X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará

USANDO REDES NEURAIS ARTIFICIAIS PARA RECOMENDAR META-HEURÍSTICAS PARA O PROBLEMA DO CAIXEIRO VIAJANTE

Jorge Y. Kanda
Instituto de Ciências Exatas e de Tecnologias – Universidade Federal do Amazonas – Itacoatiara, Brasil
Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, São Carlos, Brasil
jkanda@ufam.edu.br, kanda@icmc.usp.br

André C.P.L.F. Carvalho, Eduardo R. Hruschka
Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo – São Carlos, Brasil
{andre,erh}@icmc.usp.br

Carlos Soares
LIAAD-INESC Porto LA/Faculdade de Economia – Universidade do Porto – Porto, Portugal
csoares@fep.up.pt

Resumo – Diversos algoritmos de otimização podem encontrar boas soluções para diferentes exemplos do problema do caixeiro viajante (PCV). Como não existe um algoritmo que gere a melhor solução para todos os exemplos, a seleção do algoritmo mais promissor para um dado exemplo de PCV é uma tarefa difícil. Este artigo descreve uma abordagem baseada em meta-aprendizado para selecionar algoritmos de otimização para o PCV. Redes neurais artificiais (RNA) são treinadas com exemplos de PCV. Os exemplos são descritos por um conjunto de características do PCV e pelas soluções fornecidas por um conjunto de algoritmos. O modelo de RNA treinado é usado para predizer um *ranking* desses algoritmos para novos exemplos de PCV. Medidas de correlação são usadas para comparar o *ranking* predito com um *ranking* previamente conhecido. Os resultados obtidos por essas medidas sugerem que a abordagem proposta é promissora.

Palavras-chave – Seleção de algoritmos, meta-aprendizado, problema do caixeiro viajante, redes neurais artificiais.

1. INTRODUÇÃO

Problemas de otimização podem ser encontrados em diversas áreas, tais como: transporte, engenharia, indústria, planejamento e economia [1]. A solução ótima para um problema de otimização é o valor mínimo (ou máximo) de uma função objetivo que satisfaça um conjunto de restrições. A função objetivo avalia cada solução representada por uma combinação dos valores das variáveis do problema e as restrições são condições pertinentes ao mesmo que não devem ser violadas pela solução [2]. Um exemplo clássico do problema de otimização é o bem-conhecido problema do caixeiro viajante (PCV). Informalmente, um PCV pode ser descrito por um conjunto de cidades e um conjunto de valores de custos (e.g. distâncias) entre os pares de cidades. A solução para um PCV é a rota de menor custo possível. A rota é iniciada em uma das cidades, as demais cidades são visitadas uma única vez, retornando-se à cidade inicial, onde a rota termina [3]. Algumas das propriedades relevantes em um PCV são: a simetria e o grau de conectividade. Um PCV é simétrico se os custos são iguais nos dois sentidos de deslocamento entre duas cidades quaisquer que sejam adjacentes, caso contrário é assimétrico. Um PCV é fortemente conectado se existe pelo menos uma conexão para cada par de cidades, caso contrário é denominado fracamente conectado [2].

A melhor solução para muitos exemplos de PCV pode não ser encontrada por um procedimento de busca exaustiva, pois em um PCV fortemente conectado com n cidades existem $(n-1)!$ rotas possíveis e, para cada rota, n operações matemáticas são realizadas. Por exemplo, o custo de todas as rotas de um PCV com 50 cidades seria obtido após o cálculo de 10^{64} operações, que terminariam somente após 10^{45} séculos em um computador com capacidade de processamento de um milhão de operações por segundo. Esse tempo de processamento inviabiliza o uso desse procedimento para resolver a maioria dos exemplos reais de PCV.

Devido à dificuldade para encontrar a solução ótima, heurísticas são geralmente aplicadas em problemas de otimização dessa natureza. As heurísticas encontram rapidamente boas soluções locais, mas não necessariamente a ótima global. Soluções ainda melhores podem ser obtidas por meta-heurísticas, que tentam escapar dos valores ótimos locais direcionando as buscas para regiões mais promissoras. Diversas meta-heurísticas podem encontrar boas soluções para o PCV, tais como: Busca Tabu (BT) [4], *Greedy Randomized Adaptive Search Procedure* (GRASP) [5], *Simulated Annealing* (SA) [6,7] e Algoritmos Genéticos (AG) [8, 9].

Se várias meta-heurísticas podem encontrar soluções para um novo exemplo de PCV, qual delas pode obter a melhor solução? Esta questão, originalmente apresentada por Rice [10], representa um típico problema de seleção de algoritmos. Em um cenário ideal, sem limitação de recursos (tempo, memória, processador etc.), a melhor alternativa seria executar todos os algoritmos disponíveis e escolher aquele que encontra a melhor solução. Entretanto, essa alternativa muitas vezes não é factível na prática por seu alto custo computacional. Além disso, o Teorema "*No Free Lunch*"garante que não há algoritmo capaz de gerar a melhor solução para qualquer tipo de problema, pois seu desempenho será sempre melhor para um subconjunto do problema e pior para

os demais [11]. Diante desse contexto, as principais contribuições desse artigo são: (i) descrever o uso de meta-aprendizado para a seleção de algoritmos, em particular, para o PCV; (ii) apresentar um novo conjunto de meta-atributos a partir de propriedades do PCV capaz de facilitar o aprendizado do modelo preditor; (iii) propor uma abordagem para recomendar um *ranking* de algoritmos para o PCV; (iv) comparar três tipos de abordagens de meta-aprendizado.

O restante deste artigo está organizado como segue. A seção 2 descreve resumidamente meta-aprendizado no contexto de seleção de algoritmos. Alguns trabalhos relacionados são abordados na seção 3. Para introduzir a notação adotada, uma breve descrição sobre o PCV está contida na seção 4. As configurações experimentais e os principais resultados são detalhados na Seção 5. E, finalmente, as conclusões são apresentadas na seção 6.

2 META-APRENDIZADO PARA SELEÇÃO DE ALGORITMOS USANDO REDES NEURAIS ARTIFICIAIS

Meta-aprendizado induz o meta-conhecimento capaz de selecionar os algoritmos mais promissores para um dado problema. Esse problema pode ser de diferentes naturezas, como, por exemplo aprendizado de máquina (AM) ou otimização (OT) [12]. O meta-conhecimento é o resultado de um mapeamento das características de um conjunto de problemas para o desempenho de um conjunto de algoritmos quando aplicados para esses problemas. As características essenciais dos problemas são transformadas em meta-atributos e o desempenho dos algoritmos em um atributo-alvo. Neste trabalho, o desempenho representa a qualidade da solução de um algoritmo ou a posição da solução por ele encontrada em um *ranking* de soluções obtidas por um conjunto de algoritmos. Em meta-aprendizado, os metadados são instâncias de um conjunto de exemplos do problema descritas pelos meta-atributos e atributo-alvo. Um subconjunto do metadados é usado por uma técnica de aprendizado para induzir um modelo capaz de predizer o valor do atributo-alvo para novos problemas do conjunto de dados. Mais detalhes sobre meta-aprendizado podem ser encontrados em [12].

A abordagem proposta de usar meta-aprendizado na seleção de algoritmos para o PCV está ilustrada na Figura 1. Basicamente, o processo de construção é dividido em duas fases: a primeira consiste na aquisição do meta-conhecimento sobre um conjunto de exemplos de PCV por meio da identificação de características essenciais dos exemplos (meta-atributos) e da obtenção das soluções das meta-heurísticas (atributos-alvo) para esses exemplos. A identificação das características que melhor representam os desempenhos dos algoritmos é determinante para o sucesso de uma abordagem de meta-aprendizado [12]. Na segunda fase, um modelo preditivo é induzido por meio de uma técnica de aprendizado. Nesse trabalho a técnica de aprendizado é uma rede neural artificial (RNA) multicamadas.

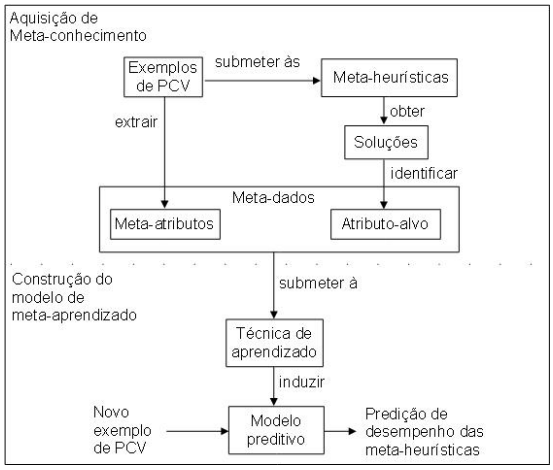

Figura 1: Processo de meta-aprendizado na seleção de algoritmos para PCV.

Uma RNA multicamadas é constituída de uma camada de entrada, uma ou mais camadas escondidas e uma camada de saída [13]. RNA é bastante aplicada em problemas não-lineares. Nesse estudo, as RNAs foram treinadas com o algoritmo *back-propagation* [14]. O aprendizado decorre das atualizações dos pesos dos neurônios, que são calculadas a partir das informações do erro de cada neurônio de saída. Desse modo, uma RNA pode ser capaz de predizer vários valores de saída para um mesmo exemplo (neste trabalho cada valor é relativo a uma meta-heurística). Mais detalhes sobre RNA podem ser encontrados em [13].

O modelo de aprendizado induzido pelas RNAs está ilustrado na Figura 2. A saída do modelo é um *ranking* de algoritmos de otimização para cada novo exemplo de TSP e sua entrada é o conjunto de meta-atributos com valores extraídos para o exemplo. O algoritmo na menor posição no *ranking* indica ser o mais promissor para ser aplicado no exemplo. A predição da posição do *ranking* de cada algoritmo é feita através de uma função de regressão não-linear, que é uma maneira de estimar a expectativa condicional da variável dependente, mantendo fixas as variáveis independentes [15]. O sucesso de predição do modelo de aprendizado proposto é medido por meio de taxas de correlação entre os *rankings* preditos e os *rankings* desejados.

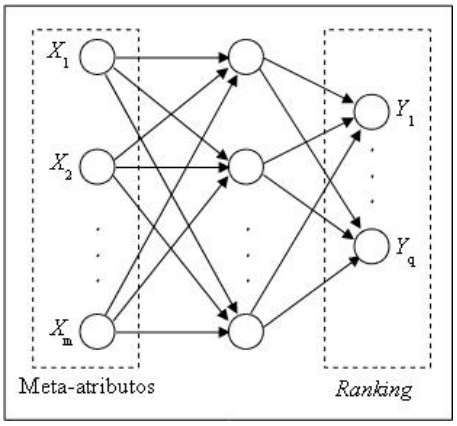

Figura 2: Modelo de RNA para seleção de algoritmos.

3 TRABALHOS RELACIONADOS

Nos últimos anos, meta-aprendizado tem sido aplicado com sucesso na seleção de algoritmos em vários tipos de problemas [12]. Um método de meta-aprendizado descrito em [16] auxilia usuários na seleção de algoritmos de AM para um conjunto de dados. Para isso, o método avalia os valores dos meta-atributos de todos os conjuntos de dados disponíveis, identificando os mais similares ao conjunto de dados desejado. A recomendação dos algoritmos de aprendizado para esse conjunto é feita por meio de um *ranking* dos algoritmos que obtiveram as melhores soluções para os conjuntos de dados similares.

A idéia de aplicar regressão em problemas de meta-aprendizado foi primeiramente abordada em [17] para predizer o erro de um algoritmo de aprendizado para um conjunto de dados utilizando meta-atributos definidos no projeto STATLOG. Vários modelos de regressão foram avaliados e os resultados reportados por meio do Erro Quadrático Médio Normalizado (NMSE). Um trabalho similar foi apresentado em [18], que investigou o uso de meta-aprendizado para estimar a acurácia preditiva de um classificador por meio de modelos de regressão.

Apesar das pesquisas em meta-aprendizado serem mais focadas na seleção de algoritmos de classificação, alguns trabalhos aplicam meta-aprendizado em outras áreas, como a otimização. Em [19], modelo de meta-aprendizado com RNA é usado para selecionar algoritmos de otimização para o problema do assinalamento quadrático.

AM é também usada em problemas de satisfatibilidade proposicional como em [20], que descreve a abordagem SATzilla para selecionar algoritmo que resolva exemplos desse tipo de problema. Modelos de AM são usados para predizer o tempo de execuções dos algoritmos para um dado exemplo do problema baseado em atributos de exemplos similares e tempo de execução conhecido dos algoritmos nesses exemplos.

Uma abordagem de meta-aprendizado para recomendar algoritmos para o PCV está descrita em [21]. Exemplos de PCV são classificados conforme as soluções obtidas por um conjunto de meta-heurísticas. A melhor solução para os exemplos pode ser obtida por mais de uma meta-heurística, por isso técnicas de classificação multirrótulo são aplicadas no conjunto original. Em seguida, algoritmos de classificação usam os conjuntos resultantes para induzir os modelos de aprendizado.

Um outro trabalho que aplica meta-aprendizado em PCV pode ser encontrado em [22]. Exemplos de PCV foram classificados conforme a facilidade ou a dificuldade de cada algoritmo para encontrar uma boa solução. Os meta-atributos foram gerados a partir das informações de localização bidimensional das cidades. Para um aprendizado não supervisionado, um algoritmo de árvore de decisão gerou regras, baseado nos meta-atributos, para descrever as classes dos exemplos de PCV. Para um aprendizado supervisionado, um modelo de RNA usou os exemplos de PCV para predizer o log de esforço de busca da melhor solução para cada algoritmo, classificando cada exemplo com o algoritmo para o qual foi predito o menor esforço.

As pesquisas em aprendizado de *ranking* com RNA são frequentemente aplicados para buscas de páginas na internet [23–25]. No entanto, as pesquisas em recomendação de algoritmos para PCV por meio de técnicas de aprendizado são recentes [21, 22]. Esse artigo apresenta abordagens de aprendizado de *ranking* usando RNA para recomendar os algoritmos mais promissores para exemplos de PCV. O aprendizado é feito a partir de um conhecimento prévio sobre um conjunto de exemplos de PCV, que são descritos por um novo conjunto de meta-atributos e pelas soluções conhecidas de algumas meta-heurísticas.

4 PROBLEMA DO CAIXEIRO VIAJANTE

Formalmente, um Problema do Caixeiro Viajante (PCV) é definido por meio de um grafo $G = (V, A)$, onde $V = \{1, 2, ..., n\}$ é um conjunto de vértices e $A = \{\langle i, j\rangle : i,j \in V\}$ é um conjunto de arestas. Cada vértice $i \in V$ representa uma cidade do problema e cada aresta $\langle i, j\rangle$ conecta os vértices i e j. O custo de viagem da cidade i para a cidade j é indicado pelo valor c_{ij} associado à aresta $\langle i, j\rangle$. Se $c_{ij} = c_{ji} \; \forall \; \langle i, j\rangle \in A$, o PCV é simétrico. Se $\exists \; \langle i, j\rangle \in A$ para cada par de vértice (i,j), o PCV é fortemente conectado. A melhor solução para um PCV é dado pelo ciclo hamiltoniano de menor custo possível. Um ciclo é hamiltoniano, se a partir de uma cidade todas as demais são visitadas uma única vez e, ao final, retorna para a cidade inicial [26].

Boas soluções para um PCV podem ser geralmente obtidas por meta-heurísticas, porque fornecem soluções próximas da solução ótima com um custo computacional relativamente baixo. Existem inúmeras pesquisas (e.g. [27, 28]) que procuram aperfeiçoar as metodologias subjacentes às meta-heurísticas e outros trabalhos (e.g. [29–31]) que comparam os desempenhos de diferentes algoritmos para o PCV, em termos de soluções e tempo de processamento.

Em nossa pesquisa, as meta-heurísticas (BT, GRASP, SA e AG) são usadas para avaliar a capacidade de recomendação de algoritmos para o PCV por meio de um processo de meta-aprendizado.

5 EXPERIMENTOS

Para avaliar a abordagem proposta, RNAs foram treinadas a partir de exemplos de PCV. Mais especificamente, os modelos de aprendizado foram induzidos a partir de três abordagens de RNA. A primeira, denominada $tipo1$, usa uma RNA com quatro neurônios na camada de saída, que identificam os valores relativos às soluções das meta-heurísticas. A segunda abordagem, denominada $tipo2$, também usa uma RNA com quatro saídas, porém as mesmas indicam as posições das meta-heurísticas em um *ranking* de recomendação. Finalmente na terceira abordagem ($tipo3$), baseada na estratégia de predição de *ranking* para classificadores [18], quatro modelos de RNA com um neurônio de saída foram implementadas para, cada modelo, predizer a posição de uma das meta-heurísticas no *ranking* sugerido. Para efeito de comparação entre os resultados obtidos pelas três abordagens, as soluções conhecidas e preditas para cada meta-heurística na abordagem $tipo1$ serão inseridas em uma lista para a composição de um *ranking*.

5.1 Preparação dos dados

O sucesso da predição de um modelo de aprendizado depende, dentre outros fatores, de uma quantidade significativa de exemplos usados no treinamento. Como não há disponibilidade de uma grande quantidade de exemplos reais de PCV, foram gerados 2000 subproblemas de PCV simétricos e fortemente conectados. Esses problemas foram gerados a partir de 10 arquivos ($d1655$, $fl1400$, $nrw1379$, $pcb3038$, $pr2392$, $rl1889$, $u1817$, $vm1748$, $fnl4461$ e $rat783$) disponíveis na biblioteca TSPLIB [32]. O nome do arquivo indica o número de cidades envolvidas no problema. Por exemplo, $d1291$ é um PCV com 1291 cidades. De cada arquivo, 200 subproblemas diferentes foram gerados a partir de 100 cidades escolhidas aleatoriamente, sendo mantidos os valores originais de custos entre as cidades.

Todos os 2000 exemplos foram submetidos às meta-heurísticas: Busca Tabu (BT) [4], *Greedy Randomized Adaptive Search Procedure* (GRASP) [5], *Simulated Annealing* (SA) [6,7] e Algoritmos Genéticos (AG) [8,9]. Cada meta-heurística possui diversos parâmetros livres e após vários experimentos preliminares, foram usadas as configurações que maximizaram o desempenho de cada algoritmo: BT: tamanho da lista tabu $= 3$; número de iterações sem melhora na solução corrente $= 2$; GRASP: número de iterações $= 50$; $\alpha = 0.6$; SA: $\beta = 0.9$; $t_0 = 1$; $\alpha = 0.01$; GA: operador de cruzamento = PMX [33]; tamanho da população = 128; seleção dos pais por torneio; taxa de mutação = 5%; seleção da próxima geração por elitismo.

Essas meta-heurísticas são estocásticas e por isso podem fornecer diferentes resultados após cada execução. Para determinar os valores das soluções das meta-heurísticas para cada exemplo, os algoritmos foram executados 50 vezes sob um mesmo tempo de processamento e a melhor solução foi obtida como medida de desempenho da meta-heurística em questão. Para cada exemplo, as soluções foram transformadas em desempenhos relativos para a composição de um *ranking* de meta-heurísticas, que foi usado como atributo-alvo pelos modelos de meta-aprendizado nas abordagens $tipo2$ e $tipo3$. Enquanto que para o $tipo1$, os atributos-alvos foram representados pelas soluções obtidas pelas meta-heurísticas. A Tabela 1 mostra o desempenho relativo percentual de cada *ranking* de meta-heurística. O *ranking* mais frequente foi (AG, SA, GRASP e BT) em 38.5% dos exemplos. Essa ordem majoritária é usada na Seção 5.4 como uma referência para mensurar a qualidade dos resultados obtidos pelos modelos de meta-aprendizado. Por exemplo, pode-se afirmar que a escolha arbitrária de AG forneceria a melhor solução em 74.3% dos PCVs.

5.2 Extração de meta-atributos

Identificar as propriedades mais relevantes de um conjunto de problemas é outro fator fundamental para melhorar a capacidade preditiva de um sistema de meta-aprendizado. Para os exemplos de PCV ilustrados nesse artigo, algumas propriedades inerentes aos grafos correspondentes foram transformadas em meta-atributos. Essas propriedades são baseadas em medidas simples e foram extraídas a partir dos custos associados às arestas e aos vértices. Os custos relativos às arestas são facilmente identificados no grafo, enquanto o custo de um vértice (c_i) é definido como o valor resultante da razão entre a soma dos custos das arestas associadas a esse vértice e o número (n) de arestas conectadas ao mesmo:

$$c_i = \frac{\sum_{j=1}^{n} c_{ij}}{n} \tag{1}$$

A Tabela 2 lista os 14 meta-atributos usados para descrever os exemplos de PCV. Os sete primeiros meta-atributos são relativos aos custos de vértices e os demais são relativos aos custos de arestas. Os meta-atributos V e A indicam, respectivamente, o número de cidades e de arestas envolvidas no problema; V_{menor} e V_{maior} contém, respectivamente, o menor e o maior custo de vértice; Similarmente, A_{menor} e A_{maior} assumem o valor da aresta de menor e maior custo, respectivamente; Valores sobre as principais medidas estatísticas (média, desvio padrão e mediana) são obtidos pelos meta-atributos: V_{media}, V_{desvio} e $V_{mediana}$ para os vértices e A_{media}, A_{desvio} e $A_{mediana}$ para as arestas; Finalmente, os meta-atributos $V_{menores}$ e $A_{menores}$ capturam a

Tabela 1: Frequência relativa dos *rankings* de meta-heurísticas.

1o.	2o.	3o.	4o.	Frequência(%)
AG	SA	GRASP	BT	38.50
AG	SA	BT	GRASP	19.95
AG	GRASP	SA	BT	10.50
SA	AG	GRASP	BT	9.90
GRASP	AG	SA	BT	7.10
SA	AG	BT	GRASP	5.50
AG	BT	SA	GRASP	3.65
GRASP	SA	AG	BT	1.60
AG	BT	GRASP	SA	1.15
BT	AG	SA	GRASP	0.55
AG	GRASP	BT	SA	0.55
GRASP	AG	BT	SA	0.50
SA	GRASP	AG	BT	0.20
BT	SA	AG	GRASP	0.10
GRASP	BT	AG	SA	0.10
BT	GRASP	SA	AG	0.05
BT	AG	GRASP	SA	0.05
SA	BT	AG	GRASP	0.05

soma dos V menores custos de vértice e aresta, respectivamente. Os meta-atributos relativos aos custos de arestas $\{A_8, ..., A_{14}\}$ e o meta atributo A_1 são os mesmos usados em [21].

Tabela 2: Meta-atributos de PCV usados no treinamento do algoritmo de meta-aprendizado.

Código	Meta-atributo	Descrição
A_1	V	Quantidade de vértices
A_2	V_{menor}	Menor custo de vértice
A_3	V_{maior}	Maior custo de vértice
A_4	V_{media}	Médias dos custos dos vértices
A_5	V_{desvio}	Desvio padrão dos custos dos vértices
A_6	$V_{mediana}$	Mediana dos custos dos vértices
A_7	$V_{menores}$	Soma das V menores custos dos vértices
A_8	A	Quantidade de arestas
A_9	A_{menor}	Menor custo de aresta
A_{10}	A_{maior}	Maior custo de aresta
A_{11}	A_{media}	Médias dos custos das arestas
A_{12}	A_{desvio}	Desvio padrão dos custos das arestas
A_{13}	$A_{mediana}$	Mediana dos custos das arestas
A_{14}	$A_{menores}$	Soma das V menores custos de arestas

Antes de usar esses meta-atributos na entrada de um modelo de meta-aprendizado, o valor de cada meta-atributo para cada exemplo do conjunto de metadados foi normalizado (A'_e) por:

$$A'_e = \frac{A_e - min}{max - min}, \tag{2}$$

onde A_e é o valor de um dos meta-atributos no exemplo e; min e max representam, respectivamente, o menor e o maior valor do meta-atributo em questão nos metadados manipulados.

A normalização do meta-atributo alvo depende do tipo de aprendizado implementado no modelo. Por exemplo, para o aprendizado da abordagem $tipo1$, o valor da solução de cada meta-heurística para cada exemplo foi normalizado (S'_{he}) por:

$$S'_{he} = \frac{S_{he}}{\sum_{r=1}^{q} S_{re}}, \tag{3}$$

onde S_{he} é o valor da solução obtida pela meta-heurística h para o exemplo e; S_{re} é o valor da solução obtida por cada uma das meta-heurísticas restantes para o mesmo exemplo e; e q é a quantidade de meta-heurísticas candidatas.

5.3 Geração do modelo de aprendizado

Um conjunto de metadados foi criado a partir das 2000 instâncias de PCV, no qual cada instância é descrita pelos valores normalizados dos meta-atributos e dos atributos-alvo. Seguindo a prática padrão de manipulação de dados em experimentos com RNA [14], as instâncias foram distribuídas aleatoriamente de forma estratificada em três conjuntos: treinamento (60%), validação (20%) e teste (20%). O primeiro conjunto é usado pelo algoritmo de aprendizado para induzir o modelo preditor, o segundo é para interromper o treinamento e o terceiro para testar a capacidade de generalização do modelo.

As RNAs [1] foram treinadas com as seguintes configurações: 14 neurônios na camada de entrada, na qual cada neurônio corresponde a um meta-atributo extraído do PCV; 4 neurônios na camada de saída para os aprendizados $tipo1$ e $tipo2$ e 1 neurônio de saída para o $tipo3$. Para a camada escondida, a configuração que maximiza o desempenho da RNA depende do conjunto de dados [14]. Durante o treinamento foi adotada a metodologia de validação cruzada [32], sendo processadas RNAs com 2 a 36 neurônios na camada escondida.

A escolha da RNA que induziu o modelo de meta-aprendizado foi feita a partir do bem conhecido erro quadrático médio (EQM) obtido no conjunto de validação.

5.4 Resultados experimentais

Para cada uma das três abordagens de aprendizado adotadas em nossos experimentos, o menor valor de EQM no conjunto de validação foi identificado por uma RNA diferente, conforme mostra a Tabela 3. Dentre as abordagens de aprendizado, o $tipo1$ teve o menor erro por causa da informação representada pelo atributo-alvo, que é a solução de cada meta-heurística. Para cada instância de PCV no conjunto de meta-dados, os valores normalizados das soluções das meta-heurísticas ficaram muito mais próximos entre si do que os valores normalizados das posições do *ranking* das meta-heurísticas.

Tabela 3: Menor EQM das RNAs no conjunto de validação em três abordagens de aprendizado.

Abordagem	EQM	Número de neurônios na camada escondida
$tipo1$	8.562×10^{-5}	3
$tipo2$	6.414×10^{-3}	19
$tipo3$	7.307×10^{-3}	25

Uma vez identificado as melhores configurações das RNAs para cada abordagem, as mesmas foram utilizadas no conjunto de teste para avaliar a capacidade de generalização dos modelos induzidos. A qualidade desses modelos foi mensurada por meio de duas medidas de correlação: Coeficiente de Spearman (CS) [34] e Coeficiente de Goodman-Kruskal Ponderado (GKP) [35]. A primeira medida é basicamente a soma dos erros ao quadrado das posições do *ranking*, podendo ser vista como uma medida normalizada do erro quadrático médio. A segunda medida pode ser vista como um complemento da primeira porque correlaciona duas sequências considerando a magnitude dos valores contidos nas mesmas, calculando não somente a tendência dos valores mas também a taxa de variação dos mesmos. Para essas duas medidas, o valor +1 indica uma perfeita correlação entre as duas sequências comparadas e o valor -1 significa que as sequências são completamente discordantes.

Os desempenhos das RNAs medidos pelo CS e GKP nas três abordagens de aprendizado estão mostrados na Tabela 4. Para determinar a qualidade de cada resultado, o mesmo é comparado com um valor de referência. No caso do coeficente de Spearman, o valor de referência (terceira coluna) é a média desse coeficiente obtida entre as listas conhecidas sobre a ordem das meta-heurísticas e a lista majoritária no conjunto de dados. Enquanto que o valor de referência para GKP (sexta coluna) representa a média dessa taxa de correlação entre o *ranking* das meta-heurísticas para cada instância do conjunto de dados e o *ranking* médio das mesmas. Como os atributos-alvos da abordagem $tipo1$ são diferentes das outras duas abordagens, a magnitude dos valores no *ranking* desejado é diferente para essa abordagem. Por isso, o valor de referência para GKP é diferente para o $tipo1$. A quarta e a sétima coluna indicam o grau de superioridade dos modelos de meta-aprendizado em relação a um modelo de predição padrão cujo desempenho é estimado pelo valor de referência. Os resultados mostram que os modelos de meta-aprendizado tiveram uma qualidade melhor na predição do ranking com uma taxa de erro menor que a recomendação do *ranking* majoritário para todas as instâncias.

Tabela 4: Desempenho das predições dos modelos de meta-aprendizado no conjunto de teste.

Abordagem	CS	Valor de referência (R)	(CS-R)/R	GKP	Valor de referência (R)	(GKP-R)/R
$tipo1$	0.8210	0.7895	0.040	0.8106	0.7538	0.075
$tipo2$	0.8330	0.7895	0.055	0.8346	0.6712	0.243
$tipo3$	0.8255	0.7895	0.046	0.7599	0.6712	0.132

As duas medidas de correlações adotadas sugerem que o desempenho do modelo de aprendizado foi maior no modelo induzido por uma RNA com 4 saídas ($tipo2$), conforme mostram os valores da quarta e da sétima coluna da Tabela 4. A abordagem

[1] implementadas no Software R usando o pacote *neuralnet* sem alteração dos valores padrão.

$tipo1$ adotada para o aprendizado da solução relativa dos algoritmos tem a desvantagem de ser necessário ordenar primeiramente os valores preditos antes de recomendar as meta-heurísticas para o usuário. E a abordagem $tipo3$ tem o custo de implementar várias redes cuja quantidade depende das meta-heurísticas disponíveis para ser aplicada a um dado PCV.

6. CONCLUSÃO

O problema da seleção de algoritmos é sempre uma tarefa complexa, não sendo muito diferente no contexto da escolha de uma meta-heurística para encontrar a melhor solução para um exemplo de PCV. Neste artigo, foi apresentada uma proposta para aplicar meta-aprendizado na recomendação das meta-heurísticas mais promissoras para o PCV. Modelos de meta-aprendizado foram induzidos a partir de três abordagens de Redes Neurais Artificiais. As redes foram treinadas com um conjunto de instâncias para os quais os desempenhos das meta-heurísticas são previamente conhecidos. Os modelos treinados foram usados em duas abordagens para estimar o *ranking* das meta-heurísticas para um novo exemplo, e em uma outra abordagem, o modelo treinado estimou as soluções relativas das meta-heurísticas. Dentre os modelos, o melhor foi aquele que foi usado uma RNA para predizer os *rankings* de soluções das meta-heurísticas. Os resultados das medidas de correlação entre o *ranking* predito e o desejado indicaram que o melhor modelo teve um desempenho superior a uma predição do *ranking* majoritário, com menor erro de predição (aproximadamente 5%) e com uma qualidade melhor do *ranking* predito (aproximadamente 24%).

Para serem usados como atributos de entrada em um modelo de meta-aprendizado, foi proposta neste artigo um novo conjunto de meta-atributos extraídos a partir de características contidas na representação gráfica dos exemplos de PCV. A identificação dos meta-atributos apropriados é essencial para o sucesso de um sistema de meta-aprendizado. Os resultados experimentais sugerem que a pesquisa sobre o uso de meta-aprendizado para a recomendação das melhores meta-heurísticas para um PCV é promissora. Em futuros trabalhos, a abordagem de meta-aprendizado proposta será aplicada em exemplos de PCV, para os quais cada cidade é identificada por um ponto em um espaço bidimensional. Com as informações de localização das cidades será possível extrair meta-atributos como aqueles usados em [22] e outros que serão investigados para melhorar a capacidade de predição do modelo induzido.

AGRADECIMENTOS

Os autores agradecem CAPES, CNPq, FAPESP e FAPEAM pelo suporte financeiro. Este trabalho foi parcialmente suportado pelo Projeto FCT Rank! (PTDC/EIA/81178/2006).

REFERÊNCIAS

[1] M. Bazaraa, J. Jarvis and H. Sheralt. *Linear Programming and Networks Flows*. Wiley-Interscience, Hoboken, NJ, third edition, 2005.

[2] D. Bertsimas and J. Tsitsiklis. *Introduction to linear optimization*. Athena Scientific, Belmont, Mass, 1997.

[3] G. Gutin and A. Punnen. *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, Netherlands, 2002.

[4] F. Glover, E. Taillard and D. de Werra. "A user's guide to tabu search". *Annals of Operations Research*, vol. 41, pp. 3–28, 1993.

[5] T. Feo and M. Resende. "Greedy Randomized Adaptive Search Procedures". *Journal of Global Optimization*, vol. 6, pp. 109–133, 1995.

[6] S. Kirkpatrick, C. Gelatt and M. Vecchi. "Optimization by Simulated Annealing". *Science*, vol. 220, pp. 671–680, 1983.

[7] V. Cerny. "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm". *Journal of Optimization Theory and Applications*, vol. 45, pp. 41–51, 1985.

[8] J. Holland. "Genetic Algorithms and the Optimal Allocations of Trial". *SIAM J. Comp.*, vol. 2, pp. 88–105, 1973.

[9] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, first edition, 1989.

[10] J. Rice. "The Algorithm Selection Problem". *Advances in Computers*, vol. 15, pp. 65–118, 1976.

[11] D. Wolpert and W. Macready. "No free lunch theorems for optimization". *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67–82, 1997.

[12] P. Brazdil, C. Giraud-Carrier, C. Soares and R. Vilalta. *Metalearning: Applications to Data Mining*. Springer, Berlin, 2009.

[13] S. Haykin. *Neural Networks and Learning Machines*. Pearson Education inc, New York, N.Y., third edition, 2009.

[14] M. Smith. *Neural Networks for Statistical Modeling*. International Thomson Computer Press, Boston, 1996.

[15] G. Seber and C. Wild. *Nonlinear Regression*. John Wiley and Sons, New York, 1989.

[16] P. Brazdil, C. Soares and J. D. Costa. "Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results". *Machine Learning*, vol. 50, pp. 251–257, 2003.

[17] J. Gama and P. Brazdil. "Characterization of Classification Algorithms". In *Progress in Artificial Intelligence*, edited by C. Pinto-Ferreira and N. Mamede, pp. 189–200. Springer-Verlag, 1995.

[18] H. Bensusan and A. Kalousis. "Estimating the Predictive Accuracy of a Classifier". In *Proceedings of the 12th European Conference on Machine Learning*, edited by P. Flach and L. de Raedt, pp. 25–36. Springer, 2001.

[19] K. Smith-Miles. "Towards insightful algorithm selection for optimisation using meta-learning concepts". In *Proceedings of the IEEE International Joint Conference on Neural Networks 2008*, volume 978, pp. 4118–4124, 2008.

[20] L. Xu, F. Hutter, H. H. Hoos and K. Leyton-Brown. "SATzilla: portfolio-based algorithm selection for SAT". *J. Artif. Int. Res.*, vol. 32, pp. 565–606, June 2008.

[21] J. Kanda, A. Carvalho, E. Hruschka and C. Soares. "Using Meta-learning to Classify Traveling Salesman Problems". *Brazilian Symposium on Neural Networks*, pp. 73–78, 2010.

[22] K. Smith-Miles, J. van Hemert and X. Lim. "Understanding TSP difficulty by learning from evolved instances". In *Proceedings of the 4th international conference on Learning and intelligent optimization*, volume 6073 of *LION'10*, pp. 266–280, Berlin, Heidelberg, 2010. Springer-Verlag.

[23] F. Scarselli, S. L. Yong, M. Gori, M. Hagenbuchner, A. C. Tsoi and M. Maggini. "Graph Neural Networks for Ranking Web Pages". In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '05, pp. 666–672, Washington, DC, USA, 2005. IEEE Computer Society.

[24] L. Lu, R. Safavi-Naini, M. Hagenbuchner, W. Susilo, J. Horton, S. L. Yong and A. C. Tsoi. "Ranking Attack Graphs with Graph Neural Networks." In *ISPEC*, edited by F. Bao, H. Li and G. Wang, volume 5451 of *Lecture Notes in Computer Science*, pp. 345–359. Springer, 2009.

[25] S. L. Yong, M. Hagenbuchner and A. C. Tsoi. "Ranking Web Pages Using Machine Learning Approaches". In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, pp. 677–680, Washington, DC, USA, 2008. IEEE Computer Society.

[26] D. Applegate, R. Bixby, V. Chvátal and W. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, Princeton, 2006.

[27] J. Santos, F. L. Júnior, R. Magalhães, J. Melo and A. D. Neto. "A Parallel Hybrid Implementation Using Genetic Algorithms, GRASP and Reinforcement Learning for the Salesman Traveling Problem". In *Computational Intelligence in Expensive Optimization Problems*, edited by L. M. Hiot, Y. S. Ong, Y. Tenne and C.-K. Goh, volume 2 of *Adaptation, Learning, and Optimization*, pp. 345–369. Springer Berlin Heidelberg, 2010.

[28] P. Pop, O. Matei and C. Sabo. "A New Approach for Solving the Generalized Traveling Salesman Problem". In *Proceedings of the 7th international conference on Hybrid metaheuristics*, volume 6373, pp. 62–72, Berlin, Heidelberg, 2010. Springer-Verlag.

[29] T. Stützle, A. Grün, S. Linke and M. Rüttger. "A Comparison of Nature Inspired Heuristics on the Traveling Salesman Problem". In *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, pp. 661–670, London, UK, 2000. Springer-Verlag.

[30] D. Johnson and L. McGeoch. "Experimental Analysis of Heuristics for the STSP". In *Local Search in Combinatorial Optimization*, pp. 369–443. Wiley & Sons, 2001.

[31] T. Öncan, I. Altinel and G. Laporte. "A comparative analysis of several asymmetric traveling salesman problem formulations". *Computers & Operations Research*, vol. 36, no. 3, pp. 637–654, 2009.

[32] G. Reinelt. "TSPLIB - A Traveling Salesman Problem Library". *Informs Journal on Computing*, vol. 3, no. 4, pp. 376–384, 1991.

[33] D. Goldberg and R. L. Jr. "Alleles, loci, and the traveling salesman problem". In *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, edited by J. J. Grefenstette, pp. 154–159. Publishers Lawrence Erlbaum Associates, 1985.

[34] C. Spearman. "The Proof and Measurement of Association between Two Things". *The American Journal of Psychology*, vol. 100, no. 3/4, pp. 441–471, 1987.

[35] R. Campello and E. Hruschka. "On comparing two sequences of numbers and its applications to clustering analysis". *Information Sciences*, vol. 179, no. 8, pp. 1025–1039, 2009.