RESOLVENDO O PROBLEMA DO CAIXEIRO VIAJANTE VIA PROCEDIMENTO DE BUSCA ADAPTATIVA ALEATÓRIA GULOSA COM CONSTRUÇÃO BASEADA EM REDES NEURAIS AUTO-ORGANIZÁVEIS

Lucas de S. Batista*, Alan R.R. de Freitas†, Frederico G. Guimarães†, Jaime A. Ramírez*

*Dep. de Engenharia Elétrica, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil

†Dep. de Computação, Universidade Federal de Ouro Preto, Ouro Preto, MG, Brasil

Emails: lusoba@ufmg.br, alandefreitas@gmail.com, frederico.g.guimaraes@gmail.com, jramirez@ufmg.br

Abstract— Most combinatorial optimization problems belong to the *NP* class, which means that finding the optimal solution by enumeration methods may require an infeasible processing time. Thus, the use of heuristics is often preferred, for they are capable of providing a good trade-off between the quality of the solution and the time spent to find it. This paper proposes a meta-heuristic for solving the traveling salesman problem (TSP), in which a greedy randomized adaptive search procedure (GRASP) is performed, employing a self-organizing map (SOM) in the construction phase, and the Lin-Kernighan heuristic for the local search. The results for 15 benchmark instances present an average error of about 1% in relation to the best known solutions in the literature.

Keywords— Traveling salesman problem, Self-organizing networks, Combinatorial optimization.

Resumo— A maioria dos problemas de otimização combinatória pertencem à classe *NP*, o que significa que a sua solução ótima por meio de métodos enumerativos pode exigir um tempo de processamento inviável. Assim, o uso de heurísticas é frequentemente mais interessante, pois são capazes de encontrar uma relação *custo/benefício* aceitável entre a qualidade da solução e o tempo de processamento. Este artigo propõe uma meta-heurística para a solução do problema do caixeiro viajante (TSP), em que um procedimento de busca adaptativa aleatória gulosa (GRASP) é implementado, empregando-se um mapa auto-organizável (SOM) na fase construtiva, e a heurística de Lin-Kernighan para a busca local. Os resultados para 15 instâncias testadas apresentam um erro médio de aproximadamente 1% em relação aos ótimos conhecidos na literatura.

Keywords— Problema do caixeiro viajante, redes auto-organizáveis, otimização combinatória.

1 Introdução

O estudo de problemas de otimização combinatória é relevante tanto do ponto de vista teórico quanto prático. Estes problemas são comumente encontrados em diversas situações reais, tais como, roteamento de veículos ("Vehicle Routing Problems" (VRP)) e programação de tabela de horários ("Scheduling"), e são usualmente classificados como *NP-completo* ou *NP-difícil* (Lewis and Papadimitriou, 1997), o que implica na não existência de uma solução determinística capaz de ser executada em tempo polinomial. Por isso, o uso de heurísticas é frequentemente mais interessante (Glover and Kochenberger, 2003), pois são capazes de encontrar uma relação *custo/benefício* aceitável entre a qualidade da solução e o tempo necessário para o seu cálculo.

O problema do caixeiro viajante ("Traveling Salesman Problem" (TSP)) representa um problema combinatorial padrão que pertence à classe *NP*, sendo largamente utilizado para testar a eficiência de novas técnicas de otimização combinatória. Em poucas palavras, dado um conjunto de n cidades, o caixeiro viajante deve visitar todas elas uma única vez e retornar ao ponto de partida, minimizando o custo da rota. Como é sabido, um TSP simétrico com n cidades possui $(n-1)!/2$ rotas possíveis, valor este que cresce fatorialmente com o número de cidades.

Dentre as heurísticas propostas para a solução de problemas combinatórios, cita-se, por exemplo, o algoritmo k-opt, em que k laços da rota corrente são substituídos por outros k laços a cada iteração, tal que uma rota de menor custo seja produzida. Baseado neste estudo, Lin e Kernighan (Lin and Kernighan, 1973) propuseram uma modificação do k-opt, tal que o valor ideal de k é determinado a cada passo, melhorando assim o desempenho do método. Além destes, algumas meta-heurísticas eficientes são recozimento simulado (SA), busca tabu (TS), colônia de formigas (ACO), algoritmos evolucionários (EA), procedimento de busca adaptativa aleatória gulosa (GRASP), redes neurais artificiais auto-organizáveis (ANN), e algumas variantes do mapa auto-organizável de Kohonen (SOM) (Blum and Roli, 2003; Smith, 1999; Kohonen, 2001).

A aplicação de mapas auto-organizáveis em problemas de otimização é interessante, pois os mesmos geralmente não requerem a avaliação da qualidade das soluções geradas ao longo do processo, e além disso, são capazes de encontrar soluções eficientes que podem ser usadas como a solução final do problema, ou podem ser combinados a outras técnicas visando a realização de buscas locais sobre as soluções geradas. Considerando a questão descrita, este artigo propõe a resolução do problema do caixeiro viajante utilizando GRASP, em que a fase construtiva será

desenvolvida por um mapa de Kohonen, e a busca local será desempenhada pela heurística Lin-Kernighan. As próximas seções são destinadas a descrição destes métodos e apresentação dos resultados.

2 Procedimento de Busca Adaptativa Aleatória Gulosa

O Procedimento de Busca Adaptativa Aleatória Gulosa (GRASP) (Resende and Ribeiro, 2002) representa uma meta-heurística para a solução de problemas combinatórios, em que cada iteração consiste basicamente de duas fases, sendo a primeira *construtiva*, e a segunda uma *busca local*. A fase de construção é responsável pela geração de uma solução viável, cuja vizinhança é investigada até que um mínimo local seja encontrado durante a fase de busca local. Ao fim do procedimento, a melhor solução encontrada é retornada. A estrutura fundamental de funcionamento do GRASP é mostrada no Alg. 1.

Algoritmo 1: GRASP
Data: num_max_iter
for ($\lambda \leftarrow 1$ **to** *num_max_iter)* **do**
  Solução ← Construção_Aleatória_Gulosa(ψ);
  Solução ← Busca_Local(Solução);
  Atualiza_Solução(Solução,Melhor_Solução);
end
Retorna Melhor_Solução;

Conforme observado, o GRASP pode ser facilmente implementado, e além disso, possui apenas dois parâmetros principais, sendo o primeiro referente ao critério de parada adotado, e o segundo é o parâmetro que pondera entre a característica aleatória e determinística da solução gerada durante a fase de construção. Dessa forma, considerando-se o parâmetro normalizado $\psi \in [0, 1]$, pode-se assumir que para $\psi = 0$ tem-se a construção de soluções de maneira mais determinística, enquanto $\psi = 1$ equivale a uma construção mais aleatória. Logo, para o emprego eficaz da rede de Kohonen na fase construtiva do GRASP, torna-se necessário a identificação dos possíveis parâmetros que influenciam na diversidade das soluções geradas pelo mapa auto-organizável.

Como a etapa de busca local não representa a contribuição fundamental deste artigo, será enfatizado principalmente a fase de construção por redes auto-organizáveis de Kohonen.

3 Mapas Auto-Organizáveis de Kohonen

As redes auto-organizáveis (SOM) possuem o seguinte funcionamento básico (Braga et al., 2007): quando um padrão de entrada p é apresentado aos neurônios, a rede procura a unidade neural mais parecida com p. Durante o seu treinamento, a rede aumenta a semelhança do neurônio escolhido e de seus vizinhos ao padrão p. Dessa forma, a rede constrói um mapa topológico onde neurônios que estão topologicamente próximos respondem de forma semelhante a padrões de entrada semelhantes. O mecanismo de atualização dos pesos da rede é realizado por meio de um algoritmo de aprendizado competitivo.

As subseções seguintes se dedicam a explicação do algoritmo construtivo auto-organizável, bem como ao estudo do comportamento da rede frente a variação de seus parâmetros, que representa a principal contribuição do trabalho.

3.1 Algoritmo Construtivo Auto-Organizável

O algoritmo baseado nos mapas de Kohonen utilizado para a solução do problema do caixeiro viajante é descrito por meio de quatro etapas principais, as quais são: *inicialização da rede*, *competição*, *cooperação*, e *adaptação*.

- Inicialização da Rede:

Como será tratado o TSP Euclidiano, a camada de entrada é representada pelas coordenadas cartesianas das cidades, ou seja, $X_i = (x_{i1}, x_{i2})$, $i = 1, \ldots, m$ em que m é o número de cidades. Já a camada de saída é representada pelo vetor de pesos dos neurônios, $W_j = (wj1, wj2)$, $j = 1, \ldots, n$ em que o número de neurônios é dado por $n = \beta m$. Uma vez que a topologia da camada de saída adotada é em anel, os neurônios são inicializados em uma caixa retangular que engloba todas as cidades da planta, sendo dispostos uniformemente ao longo desta caixa.

- Competição:

A etapa de competição consiste em encontrar o neurônio que mais se assemelha a um dado padrão apresentado. Assim, o neurônio J cuja distância Euclidiana à cidade apresentada for mínima, vence a competição e é selecionado (1):

$$J = \arg\min_j \|X_i - W_j\| \ \ \forall j \tag{1}$$

em que X_i corresponde às coordenadas cartesianas da cidade i, e W_j é o vetor de coordenadas do neurônio j. Observe que durante o processo de competição, um determinado neurônio só será considerado vencedor, caso não tenha vencido nenhuma competição durante a presente iteração. Caso contrário, escolhe-se o neurônio que mais se assemelha ao padrão de entrada e que ainda não tenha sido escolhido como vencedor.

- Cooperação:

Quando um neurônio J vence a competição, este se move em direção ao padrão reconhecido e, consequentemente, os neurônios vizinhas se movem na mesma direção, porém de forma mais amena.

A função de vizinhança (Nb_{Jj}) comumente empregada durante a etapa de cooperação é apresentada em (2):

$$Nb(\sigma, d_{Jj}) = \exp(-d_{Jj}^2/\sigma^2) \quad (2)$$

$$d_{Jj} = \min\{\|J - j\|, n - \|J - j\|\} \quad (3)$$

em que d_{Jj} representa o grau de vizinhança entre o neurônio vencedor J e neurônio vizinho j, sendo d_{Jj} dado pela distância cardinal mínima entre os neurônios Jj na estrutura em anel (3). O parâmetro $\sigma(t)$ é reduzido iterativamente ao longo do processo (4):

$$\sigma(t) = \sigma_0 \exp(-t/\tau_1) \quad (4)$$

em que σ_0 está relacionado com a porcentagem de neurônios vizinhos que são influenciados por um dado neurônio vencedor quando $t = 1$. Visando uma rápida convergência da rede, a constante de tempo estabelecida por Kohonen é reduzida de 100 vezes, ou seja, $\tau_1 = 10/\log(\sigma_0)$.

- Adaptação:

Por fim, baseando-se no treinamento competitivo, os neurônios da rede são submetidos à função de adaptação (5):

$$W_j(t+1) = W_j(t) + \alpha.Nb.(X_i - W_j(t)) \quad (5)$$

$$\alpha(t) = \alpha_0 \exp(-t/\tau_2) \quad (6)$$

em que α representa a taxa de aprendizado da rede (6), α_0 é um parâmetro inicial, e $\tau_2 = 10$ é a constante proposta por Kohonen reduzida de 100 vezes.

Note que quando $\alpha(t).Nb(\sigma(t), d_{Jj})$ for desprezível, então a rede de neurônios mantém-se estável, podendo este critério ser utilizado para interromper a auto-organização do mapa de Kohonen. De forma simplificada, o critério de parada adotado neste trabalho é acionado quando $Nb(\sigma(t), 1) \leq 10^{-3}$, ou seja, quando $\sigma(t) \leq 0.38$. Como exemplo, considerando-se $\sigma_0 = 10$, obtém-se o limite $t = 33$ iterações.

3.2 Geração de Diversidade nas Redes SOM

De forma a avaliar a capacidade de geração de soluções com diferentes graus de diversidade, os principais parâmetros do mapa de Kohonen (α_0, β, σ_0) são estudados. Para a realização dos testes, gerou-se uma instância aleatória definida no espaço $[0, 1]^2$ composta por 30 cidades. Todos os resultados a serem apresentados representam valores médios relativos a 100 execuções do GRASP.

- Influência do Parâmetro α_0:

A taxa de aprendizado inicial da rede é variada no intervalo $\alpha_0 \in [0.5, 2.0]$, mantendo-se os demais parâmetros constantes ($\beta = 2.0$ e $\sigma_0 = 0.06$). De forma geral, observa-se pela Fig. 1 que o mapa de Kohonen é capaz de gerar melhores soluções para a faixa de valores $\alpha_0 \in [1.0, 1.5]$, apresentando basicamente o mesmo custo computacional para todos os valores de α_0 considerados (Fig.2).

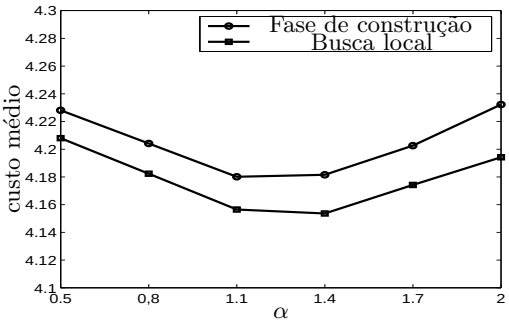

Figura 1: Comparação entre o custo médio obtido durante a fase de construção e o custo médio após a aplicação da busca local em função de α_0.

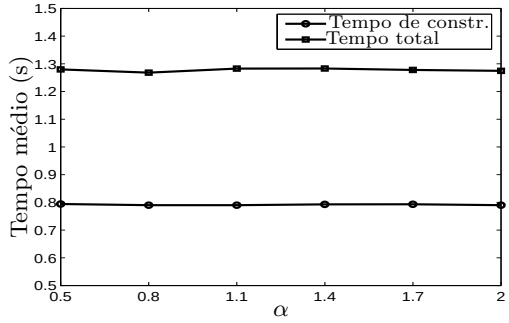

Figura 2: Comparação entre o tempo médio consumido durante a fase de construção e o tempo médio total após a aplicação da busca local em função de α_0.

- Influência do Parâmetro β:

O fator β é diretamente proporcional ao número de neurônios empregados para a solução de uma dada instância ($n = \beta m$). O parâmetro em questão é variado no intervalo $\beta \in [1, 2]$, fixando-se $\alpha_0 = 0.5$ e $\sigma_0 = 0.06$. Os histogramas mostrados na Fig. 3, bem como as curvas médias apresentadas na Fig. 4, indicam a influência de β no grau de diversidade das soluções geradas pela rede de Kohonen, proporcionando a construção de soluções melhores distribuídas para reduzidos valores de β. Tomando-se por base o custo computacional mostrado na Fig. 5, observa-se um aumento deste com o aumento do número de neurônios da rede. Visando obter soluções com certo grau de diversidade, e com um custo computacional aceitável, sugere-se $\beta \in [1.2, 1.6]$.

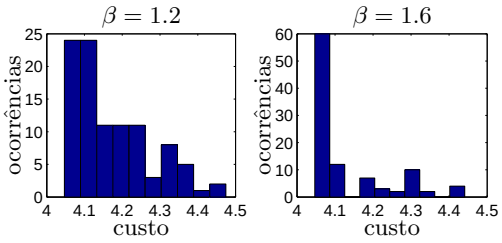

Figura 3: Distribuição das soluções após a fase de construção em função do parâmetro β.

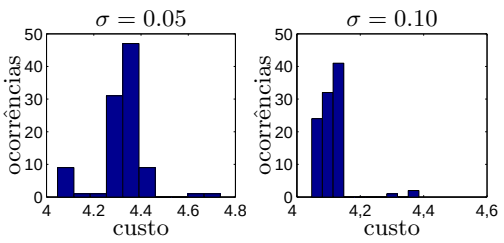

Figura 6: Distribuição das soluções após a fase de construção em função do parâmetro σ_0.

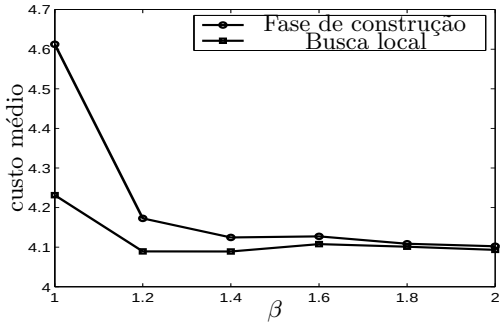

Figura 4: Comparação entre o custo médio obtido durante a fase de construção e o custo médio após a aplicação da busca local em função de β.

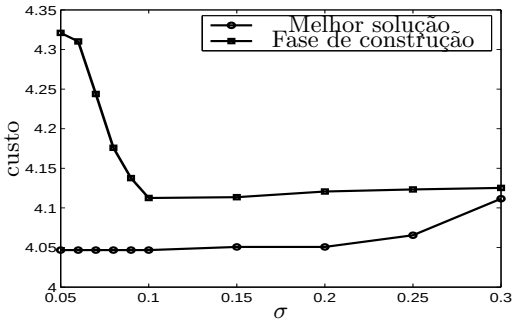

Figura 7: Comparação entre o custo médio obtido durante a fase de construção e a melhor solução gerada nesta etapa em função do parâmetro σ_0.

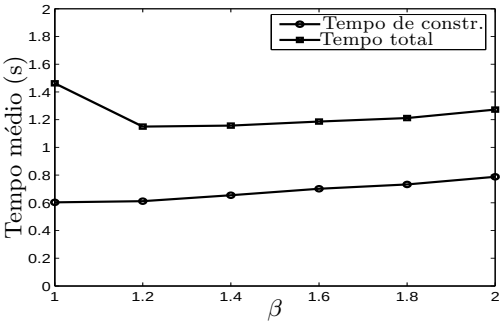

Figura 5: Comparação entre o tempo médio consumido durante a fase de construção e o tempo médio total após a aplicação da busca local em função de β.

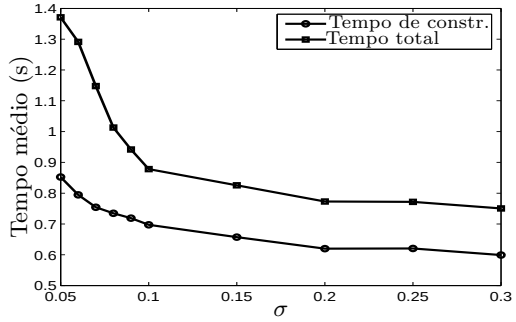

Figura 8: Comparação entre o tempo médio consumido durante a fase de construção e o tempo médio total após a aplicação da busca local em função do parâmetro σ_0.

- Influência do Parâmetro σ_0:

Uma vez que σ_0 representa a variância inicial da função de vizinha Nb, este parâmetro está diretamente relacionado ao número de neurônios que são influenciados pelo neurônio vencedor no início do processo de identificação de padrões. Assim sendo, de forma que 25 a 100% dos neurônios da rede sejam influenciados na iteração $t = 1$, o parâmetro sob estudo deverá variar no intervalo $\sigma_0 \in [0.05n, 0.30n]$. Os demais parâmetros são mantidos constantes ($\alpha_0 = 0.5$ e $\beta = 2$). Como está evidente pelas Figs. 6 e 7, as soluções geradas apresentam um maior grau de diversidade para valores reduzidos de σ_0, entretanto, o custo computacional se eleva a medida que σ_0 decresce. Com o objetivo de manter diversidade nas soluções geradas, e ainda considerar a relação qualidade por tempo consumido, sugere-se $\sigma_0 \in [0.08n, 0.15n]$.

4 Resultados

O procedimento aleatório guloso é aplicado a 15 instâncias sugeridas na literatura (Reinelt, 1991). Os parâmetros adotados para a rede de Kohonen são $\alpha_0 = 1.5$, $\beta = 1.5n$ e $\sigma_0 = 0.10n$. Os re-

Tabela 1: Resultados computacionais encontrados pelo GRASP. *EPM* é o erro percentual médio; *DPM* é o desvio percentual médio; *Melhor* é a melhor solução encontrada; e t é o tempo médio consumido.

–	**Fase de construção**				**Após busca local**			
Planta	EPM	DPM	Melhor	t (s)	EPM	DPM	Melhor	t (s)
berlin52	6.73	4.04	0.00	1.51	5.15	3.67	0.00	0.05
bier127	5.78	2.43	1.63	8.01	4.19	1.86	0.94	0.19
eil51	5.20	1.75	3.29	1.45	3.22	0.98	0.70	0.03
eil76	5.92	1.31	3.35	3.04	3.78	0.99	2.23	0.07
kroA200	5.51	1.69	2.81	19.10	3.63	1.20	1.98	0.35
lin105	3.76	2.42	0.73	5.68	2.70	1.98	0.17	0.12
pcb442	11.35	1.13	9.05	90.66	7.46	1.26	5.15	1.29
pr76	3.63	1.60	1.70	3.02	2.07	1.15	0.72	0.08
pr107	1.56	0.66	0.59	5.82	1.03	0.57	0.00	0.10
pr136	6.89	0.94	5.42	8.93	4.21	0.87	2.05	0.15
pr152	2.83	1.44	0.94	11.15	2.27	1.24	0.77	0.20
rat99	6.17	2.15	2.48	5.00	4.13	2.11	0.83	0.09
rat195	11.19	1.32	8.78	17.89	7.72	1.25	5.17	0.26
rd100	4.75	2.21	0.62	5.02	3.27	1.62	0.43	0.09
st70	3.32	1.50	0.59	2.57	2.64	1.25	0.59	0.05

sultados médios obtidos ao longo de 30 iterações do GRASP[1] são mostrados na Tab. 1, e melhor visualizados na Fig. 9.

A menos das instâncias *pcb*442 e *rat*195, o erro médio obtido na fase construtiva encontra-se próximo a 4.8%, sendo que o erro médio relativo às melhores soluções não ultrapassa 1.9%. Após a aplicação da busca local estes dados se reduzem para 3.4% e 0.9%, respectivamente.

Embora os resultados sejam interessantes, torna-se necessário a realização de novos testes considerando-se instâncias maiores. Uma possível forma de melhorar o desempenho do procedimento proposto é considerar um GRASP reativo (Resende and Ribeiro, 2002), o que permite identificar ao longo do processo iterativo os valores mais adequados para os parâmetros α_0, β e σ_0 em função das características da instância sob teste.

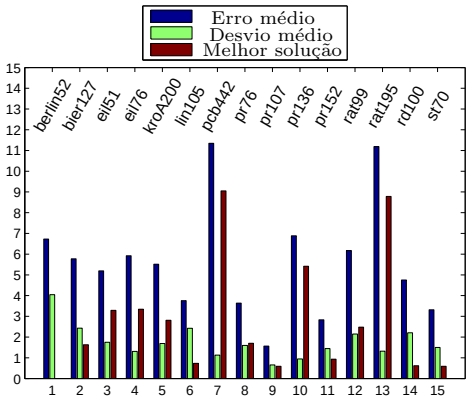

Figura 9: Comparação entre o erro percentual médio, o desvio padrão percentual médio e a melhor solução encontrada durante a fase de construção.

[1]Intel32 Core 2 Duo CPU T8300, 2.4GHz - 2GB RAM (considerou-se um único núcleo).

5 Conclusões

Este artigo propôs a utilização do procedimento de busca adaptativa aleatória gulosa para a solução do problema do caixeiro viajante, em que a fase de construção foi implementada por meio de uma rede auto-organizável de Kohonen, e a busca local foi desenvolvida pela heurística Lin-Kernighan. Os resultados finais, considerando-se as 15 instâncias testadas, apresentam um erro médio de aproximadamente 1% em relação aos valores ótimos conhecidos na literatura, evidenciando assim a eficiência do mecanismo proposto, bem como a pertinência deste trabalho no que diz respeito à otimização de problemas combinatórios.

Referências

Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Computing Surveys* **35**(3): 268–308.

Braga, A. P., Carvalho, A. C. P. L. F. and Ludermir, T. B. (2007). *Redes Neurais Artificiais - Teoria e Aplicações*, 2nd edn.

Glover, F. W. and Kochenberger, G. A. (2003). *Handbook of Metaheuristics*.

Kohonen, T. (2001). *Self-Organizing Maps*, 3rd edn.

Lewis, H. R. and Papadimitriou, C. H. (1997). *Elements of the Theory of Computation*, 2nd edn.

Lin, S. and Kernighan, B. (1973). An effective heuristic algorithm for the traveling-salesman problem, *Operations Research* **21**(2): 498–516.

Reinelt, G. (1991). Tsplib – a traveling salesman problem library, *ORSA Journal On Computing* **3**(4): 376–384.

Resende, M. G. and Ribeiro, C. C. (2002). Greedy randomized adaptive search procedures, *Technical report*, AT&T Labs Research.

Smith, K. A. (1999). Neural networks for combinatorial optimization: A review of more than a decade of research, *INFORMS Journal On Computing* **11**(1): 15–34.