a_{kt}\}$, onde t'_j é o tempo em que a atividade j foi escalonada, d'_{ji} é a duração da atividade j no modo i, e a_{kt} é o conjunto de intervalos de tempo em que a demanda a_k do recurso do tipo k foi alcançada.

O conjunto F é formado pelas atividades j que não alocaram novos recursos para sua conclusão, assim as atividades que serão alteradas pelo novo modelo, são consideradas criticas para a solução em relação ao custo. O conjunto F deve conter mais de 60% no número total de atividade, caso isso não ocorra, novas atividades são escolhidas de forma aleatória para que a solução criada s'' não seja descaracterizada da solução original s'. Todas as outras variáveis descritas na Formulação Matemática, também são utilizadas nessa adaptação e os novos conjuntos criados são:

- s' = escalonamento viável com todas as atividades j alocadas no tempo t no modo i.
- a_{kt} = conjunto de intervalos de tempo t que foram utilizados toda a demanda do recurso do tipo k.
- F = conjunto de atividades de uma solução s' que satisfazem a condição $t'_j - d'_{ji} \notin a_{kt}$.

$$Minimizar \sum_{k=1}^{m} C_k(a_k) \tag{11}$$

sujeito a:

$$\sum_{i=1}^{M_j} \sum_{t=EF_j}^{LF_j} x_{jit} = 1,\ j = 1, ..., n,\ \forall j \notin F \tag{12}$$

$$x_{jit} = 1,\ \forall j \in F,\ \forall j, i, t \in s' \tag{13}$$

$$\text{Restrições } (3), (4), (5), (6) \tag{14}$$

A descrição acima, modifica a restrição (2) do modelo original removendo as atividades pertencentes ao conjunto F, e adiciona a restrição (13) que fixa o tempo t o modo i da atividade $j \in F$ na solução s'. Todas as outras restrições (14) permaneceram idênticas ao modelo original.

Ao final da busca local exata, as soluções escolhidas para serem modificadas pelo novo modelo, são reavaliadas em busca de novos valores para seus $pbest$, apos essa etapa uma verificação de quantas épocas já ocorreram é efetivada, e caso o número de gerações não tenha alcançados 50, o algoritmo retorna ao processo de busca do novos $pbest$ e$gbest$, caso contrário o algoritmo entra na faze final para a criação da curva de *trade-off*, baseada na heurística da Fronteira de Pareto, onde uma solução s^1 é dita dominante da solução s^2, se ambas as condições a seguir forem satisfeita:

- A solução s^1 não é pior do que a solução s^2 em nenhuma das grandeza (custo e tempo), ou seja, $custo_{s^1} \leq custo_{s^2}$ e $tempo_{s^1} \leq tempo_{s^2}$.
- A solução s^1 é melhor do que a solução s^2 em no mínimo uma grandeza, ou seja, $custo_{s^1} < custo_{s^2}$ ou $tempo_{s^1} < tempo_{s^2}$.

O conjunto final de soluções serão formados por indivíduos as quais suas dominantes não foram conhecidas pelo algoritmo. Esse processo entrega o conjunto de soluções que formarão o *trade-off* para o decisor eleger entre uma solução balanceada que atenda suas expectativas de tempo e custo do projeto.

5. Resultados e Conclusões

Uma adaptação foi necessária para utilizar os dados de entrada disponível na biblioteca PSPLIB. Nestas instâncias existe a inclusão dos recursos não renováveis, porém para no PCDRMM é considerar todos os recursos como renováveis. Esses dados, são diferenciados pela quantidade de atividades com 10, 20 e 30 identificadas como (J10), (J20) e (J30) respectivamente.

O modelo e sua adaptação, foram implementados no software *ILOG CPLEX*. Todo código deste trabalho foi desenvolvido na linguagem de programação C++, e os testes executados em um computador com processador Intel Core i5 2,3 GHz, com 4 GB de memória Ram. Os parâmetros de configurações do PSO com a Matheuristica (PSOM) : $E = 60$ quantidade de épocas, $A = 50$ quantidade de partículas no enxame, $w = 0,35$ inércia da velocidade atual, $c1 = 0,30$ coeficiente da experiência local, e $c2 = 0,45$ coeficiente da experiência global, foram obtidos por treinamento através da ferramenta IRACE Ibáñez et al. [2011], com 50 iterações em instâncias excluídas dos resultados finais.

Nas tabelas abaixo o custo das soluções são apresentados pelas colunas com o "(C)", e o tempo de execução que é dado em segundos demarcadas com "(T)". Foram necessária 50 execuções de cada algoritmo para chegar nos resultados abaixo em um conjunto de 51 instâncias.

Um outro conjunto de 12 instâncias testes criado no Software Progen para o PCDRMM foi disponibilizado por Yamashita e Morabito, nas quais todas as atividades possuem 3 modos de execução distintos, 4 tipos de recursos disponíveis e 12 atividades, sendo a primeira e a última atividades virtuais, com tempo e quantidade de recursos nulas.

Instâncias	CPLEX (C)	PSOM (C)	CPLEX(T)	PSOM (T)
1	38,76	44,78	1,03	2,46
2	117,10	*117,10*	2,86	2,01
3	277,29	292,56	39,44	2,95
4	376,10	378,31	55,28	5,83
5	93,49	101,75	4,49	3,87
6	121,52	132,36	1,57	2,74
7	105,52	111,36	7,08	2,38
8	148,75	156,67	79,42	4,71
9	21,31	*21,31*	1,73	2,94
10	172,07	194,34	4,64	3,82
11	104,31	*104,31*	90,19	3,93
12	159,84	166,22	102,67	5,54

Table 2: Comparação entre PSOM e o *CPLEX* com instâncias do PCDRMM

Uma diferença de 4,90% separam os resultados obtidos do PSOM das soluções ótimas, contudo, o PSOM consegue alcançar em 3 das 12 instâncias o valor obtido pelo *CPLEX*, essa comparação se inverte quando observado o tempo de processamento computacional do *CPLEX* retardando em média de 73,05% dos valores obtidos pelo PSOM.

Instâncias	CPLEX (C)	PSO (C)	Diferença	CPLEX (T)	PSO (T)	Diferênça
51 x J10	3547,22	4618,10	30,19%	5780,33	130,08	97,75%
2 x J20	108,54	152,93	40,90%	33502,41	4,73	99,99%
51 x J30	*	6543,97	*	*	328,40	*

Table 3: Comparação entre PSO com o *CPLEX*

Com um tempo máximo de 5 horas de processamento, foram obtidos resultados em apenas duas instâncias pelo *CPLEX* nas instâncias de 20 atividades (J20), isso se agrava em instâncias de 30 atividades (J30), as quais não conseguimos encontrar nenhum resultado obtido pelo *CPLEX*.

Instâncias	CPLEX (C)	PSOM (C)	Diferença	CPLEX (T)	PSO (T)	Diferênça
51 x J10	3547,22	3694,04	4,14%	5780,33	346,15	94,01%
2 x J20	108,54	113,48	4,55%	33502,41	8,31	99,98%
51 x J30	*	5943,97	*	*	610,52	*

Table 4: Comparação entre PSOM com o *CPLEX*

Para alcançar os resultados da Tabela 5, a etapa da Matheurística foi inserida no algoritmo que proporcionou uma melhora de 16,03% nas soluções. É notável com o crescimento do número de atividades, que o tempo de processamento para a criação de uma solução do *CPLEX* torna-se exponencial e inviabiliza o uso em aplicações reais, já o PSOM se mantém com a mesma proporção de tempo de processamento por quantidade de atividades nas instância, justificando a adoção do PSOM como um método para construção de soluções.

A Matheurística PSOM, constitui-se como uma boa alternativa para solucionar o Problema de Custo de Disponibilidade de Recursos com Múltiplos Modos, uma vez que cria soluções com diferença de apenas 4% dos valores ótimos em um tempo 94% vezes menor do que os obtidos pela implementação matemática do problema. O PSOM obteve um desempenho conclusivamente superior na solução das instâncias examinadas. Em decorrência, sugere-se um potencial promissor para a solução do PCDRMM conforme presentemente adaptado ou mesmo sua aplicação em problemas com estrutura matemática correlata.

Pela experiência adquirida no teste computacional, sugere-se que a modelagem dos movimentos dos recursos entre as atividades, bem como o uso de aperfeiçoamento heurístico baseada em grafo de dependência descrito em Procópio et al. [2013]. são alternativas promissoras de melhoria de performance do algoritmo proposto. Adicionalmente aconselha-se o desenvolvimento de heurísticas construtivas como o GRASP ou o uso dos controles de memória típicos da Busca Tabu para hibridizar as etapas de busca local.

References

Blazewicz, J., Lenstra, J., e Rinnooy Kan, A. (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics*, 5:11–24.

Calazans, A. T. S. (2012). Qualidade da informação: conceitos e aplicações. *TransInformação*, 20 (1).

Chen, L., Li, X., e Cai, Z. (2012). Heuristic methods for minimizing resource availability costs in multi-mode project scheduling. *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*.

Hsu, C. C. e Kim, D. S. (2005). A new heuristic for the multi-mode resource investment problem. *Journal of the Operational Research Society*.

Ibáñez, M. L., Dubois-Lacoste, J., Stutzle, T., e Birattari, M. (2011). The irace package, iterated race for automatic algorithm configuration. (TR/IRIDIA/2011-004). URL `http://iridia.ulb.ac.be/IridiaTrSeries/IridiaTr2011-004.pdf`.

Kennedy, J. e Eberhart, R. (1995). Particle swarm optimization. *International Conference on Neural Networks*.

Möhring, R. (1984). Minimizing costs of resource requirements, project networks subject to a fixed completion time. *Operations Research*, 32:89–120.

Nadjafi, B. A. (2014). Multi-mode resource availability cost problem with recruitment and release dates for resources. *Applied Mathematical Modelling*, 38.

Procópio, L. D. P., Costa, W., Filho, G. F. S., Cabral, L. F. A., e Silva, G. C. (2013). Uma abordagem híbrida aplicada ao problema da alocação dinâmica de espaços. *11th Brazilian Congress on Computational Intelligence (BRICSCCI and CBIC)*.

Qi, J.-J., Liu, Y.-J., Jiang, P., e Gou, B. (2015). Schedule generation scheme for solving multi-mode resource availability cost problem by modified particle swarm optimization. *Journal of Scheduling*, 18:285–298.

Talbot, F. B. (1982). Resource-constrained project scheduling problem with time- resource tradeoffs: The nonpreemptive case. *Management Science*, 28(10):1197 – 1210.

Taylor, F. W. (2004). *Scientific management*. Routledge.

Yamashita, D. e Morabito, R. (2012). Um algoritmo branch-and-bound para o porblema de programação de projetos com custo de disponibilidade de recursos e múltiplos modos. *Gestão e Produção*.