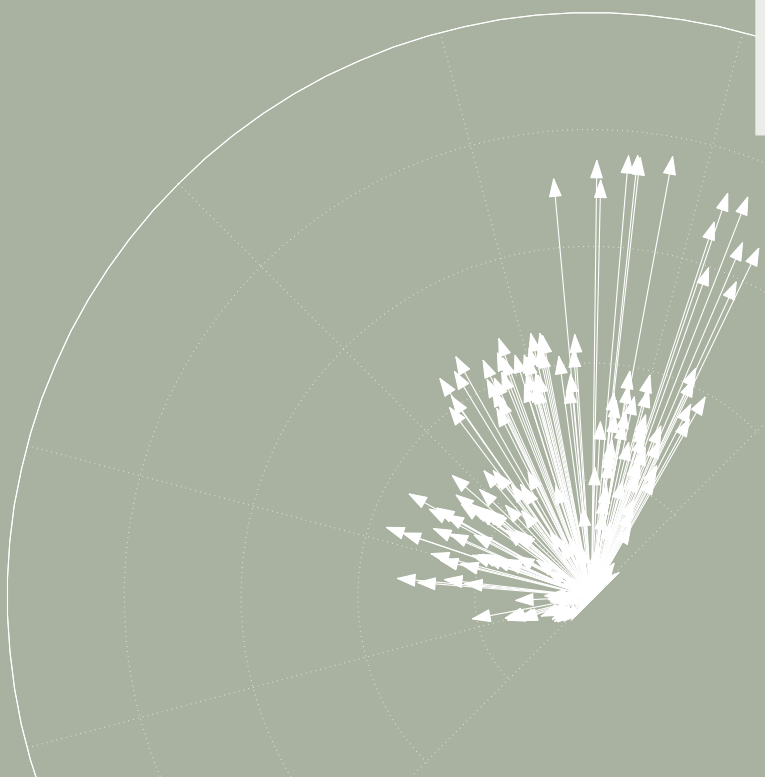
MANUAL DE COMPUTAÇÃO EVOLUTIVA E METAHEURÍSTICA

ANTÓNIO GASPAR-CUNHA
RICARDO TAKAHASHI
CARLOS HENGGELER ANTUNES
COORDENADORES



IMPRENSA DA UNIVERSIDADE DE COIMBRA
COIMBRA UNIVERSITY PRESS

EDITORA ufmg

CAPÍTULO 5

Colônia de Formigas

Priscila V. Z. C. Goliatt *Jaqueline S. Angelo* *Helio J. C. Barbosa*

Laboratório Nacional de Computação Científica, Petrópolis, RJ

Aborda-se neste capítulo uma metaherística de inspiração natural relativamente recente: a Otimização por Colônia de Formigas (em inglês *Ant Colony Optimization*, ou ACO). Para isto apresenta-se inicialmente o comportamento de uma colônia de formigas que deu origem à ACO, os principais experimentos realizados e os modelos matemáticos desenvolvidos na literatura. Antes de passar ao primeiro algoritmo da "família ACO"coloca-se o Problema do Caixeiro Viajante (PCV) – o primeiro a ser abordado pela ACO – que fornece o cenário ideal para a apresentação da técnica. Em seguida, as principais variantes da ACO são abordadas, referindo-se sempre ao PCV. Algumas aplicações da ACO em outros problemas são listadas incluindo-se possíveis extensões da técnica. O caso importante de problemas de otimização com restrições é exemplificado em um problema de otimização de estruturas de barras. Em seguida discute-se como estender a ACO ao caso de múltiplos objetivos. Finalmente, algumas ideias sobre como tratar o caso, não previsto originalmente, de variáveis de decisão contínuas são apresentadas, e conexões com outras metaheurísticas são evidenciadas.

1. Aprendendo com as Formigas Reais

Quando olhamos para uma única formiga pensamos em sua fragilidade física e em suas limitações de ações e tomadas de decisão. Entretanto, pense agora em um conjunto de formigas. Uma colônia

desses insetos sociais é capaz de organizar-se de forma a realizar tarefas de extrema complexidade.

Uma colônia de formigas é organizada em castas cujas funções variam de acordo com o tamanho do inseto (*e.g.* proteger a colônia, buscar por alimentos, transportar alimentos). A interação entre os indivíduos ocorre através de um fenômeno denominado por estigmergia [1], termo introduzido pelo zoólogo francês Pierre-Paul Grassé em 1959 (Grassé, 1959). Estigmergia refere-se a noção de que uma ação de um determinado agente deixe sinais no meio ambiente, e este sinal poderá ser percebido por outros agentes (geralmente da mesma espécie) de forma a incitar ou determinar suas ações subsequentes. Em diversas espécies de formigas esta sinalização (ou comunicação) é feita através da deposição de feromônio (ferormônio ou feromona) no meio ambiente.

O feromônio[2] é uma substância química de reconhecimento usada para a sinalização de, por exemplo, alimento, perigo e maturação sexual. No caso da busca por alimento, Goss et al. (1989) realizaram uma experiência conectando o ninho de uma colônia de formigas a uma fonte de alimento através de uma ponte dupla. Foram realizados testes com dois módulos idênticos da ponte dupla variando apenas a relação $r = \frac{l_M}{l_m}$ entre o comprimento dos braços menor (l_m) e maior (l_M) da ponte, iniciando com $r = 1$. No teste com $r = 2$, ou seja $l_M = 2l_m$ (Figura 5.1), no instante inicial da exploração a primeira formiga saía do ninho escolhendo aleatoriamente um dos braços da ponte e, ao encontrar o alimento, esta retornava ao ninho também selecionando uma rota ao acaso (Figura 5.1a). Chegando ao ninho, as demais formigas eram recrutadas para a exploração de rotas até o alimento. No início do recrutamento, as formigas permaneciam escolhendo aleatoriamente qual braço da ponte seguir (Figura 5.1b). Após certo tempo, a maioria das formigas escolhiam a menor rota e, eventualmente, algumas formigas exploravam o trajeto mais longo (Figura 5.1c).

A explicação proposta é que ao iniciar a experiência, como nenhuma formiga percorreu nenhum dos trajetos (não havendo assim nenhum feromônio depositado no meio), os dois braços da ponte possuem a mesma chance de serem percorridos. Encontrando o alimento a formiga retorna ao seu ninho depositando feromônio que, ao longo do tempo, sofre uma evaporação natural. Como no braço de menor comprimento a deposição de feromônio é rapidamente realimentada (*feedback* positivo) por um número cada vez maior de formigas, após algum tempo esta rota torna-se mais atrativa para as formigas subsequentes.

Quanto a observação de que algumas formigas eventualmente percorriam outra rota, podemos ver em Dorigo e Stützle (2004) uma discussão sobre os resultados da experiência de Goss et al. (1989) com relação ao papel do feromônio. A evaporação do feromônio ocorre muito lentamente o que significa que uma vez depositado diferentes regiões podem ser exploradas. Desta forma, possíveis trilhas subótimas podem ser preteridas pela colônia quando uma melhor rota for encontrada.

2. Construindo Formigas Artificiais

Até então sabemos que na natureza as formigas conseguem otimizar a busca por alimento de acordo a quantidade de feromônio depositada no ambiente por formigas que percorreram este caminho anteriormente. Com base nestas informações, obtidas com a experiência da ponte dupla, um modelo estocástico foi desenvolvido no intuito de tentar reproduzir a dinâmica de uma colônia de formigas em busca por alimento (Goss et al., 1989; Deneubourg et al., 1990).

Neste modelo estocástico, Φ formigas por segundo cruzam a ponte depositando uma unidade de feromônio cada. A escolha entre os braços menor (m) e maior (M) da ponte a partir de um ponto/nó $i \in \{1, 2\}$, onde $i = 1$ refere-se ao caminho/arco ninho-alimento e $i = 2$ ao caminho/arco alimento-ninho, é dada por $p_{i,m}(t)$ e $p_{i,M}(t)$. Esta decisão depende das quantidades de feromônio em cada braço, denotadas por $\phi_{i,m}(t)$ e $\phi_{i,M}(t)$. Por exemplo, desconsiderando a evaporação do feromônio, no

[1] Estigmergia: palavra proveniente do grego stigma (marca, sinal) e érgon (ação, trabalho).

[2] Feromônio: palavra proveniente do grego féro (transportar, transmitir) e órmon, particípio presente de órmao (excitar).

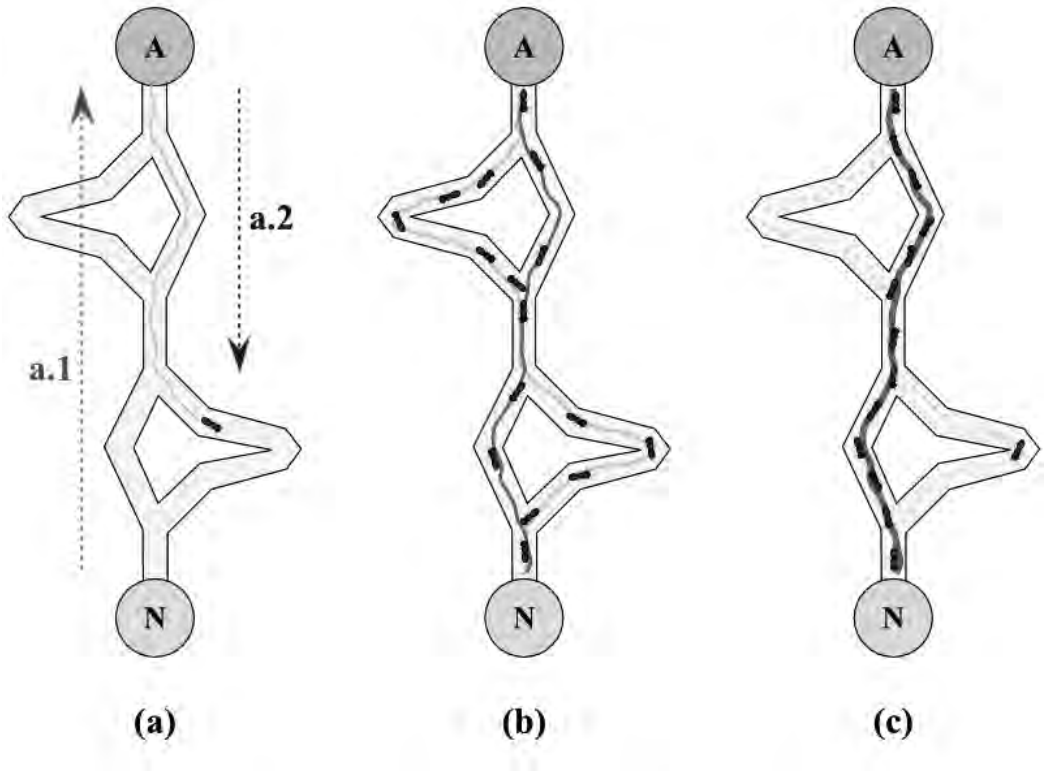


Figura 5.1: Experiência da ponte dupla com braços de tamanhos distintos. Esquema adaptado de Goss *et al* (1989). (a) Formiga inicia a exploração do meio (**a.1**) em busca por alimento. Ao encontrar o alimento (**A**), esta retorna ao ninho (**N**) depositando feromônio (**a.2**) ao longo da rota. (b) Demais formigas em **N** são recrutadas para a exploração de caminhos até **A**. (c) Após certo tempo, a maioria das formigas escolhem a menor rota (em vermelho). Eventualmente, algumas formigas são liberadas para explorar o território (linha tracejada).

instante t a probabilidade de se escolher o braço menor da ponte é dada por

$$p_{i,m}(t) = \frac{(k + \phi_{i,m}(t))^{\alpha}}{(k + \phi_{i,m}(t))^{\alpha} + (k + \phi_{i,M}(t))^{\alpha}} \qquad (p_{i,m} + p_{i,M} = 1) \tag{5.1}$$

onde os valores $k = 20$ e $\alpha = 2$ são derivados de observações experimentais e simulações computacionais, usando o método de Monte Carlo, realizados por Goss et al. (1989) e Deneubourg et al. (1990). As equações diferenciais que descrevem a dinâmica deste sistema são

$$\frac{d\phi_{i,m}}{dt} = \psi p_{i',m}(t - k) + \psi p_{i,m}(t), \qquad (i = 1, i' = 2; i = 2, i' = 1), \tag{5.2}$$

$$\frac{d\phi_{i,M}}{dt} = \psi p_{i',M}(t - kr) + \psi p_{i,M}(t), \qquad (i = 1, i' = 2; i = 2, i' = 1), \tag{5.3}$$

onde a constante $\psi = 0,5$ representa o fluxo de formigas (número de formigas por segundo). A constante k na equação (5.2) representa o tempo necessário para que as formigas atravessem o braço menor, enquanto kr na equação (5.3) expressa o mesmo mas para o braço maior. Na Figura 5.2 são apresentados os resultados da simulação de Monte Carlo, realizados por Goss et al. (1989), usando o modelo estocástico apresentado nas Equações (5.1), (5.2) e (5.3). Podemos observar que em 1.000

simulações de Monte Carlo, para $r = 1$ (braços da ponte de mesmo tamanho) os dois braços da ponte são escolhidos com igual probabilidade, enquanto para $r = 2$ (um dos braços é duas vezes maior que o outro) na maioria das simulações o braço menor foi preferencialmente escolhido.

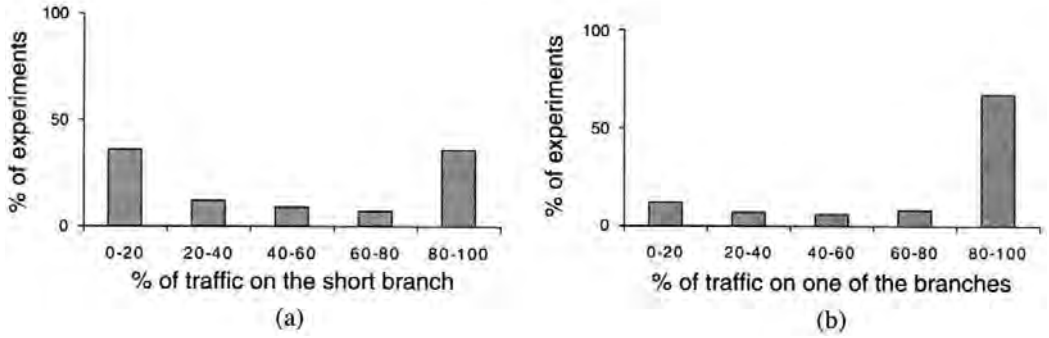


Figura 5.2: Resultado de 1.000 simulações de Monte Carlo usando o modelo estocástico dado pelas Equações (5.1), (5.2) e (5.3), para $\psi = 0,5$, $r = 1$ em (a) e $r = 2$ em (b). Figura retirada de Dorigo e Stützle (2004).

Visto que um conjunto de equações diferenciais consegue reproduzir as observações experimentais do comportamento de formigas reais em busca de alimento, usando caminhos mais curtos em um esquema de grafo simplificado (com apenas dois arcos), apresentaremos, na próxima seçao, os conceitos básicos para a construção de um algoritmo, inspirado neste comportamento das formigas reais, visando a solução de problemas de otimização que podem ser formulados em grafos/matrizes mais complexos.

3. Otimização por Colônia de Formigas

Computacionalmente, a metaheurística ACO é um método de busca construtivo no qual uma população de agentes (formigas artificiais) constroem cooperativamente soluções candidatas para um dado problema. A construção é probabilística, guiada pela heurística do problema e por uma memória compartilhada entre os agentes, contendo a experiência de iterações anteriores. Esta memória consiste de uma trilha artificial de feromônio, baseada na atribuição de pesos às variáveis das soluções candidatas. As variáveis do problema podem ser representadas por matrizes/grafos contendo os valores de sua informação heurística. Esta matriz é associada a uma matriz de taxa de feromônio como exemplificado na Figura 5.3.

Para a aplicação da ACO a um problema de otimização combinatorial, Dorigo e Stützle (2004) distinguem 6 passsos, sendo os 4 primeiros cruciais para um bom desempenho do algoritmo resultante:

1. Representar o problema usando conjuntos de componentes e transições, ou um grafo ponderado no qual as formigas construirão as soluções candidatas.

2. Definir adequadamente o que seriam as trilhas de feromônio (matriz τ) para o problema em questão.

3. Definir o que seria a informação heurística (matriz η) associada ao problema tratado, que cada formiga levaria em conta ao efetuar suas decisões.

4. Sempre que possível, implementar um algoritmo de busca local eficiente para o problema em questão.

5. Escolher uma das variantes de ACO já existentes (com ou sem modificações).

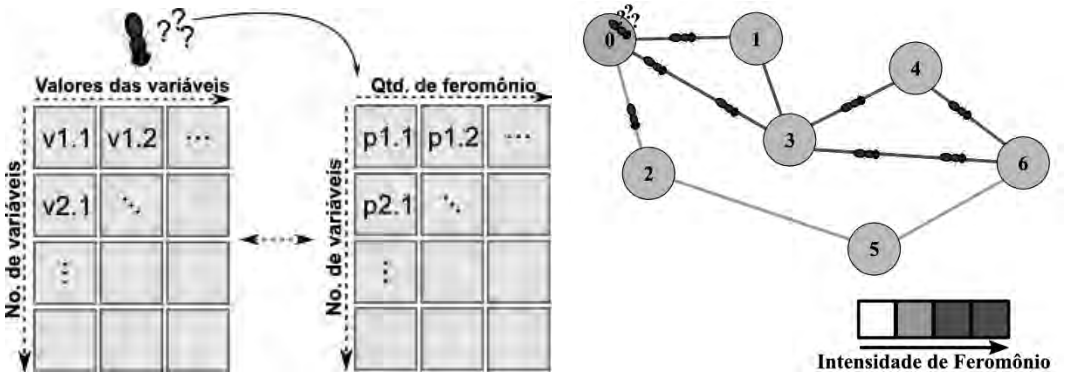


Figura 5.3: Esquema da representação e escolha das variáveis para a construção de uma rota (solução) em ACO, de acordo com a taxa de feromônio a elas associadas. Baseado em Di Caro e Dorigo (1997).

6. A partir de valores previamente utilizados em aplicações similares, ajustar os parâmetros da ACO para o problema em questão.

Evidentemente, o processo de solução de um dado problema é iterativo, e à medida que se vai ganhando conhecimento sobre o mesmo, é provável que decisões iniciais venham a ser modificadas, tendo em vista as informações obtidas no processo. Antes de abordarmos os algoritmos inspirados no comportamento de uma colônia de formigas, convém introduzir, como exemplo de aplicação, o problema do caixeiro viajante.

O Problema do Caixeiro Viajante

No Problema do Caixeiro Viajante (PCV) (em inglês *Traveling Salesman Problem (TSP)*), um vendedor, partindo de uma cidade inicial, deseja percorrer o menor caminho para atender seus clientes nas cidades vizinhas, retornando por fim à cidade de onde ele partiu, visitando cada cidade uma única vez.

A representação do PCV é feita através de um grafo completamente conectado $G = (N, A)$, onde N é o conjunto de nós (vértices), que representam as cidades, e A é o conjunto de arestas que ligam todos os nós do grafo. Cada aresta $a_{ij} \in A$ possui um peso d_{ij} pré-determinado, representando a distância entre as cidades i e j, com $i, j \in N$. A Figura 5.4 apresenta um exemplo do PCV com 4 cidades, onde a distância entre as cidades é representada por uma matriz de custo e a função objetivo é descrita em (5.4). Neste exemplo, fixando uma cidade inicial podemos gerar 6 possíveis soluções, representadas por $s_1, ..., s_6$, e o valor ótimo é alcançado pelas soluções s_3 e s_5.

O PCV pode ser simétrico ou assimétrico, onde no primeiro caso a distância entre as cidades i e j é a mesma entre j e i, ou seja $d_{ij} = d_{ji}$, e no segundo a direção utilizada para percorrer os nós é levada em consideração, desta forma pode existir pelo menos uma aresta a_{ij} tal que $d_{ij} \neq d_{ji}$. O objetivo do PCV é encontrar o menor ciclo Hamiltoneano do grafo, onde este ciclo é um caminho fechado que visita cada um dos $n = |N|$ nós [3] do grafo G apenas uma vez.

A solução ótima do PCV é portanto uma permutação π dos nós de índice $\{1, 2, ..., n\}$, tal que o tamanho $f(\pi)$ seja mínimo. A formulação do problema pode ser escrita matematicamente como:

$$\begin{aligned} &\text{minimizar } f(\pi) = \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)} \\ &\text{sujeito a } \pi \in \Pi\{1, 2, ..., n\} \end{aligned} \tag{5.4}$$

[3] $|C|$, denota a cardinalidade do conjunto C, ou seja, o número de elementos do conjunto.

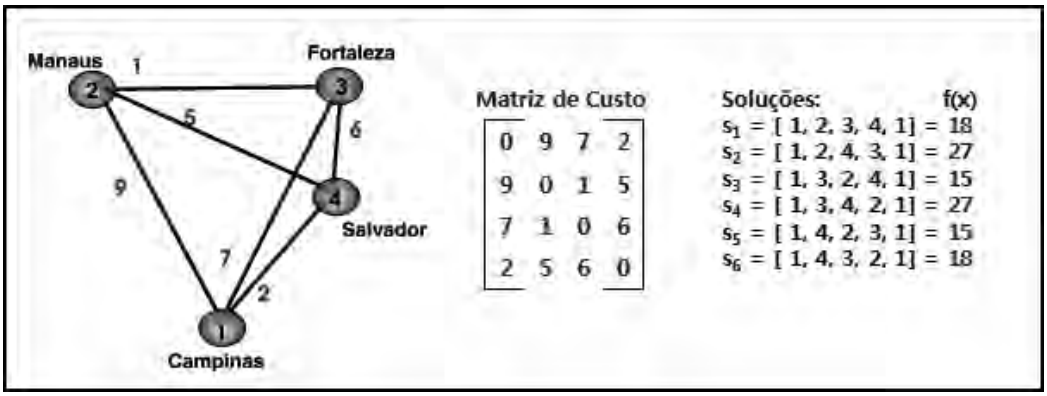


Figura 5.4: Exemplo de um PCV com 4 cidades. Figura retirada de Angelo (2008).

onde n é o número de cidades e $\Pi\{1, 2, .., n\}$ denota o conjunto de todas as possíveis permutações de $\{1, 2, .., n\}$.

ACO Aplicado ao PCV

A ACO pode ser aplicado ao PCV definindo o grafo $G = (N, A)$, onde o conjunto A conecta todos os componentes de N, que representa o conjunto de todas as cidades, e onde cada aresta tem um comprimento pré-determinado d_{ij}.

Os algoritmos ACO são essencialmente construtivos pois, a cada iteração, cada formiga (artificial), ao percorrer o grafo, constrói uma solução para o problema. Cada nó do grafo representa uma possível opção de caminho que a formiga pode percorrer. Este movimento está associado a duas formas de informação passadas para a formiga: (a) a trilha de ferômonio τ_{ij}, representada por uma matriz, associada à aresta a_{ij}, que corresponde à preferência da escolha do movimento para o nó j partindo do nó i; (b) a informação heurística, definida por $\eta_{ij} = 1/d_{ij}$, representada pela visibilidade do movimento da formiga seguir do nó i para j, que é inversamente proporcional à distância entre estas duas cidades.

O processo básico de construção da rota realizado por cada formiga é iniciado quando uma formiga artificial é posicionada, de acordo com algum critério, numa cidade inicial (ponto de partida). Em seguida, utilizando o feromônio e a informação heurística, de forma probabilística, ela constrói a rota através da inclusão sucessiva das cidades que ainda não foram visitadas. O processo termina quando a formiga, após visitar todas as cidades, retorna à cidade inicial. Depois que todas as formigas completaram seu caminho, elas adicionam uma quantidade de feromônio na rota percorrida. A estrutura de um algoritmo ACO é descrita a seguir.

Algoritmo 1 Pseudocódigo da Metaheurística ACO

```
1: Inicializar parâmetros
2: Inicializar matriz de feromônio
3: enquanto condições de parada não satisfeitas faça
4:     ConstruirSoluções
5:     AplicarBuscaLocal (opcional)
6:     AtualizarFeromônio
7: fim enquanto
```

Na rotina *ConstruirSoluções*, iniciando de um ponto de partida, as formigas se movem de acordo

com uma política de decisão que faz uso de informações locais relacionados aos nós visitados (trilha de feromônio e informação heurística). A cada passo uma componente de solução (cidade) é adicionada à rota até que todos os nós sejam visitados. Após a construção da solução e antes da atualização do feromônio, é comum a realização de busca local, através da rotina *AplicarBuscaLocal*, para melhorar a qualidade da solução encontrada. Ao final, a rotina *AtualizarFeromônio* realiza o processo de modificação da trilha (matriz) de feromônio, inspirado na deposição e/ou evaporação do feromônio nas trilhas de formigas reais.

Uma ideia muito útil para melhorar o desempenho da ACO no PCV seria a de restringir –de maneira adequada– a escolha da próxima cidade a ser visitada pela formiga. Ao invés de considerar *todas* as cidades ainda não visitadas, a formiga escolheria num subconjunto bem reduzido destas cidades.

Há pelo menos duas fortes razões para adotar listas de candidatas. Uma delas seria a redução no esforço computacional decorrente da redução do número de transições cujas probabilidades teriam que ser calculadas. É fácil ver que isto vai se tornando mais crítico à medida que o número de opções disponíveis cresce. Outra razão seria uma potencial aceleração da convergência do processo no sentido de que é de se esperar que a solução ótima (no PCV, por exemplo) conecte cada cidade sempre a alguma outra não muito distante dela. De certa forma, alguma informação heurística (a distância entre cidades vizinhas no PCV) passa a ser introduzida de maneira mais direta, além da maneira usual (probabilística) via a matriz η. Na realidade está se reduzindo drasticamente o grafo sobre o qual as formigas construirão suas rotas, focando-se a busca nas componentes mais promissoras do espaço de busca.

No caso do PCV, a informação heurística é conhecida de antemão e não se modifica durante a busca, podendo assim ser (pré-)computada uma única vez. Em outras situações, tal informação depende das componentes já presentes na solução parcial construída, e a lista de componentes candidatas deverá ser criada a cada passo.

4. Histórico dos Algoritmos ACO

Em 1991, foi desenvolvido o *Ant System* (AS), o algoritmo que deu início a metaheurística ACO. Inicialmente, o AS foi implementado como um algoritmo de otimização discreta para resolver o PCV (Dorigo et al., 1991; Dorigo, 1992).

O algoritmo AS passou por diversas modificações visando a melhoria dos resultados obtidos, a diminuição do esforço computacional e sua extensão para problemas multiobjetivo e de domínios contínuos. Os principais algoritmos ACO estão listados na Tabela 5.1 e uma breve comparação entre os algoritmos AS, RBAS, MMAS e ACS é apresentada na Tabela 5.2.

Algoritmo	Primeiras Referências
Ant System (AS)	Dorigo et al. (1991)
Elitist AS (EAS)	Dorigo (1992)
Ant-Q	Gambardella e Dorigo (1995)
Ant Colony System (ACS)	Dorigo e Gambardella (1997a,b)
Max-Min AS (MMAS)	Stützle e Hoos (1996)
Rank-Based AS (AS_{rank}/RBAS)	Bullnheimer et al. (1997, 1999)
ANTS	Maniezzo (1999); Maniezzo e Carbonaro (2000)
Best-Worst AS (BWAS)	Cordón et al. (2000)
Hyper-Cube AS (HCAS)	Blum et al. (2001)

Tabela 5.1: Principais variantes do algoritmo ACO. Baseado em Dorigo et al. (2006).

Assim como no algoritmo AS, os algoritmos RBAS e ACS inicializam a matriz de feromônio da seguinte forma:

$$\tau_{ij} = \tau_0 = \frac{k}{C^n}, \quad \forall(i,j) \tag{5.5}$$

onde τ_{ij} é a matriz de feromônio, k é o número de formigas e C^n está associada com a qualidade da solução encontrada de acordo com a heurística do problema.

No MMAS, cada valor da matriz de feromônio é limitada inferior e superiormente por τ_{min} e τ_{max} ($\tau_{min} \leq \tau_{ij} \leq \tau_{max}$) na tentativa de se evitar possíveis estagnações do algoritmo. Devido à evaporação do feromônio, o nível máximo possível para a trilha de feromônio é limitado por $\tau_{max} = 1/\rho C^*$, onde C^* é a qualidade da solução ótima de cada problema. Na prática, o MMAS usa $\tau_{max} = 1/\rho C^{bs}$, onde C^{bs} é a qualidade da melhor solução global. Desta forma, toda vez que a melhor solução global for atualizada, um novo valor de τ_{max} é definido. O valor do limite inferior é dado por $\tau_{min} = \tau_{max}/a$ onde a é um parâmetro que pode ser definido pelo usuário de acordo com o problema abordado. Maiores detalhes de como determinar o valor de τ_{min} podem ser encontrados em Stützle e Hoos (2000).

Outra importante alteração no algoritmo MMAS em relação ao AS ocorre na etapa de inicialização da matriz de feromônio. No MMAS o valor de τ_0 é estimado com base no limite superior τ_{max}. Se após algumas iterações não houver melhora na solução encontrada (ou após um número fixo de iterações), a matriz de feromônio pode ser reinicializada com base no valor de $\tau_{max} = 1/\rho C^{best}$ onde $C^{best} = C^{ib}$ representa a qualidade da melhor solução encontrado na última iteração, ou $C^{best} = C^{bs}$ representando a qualidade da melhor solução global.

Ant System (AS)	**AS$_{rank}$ ou RBAS**	**Max-Min AS (MMAS)**	**Ant Colony System (ACS)**

para cada colonia **faça**
  para cada formiga **faça**
  ConstruirSolução $^{(a)}$
  AplicarBuscaLocal $^{(c)}$
 	
  fim para
 	
  AtualizarFeromônio
fim para
	para cada colonia **faça**
	  para cada formiga **faça**
	  ConstruirSolução $^{(a)}$
	  AplicarBuscaLocal $^{(c)}$
	
	  fim para
	  OrdenarSoluções
	  AtualizarFeromônio
	fim para
		para cada colonia **faça**
		  para cada formiga **faça**
		  ConstruirSolução $^{(a)}$
		  AplicarBuscaLocal $^{(c)}$
		
		  fim para
		  OrdenarSoluções
		  AtualizarFeromônio
		fim para
			para cada colonia **faça**
			  para cada formiga **faça**
			  ConstruirSolução $^{(b)}$
			  AplicarBuscaLocal $^{(c)}$
			  AtualizarFeromônio
			  fim para
			  OrdenarSoluções
			  AtualizarFeromônio
			fim para

AtualizarFeromônio: 1. Evaporação: todas as arestas. $\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \forall (i,j) \in \mathcal{L},$ onde $0 < \rho \leq 1$ é a taxa de evaporação. 2. Deposição: arestas visitadas. $\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \forall (i,j) \in \mathcal{L},$ onde $\Delta\tau_{ij}^{k} = 1/C^{k}$ é a quantidade de feromônio depositada pela formiga k nas arestas que visitou.	*AtualizarFeromônio*: 1. Evaporação: idêntica ao proposto no algoritmo AS. 2. Deposição: realizada nas arestas das rotas das $w-1$ formigas melhor ordenadas (r) e nas arestas da melhor rota global (T^{bs}), sendo $\tau_{ij} \leftarrow \tau_{ij} + \sum_{r=1}^{w-1} (w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{bs},$ onde $\Delta\tau_{ij}^{r} = 1/C^{r}$ e $\Delta\tau_{ij}^{bs} = 1/C^{bs}$.	*AtualizarFeromônio*: 1. Evaporação: idêntica ao proposto no algoritmo AS. 2. Deposição: é dada por $\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best},$ onde $\Delta\tau_{ij}^{best} = \Delta\tau^{ib} = 1/C^{ib}$ representa a deposição nas arestas da melhor rota da iteração atual (T^{ib}) e $\Delta\tau_{ij}^{best} = \Delta\tau^{bs} = 1/C^{bs}$ é a deposição nas arestas da melhor rota global (T^{bs}). As duas formas podem ser usadas individualmente ou alternadamente dependendo do aspecto de convergência desejado.	*AtualizarFeromônio*: 1. Atualização Local: iterativamente assim que a aresta é visitada. $\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_{0},$ onde τ_0 possui os mesmos valores atribuídos na inicialização da matriz de feromônio. 2. Atualização Global: somente nas arestas pertencentes a melhor rota global T^{bs}. $\tau_{ij} \leftarrow (1-\rho_2)\tau_{ij} + \rho_2\Delta\tau_{ij}^{bs}, \forall (i,j) \in T^{bs},$ onde $0 < \rho_2 \leq 1$ é a taxa extra de evaporação e $\Delta\tau_{ij}^{bs} = 1/C^{bs}$.

(a) Nos algoritmos AS, RBAS e MMAS a rotina *ConstruirSolução* é dada por $p_{ij}^{k} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{l \in \mathcal{N}_i^k}(\tau_{il})^{\alpha}(\eta_{il})^{\beta}}$, se $j \in \mathcal{N}_i^k$, onde α e β são parâmetros que regulam a influência da matriz de feromônio τ_{ij} e da informação heurística local $\eta_{ij} = 1/d_{ij}$, e $\mathcal{N}_i^k$ é a vizinhança factível da formiga k situada em i.

(b) No algoritmo ACS a rotina *ConstruirSolução* é dada por $j = \begin{cases} \text{argmax}_{l \in \mathcal{N}_i^k} \left\{\tau_{il}[\eta_{il}]^{\beta}\right\}, & \text{se } q < q_0; \\ J, & \text{caso contrário;} \end{cases}$ onde $q \in [0,1]$ é uma variável aleatória uniformemente distribuída, $0 \leq q_0 \leq 1$ é um parâmetro de decisão escolhido pelo usuário e $J = p_{ij}^{k}$ é dado pela equação apresentada em (a).

(c) A rotina *AplicarBuscaLocal* é opcional em todos os algoritmos.

Tabela 5.2: Comparação entre os algoritmos *Ant System*, *Rank Based AS*, *Max-Min AS* e *Ant Colony System*. Baseado em Dorigo et al. (2006).

Por tratar-se de uma metaheurística, a ACO pode ser aplicado a um grande leque de problemas. Algumas aplicações dos algoritmos derivados da ACO estão listadas na Tabela 5.3.

Tipo	Problema
Roteamentos	PCV, Rotas de Veículos, Ordenamento Sequencial.
Atribuições	QAP, Tabela Horário, Coloração de Grafos.
Agendamentos	Projetos, Total de Atrasos, Abertura de Lojas.
Subgrupos	Número Máximo de Cliques, Cardinalidade de Árvores.
Biologia Computacional	Enovelamento Protéico, Docagem Ligante-Proteína, Multi-Alinhamento de Sequências.
Outros	Problemas Multiobjetivo, Criação de Música, Otimização Estrutural, Segmentação de Imagens.

Tabela 5.3: Exemplos de aplicações dos algoritmos baseados na ACO. Baseado em Blum (2005); Dorigo et al. (2006); Capriles et al. (2007)

5. ACO Aplicada a Problemas com Restrições

No problema do caixeiro viajante, uma formiga não pode partir da cidade atual e decidir mover-se para uma cidade já visitada. Ela não está completamente livre para decidir; há restrições. No caso, ela deve formar ao final uma rota (ciclo hamiltoneano) como solução candidata. Em outras palavras, o espaço de busca está restrito às permutações da lista de cidades $\{1, 2, \ldots, n\}$. O problema de otimização é, de fato, restrito. O leitor está convidado a ler, nesta obra, o capítulo 14, que é especialmente dedicado a este tema.

Mas, no caso do PCV, há uma maneira simples de fazer com que as formigas construam sempre uma rota válida. Basta equipar cada formiga com uma memória que lista todas as cidades já visitadas por ela. Assim, a sua escolha é feita somente entre as cidades ainda não visitadas.

Mas nem sempre as restrições permitem um tratamento tão simples. Para exemplificar uma situação que requer cuidados especiais, consideraremos aqui o problema do projeto estrutural de uma estrutura de barras – denominada treliça na engenharia estrutural – muito comum na prática.

O projeto estrutural ótimo busca produzir a melhor solução estrutural no sentido de encontrar uma relação satisfatória entre custo e desempenho. Entende-se por desempenho uma série de aspectos desejáveis de um bom projeto, que vão desde objetivos funcionais (como um peso mínimo, que possibilite maximizar a carga útil num dado veículo) à beleza, no caso de uma estrutura exposta que integra um dado conjunto arquitetônico.

O projetista buscará então um conjunto de opções e valores das chamadas variáveis de projeto que minimizem (ou maximizem) uma ou mais funções objetivo satisfazendo também um conjunto de restrições sobre tais variáveis e também sobre certas medidas de desempenho da estrutura.

Na prática, as áreas das seções transversais das barras que compõem estas estruturas, exemplificadas na Figura 5.5, são muitas vezes escolhidas dentre os tamanhos disponíveis no mercado, caracterizando assim um problema de otimização discreta, onde deseja-se encontrar o conjunto de áreas $\mathbf{a} = \{A_1, A_2, ..., A_n\}$ que minimize o peso da treliça

$$w(\mathbf{a}) = \sum_{k=1}^{N} \gamma A_k L_k \tag{5.6}$$

onde γ é o peso específico do material e L_k é o comprimento da k-ésima barra.

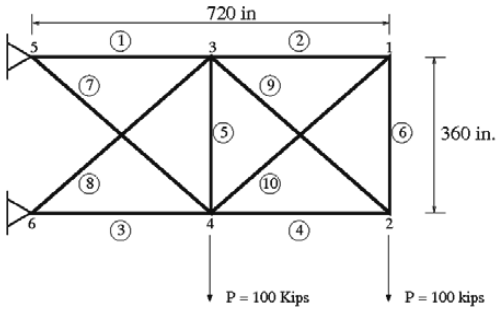
Para um desempenho aceitável, a treliça está sujeita a restrições de tensão normal nas barras (para evitar uma falha localizada)

$$\frac{|s_{j,l}|}{s_{adm}} - 1 \leq 0 \qquad 1 \leq j \leq N, \quad 1 \leq l \leq N_L \tag{5.7}$$

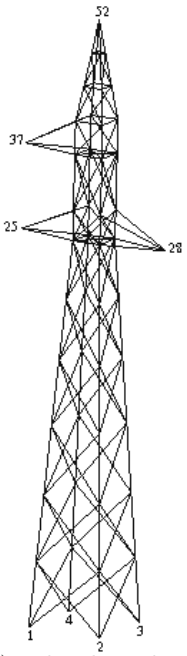
e de deslocamentos nos nós (para garantir que a sua geometria não se modifique significativamente)

$$\frac{|u_{i,l}|}{u_{adm}} - 1 \leq 0 \qquad 1 \leq i \leq M, \quad 1 \leq l \leq N_L \tag{5.8}$$

onde u_i é o deslocamento do i-ésimo grau de liberdade translacional, s_j é a tensão na j-ésima barra, s_{adm} é a tensão admissível, u_{adm} é o deslocamento máximo admissível, M é o número de graus de liberdade translacionais, N é o número total de barras e N_L é o número de casos de carregamento considerados.



(a) Treliça de 10 barras.



(b) Treliça de 160 barras.

Figura 5.5: Exemplo de treliças. (a) Estrutura de referência (*benchmark*) usada em estudos de otimização de peso de treliça. (b) Exemplo real de treliça usada como torre de transmissão. Figura retirada de Capriles et al. (2007).

Deve-se observar que as restrições acima são funções *implícitas* das variáveis de projeto **a**. De fato, é necessário resolver o problema estrutural, correspondente à solução da equação de equilíbrio discretizada

$$\mathbf{K(a)u}_l = \mathbf{f}_l \qquad 1 \leq l \leq N_L \tag{5.9}$$

onde $\mathbf{K}$ é a matriz de rigidez da estrutura construída a partir da contribuição matricial $K_j(\mathbf{a})$ de cada barra.

Para cada caso de carregamento o sistema é resolvido para o campo de deslocamentos u

$$\mathbf{u}_l = [\mathbf{K(a)}]^{-1}\,\mathbf{f}_l \tag{5.10}$$

e a tensão na j-ésima barra é calculada como $s_{j,l} = E\varepsilon(\mathbf{u}_l)$ onde E é o módulo de Young do material e ε é a deformação longitudinal unitária da barra.

Pode-se então verificar se as restrições (5.7) e (5.8) são satisfeitas para a treliça/solução candidata construída pela formiga.

Mas o que fazer se a treliça encontrada tem um bom (baixo) peso mas é inviável, isto é, viola uma ou mais das restriçoes (5.7) e (5.8)? Uma técnica simples consiste em *penalizar* tal solução: ela tem seu peso real penalizado (aumentado) via:

$$W(\mathbf{a}) = \quad w + \kappa \left(\sum_{l=1}^{N_L} \sum_{j=1}^{N} \left[\frac{|s_{j,l}|}{s_{max}} - 1 \right]_+^2 + \sum_{l=1}^{N_L} \sum_{i=1}^{M} \left[\frac{|u_{i,l}|}{u_{max}} - 1 \right]_+^2 \right) \tag{5.11}$$

onde κ é um coeficiente de penalização e $[x]_+ = x$, se $x > 0$, e 0, caso contrário.

Alternativamente, um esquema de penalização multiplicativa pode ser usado:

$$W(\mathbf{a}) = \quad w \left(1 + \sum_{l=1}^{N_L} \sum_{j=1}^{N} \left[\frac{|s_{j,l}|}{s_{max}} - 1 \right]_+ + \sum_{l=1}^{N_L} \sum_{i=1}^{M} \left[\frac{|u_{i,l}|}{u_{max}} - 1 \right]_+ \right)^{\epsilon} \tag{5.12}$$

onde $\epsilon \geq 1$ é um expoente de penalização. Uma técnica adaptativa que não requer do usuário a definição dos parâmetros de penalização pode ser encontrada em Lemonge e Barbosa (2004), mas o leitor deve consultar o capítulo 14 sobre tratamento de restrições desta obra para se familiarizar com as várias maneiras de se lidar com este problema.

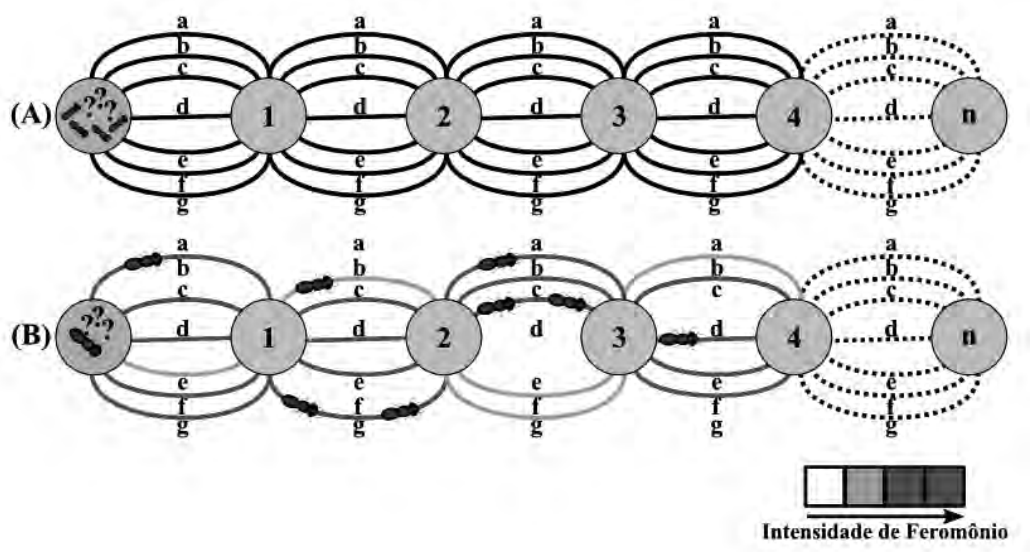


Figura 5.6: Esquema da representação e escolha das variáveis para a construção de uma rota em ACO para o problema de minimização de peso de treliças. **(A)** Para cada uma das n variáveis, cada formiga pode escolher um conjunto de áreas $A = \{\text{a,b,}\ldots\text{,g}\}$ com igual probabilidade. **(B)** Após certo tempo, algumas áreas terão maior probabilidade de serem escolhidas de acordo com a quantidade de feromônio associada.

A aplicação da ACO em problemas de otimização de peso de treliças pode ser observado em Capriles et al. (2007), onde para cada variável (barra) $k = 1 \ldots, N$, cada formiga pode escolher dentro de um conjunto de áreas A. Inicialmente, todas as áreas possuem igual probabilidade de escolha mas, após certo tempo, algumas áreas terão maior probabilidade de serem escolhidas de acordo com a quantidade de feromônio associada (esquema apresentado na Figura5.6).

Os autores fizeram testes em treliças com 10, 25, 52, 120 e 160 barras comparando variações (propostas pelos autores) do algoritmo RBAS, da seguinte forma: (i) variando os valores dos parâmetros do algoritmo RBAS (e RBAS$^+$) e considerando um esquema de penalização aditiva como apresentado na Equação 5.11; (ii) implementando uma etapa de atualização local da matriz de feromônio (RBAS$_{LU}$) e aplicando o esquema de penalização multiplicativa (Equação 5.12); (iii) modificando o algoritmo RBAS$_{LU}$ de forma a serem executadas duas fases de penalização multiplicativa e redução do espaço de busca (RBAS$_{LU,2}$). Os resultados destes testes mostraram que o algoritmo RBAS$_{LU,2}$ foi o que apresentou, na média, os melhores resultados. Os parâmetros adotados em cada variante estão descritos na Tabela 5.4.

Parâmetros	RBAS	RBAS$^+$	RBAS$_{LU}$	RBAS$_{LU,2}$
α	0,20	1,00	1,00	1,00
β	0,43	0,20	0,20	0,20
σ	0,60–1,10	0,67	0,60	0,60
σ_2	0,10–0,80	0,33	0,10	0,10
ρ	0,92	0,50	0,50	0,50
q_0	0,60–0,70	0,60	0,60	0,60
ξ	–	–	0,67	fase1: 0,67/fase2: 0,33
κ	10^7	10^7	–	–
ϵ	–	–	1,00	fase1: 1,00/fase2: 4,00
τ_0	0.50	0.50	$\frac{1}{w_{min}}$	$\frac{1}{w_{min}}$

Tabela 5.4: Parâmetros usados pelo *Rank Based Ant System* (RBAS), e suas variantes, em problemas de otimização de peso de treliças. α e β ponderam, respectivamente, as informações de feromônio e heurística, σ e σ_2 são as taxas de deposição de feromônio, ρ e ρ_2 são as taxas de evaporação de feromônio e q_0 é o parâmetro que controla a probabilidade de escolher uma determinada variável.

Nestes testes, o algoritmo RBAS inicializa a matriz de feromônio usando um valor $\tau_0 = 0.50$ em todas as entradas da matriz, enquanto no RBAS$_{LU}$ e RBAS$_{LU,2}$, a matriz de feromônio é inicializada com um valor constante dado por $\tau_0 = [w_{min}]^{-1}$, onde w_{min} é o peso da treliça quando a menor área é atribuída a cada um dos seus membros. Convém ressaltar que, desconsideradas as restrições impostas, este é o menor peso que a estrutura pode atingir.

No algoritmo RBAS$_{LU}$, quando o peso final da Equação 5.12 não se altera por um certo número consecutivo de ciclos, considera-se que a colônia de formigas convergiu para uma solução. Neste ponto completa-se a primeira fase. Para encorajar a colônia a buscar por melhores soluções em uma segunda fase, inicia-se um procedimento de busca local na vizinhança da melhor solução encontrada na primeira fase. O espaço de busca é reduzido limitando-se o número de seções transversais disponíveis na primeira fase. Esta limitação é feita da seguinte maneira: (i) para cada variável, o limite inferior do novo espaço de busca é a área transversal imediatamente menor àquela da melhor solução da primeira fase; (ii) da mesma forma, o limite superior é a área transversal imediatamente maior àquela da melhor solução da primeira fase. Esta redução do espaço de busca permite uma busca mais refinada na vizinhança e uma possibilidade de melhorar a solução encontrada. Adicionalmente, na segunda fase do algoritmo RBAS$_{LU,2}$ o coeficiente de atualização local ξ é reduzido promovendo uma busca mais vigorosa.

Na segunda fase, o parâmetro de penalização ϵ é aumentado, o que afeta diretamente o mecanismo de construção das rotas e a atualização da matriz de feromônio. Além disso, a matriz de feromônio na segunda fase é novamente inicializada com os valores de $\tau_0 = [w_{min}]^{-1}$.

6. ACO Aplicada a Problemas Multiobjetivo

Otimização Multiobjetivo

Os Problemas de Otimização Multiobjetivo (POM) buscam encontrar soluções ótimas que apresentem a melhor relação entre diversos objetivos, onde cada um precisa ser minimizado ou maximizado. A formulação matemática de um problema multiobjetivo, pode ser escrita da seguinte forma:

$$\begin{array}{l} \text{minimizar } \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})] \\ \text{sujeito a } \mathbf{x} = (x_1, x_2, ..., x_n) \in \mathcal{S} \end{array} \tag{5.13}$$

onde $k(\geq 2)$ é o número de funções objetivo, $\mathcal{S}$ é chamado conjunto ou região viável, $\mathbf{f}$ é o vetor das funções objetivo com $f_i : \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x}$ é o vetor de decisão, e se $\mathbf{x} \in \mathcal{S}$, então $\mathbf{x}$ é uma solução viável. O conjunto $\mathcal{Z}(= f(\mathcal{S}))$ corresponde ao conjunto imagem da região viável, e cada elemento de $\mathcal{Z}$, chamado de objetivo, é escrito como $\mathbf{z} = (z_1, z_2, ..., z_k)$, onde $z_i = f_i(\mathbf{x})$ para todo $i = 1, ..., k$, são chamados valores dos objetivos.

Em geral, os objetivos são conflitantes e não-comensuráveis, não sendo possível encontrar uma única solução que otimiza todos os objetivos. Lança-se mão então do conceito de otimalidade de Pareto (vide mais detalhes no capítulo 17).

Dado $\mathbf{x} \in \mathcal{S}$ um vetor de decisão qualquer, diz-se que $\mathbf{x}$ *domina* um outro vetor de decisão $\mathbf{x}' \in \mathcal{S}$, e escreve-se $\mathbf{x} \prec \mathbf{x}'$, se e somente se

$$f_i(\mathbf{x}) \leq f_i(\mathbf{x}') \quad \forall i \in \{1, ..., k\} \wedge \exists j \in \{1, ..., k\} : f_j(\mathbf{x}) < f_j(\mathbf{x}')$$

onde k é o número de objetivos. O vetor de decisão $\mathbf{x}$ é dito *não-dominado* com relação a um conjunto $\mathcal{S}' \subseteq \mathcal{S}$, se e somente se, não existir nenhum vetor em $\mathcal{S}'$ que domina $\mathbf{x}$. O vetor objetivo $\mathbf{z} \in \mathbb{Z}$ é dito não-dominado, se o seu vetor de decisão $\mathbf{x}$ correspondente é não-dominado. Um vetor de decisão $\mathbf{x}$ é um ótimo de Pareto, se e somente se, $\mathbf{x}$ é não-dominado com relação a $\mathbb{S}$. O vetor de objetivos $\mathbf{z} \in \mathbb{Z}$ é dito ótimo de Pareto, se o seu vetor de decisão $\mathbf{x}$ correspondente é um ótimo de Pareto. A melhoria de um vetor ótimo de Pareto em qualquer objetivo provoca a degradação em pelo menos algum outro objetivo.

O conjunto ótimo de Pareto contém todas as soluções ótimas de Pareto e o conjunto de vetores objetivos correspondente é denominado fronteira ótima de Pareto (Zitzler et al., 2000). A Figura 5.7 apresenta o exemplo de um problema de minimização com dois objetivos e duas variáveis de decisão.

O Problema do Caixeiro Viajante Multiobjetivo

O Problema do Caixeiro Viajante Multiobjetivo (PCVMO) pode ser apresentado da seguinte forma:

$$\begin{array}{l} \text{minimizar } \mathbf{f}(\pi) = \left\{ \begin{array}{l} f_1(\pi) = \sum_{i=1}^{n-1} d^1_{\pi(i)\pi(i+1)} + d^1_{\pi(n)\pi(1)} \\ f_2(\pi) = \sum_{i=1}^{n-1} d^2_{\pi(i)\pi(i+1)} + d^2_{\pi(n)\pi(1)} \\ \vdots \\ f_k(\pi) = \sum_{i=1}^{n-1} d^k_{\pi(i)\pi(i+1)} + d^k_{\pi(n)\pi(1)} \end{array} \right\} \\ \text{sujeito a } \pi \in \Pi\{1, 2, ..., n\} \end{array} \tag{5.14}$$

Em uma instância do PCV com k-objetivos, k grafos são utilizados, e cada um deles possui um custo diferente d^k_{ij} para a mesma aresta a_{ij}. O custo pode representar distância, tempo, gastos ou outro

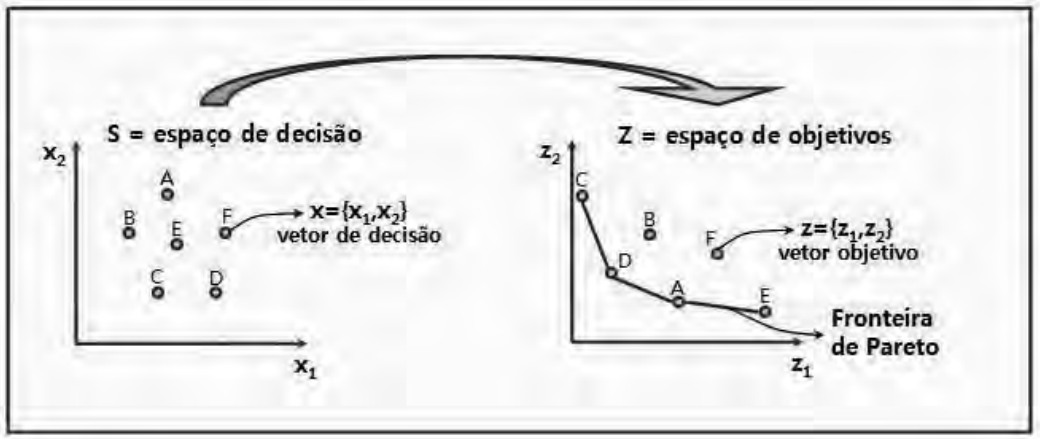


Figura 5.7: Espaço de decisão, espaço dos objetivos e a fronteira de Pareto em um problema de minimização com dois objetivos e duas variáveis de decisão, onde A, C, D e E são os vetores ótimos de Pareto ou o conjunto de soluções não-dominadas. Baseado em Zitzler et al. (2004).

parâmetro de interesse. A Figura 5.8 apresenta o mesmo exemplo da Figura 5.4, porém com dois objetivos, sendo agora as distâncias representadas por duas matrizes de custo, uma para cada objetivo e as funções objetivos são descritas em (5.14), com $k = 2$. Neste exemplo o valor ótimo no primeiro objetivo é alcançado pelas soluções s_3 e s_5 e no segundo pelas soluções s_1 e s_6.

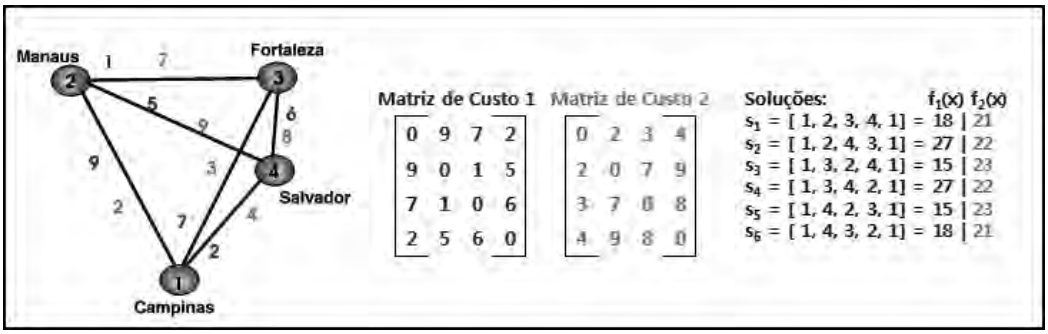


Figura 5.8: Exemplo de um PCV multiobjetivo com 4 cidades. Figura retirada de Angelo (2008).

Algoritmos ACO Multiobjetivo

García-Martínez et al. (2007) classificam os algoritmos ACO multiobjetivo (ACOMO) de acordo com a utilização de uma ou várias informações heurísticas, e a utilização de uma ou várias matrizes de feromônio, como exibido na Tabela 5.5. Em comparação aos algoritmos ACO mono-objetivo, os algoritmos ACO multiobjetivos apresentam algumas características semelhantes, porém adaptadas ao problema multiobjetivo, como por exemplo, na construção da solução e na atualização de feromônio. Para aqueles que constroem o conjunto de soluções ótimas de Pareto, esta etapa precisa ser adicionada ao algoritmo cuja estrutura básica pode ser escrita como

A seguir, os algoritmos MACS, MONACO e BicriterionAnt serão detalhados como exemplos da ACO para o caso multiobjetivo:

	Uma informação heurística	**Várias informações heurísticas**
Uma matriz de feromônio	MOACOM, MOACO-ALBP, ACOAMO.	MOAQ, MACS.
Várias matrizes de feromônio	P-ACO, MONACO.	BicriterionAnt, UnsortBicriterion, BicriterionMC, COMPETants, MACS-VRPTW, MO-PACO.

Tabela 5.5: Classificação dos algoritmos ACO multiobjetivo. Baseado em García-Martínez et al. (2007).

Algoritmo 2 Pseudocódigo da Metaheurística ACO Multiobjetivo (ACOMO)

1: Inicializar parâmetros
2: Inicializar matriz de feromônio
3: **enquanto** condições de parada não satisfeitas **faça**
4: *ConstruirSolução*
5: *AplicarBuscaLocal* (opcional)
6: *AtualizarConjuntoPareto*
7: *AtualizarFeromônio*
8: **fim enquanto**

Multiple Ant Colony System (MACS): Barán e Schaerer (2003) desenvolveram o algoritmo MACS, baseado no ACS, para a resolução de um problema de roteamento de veículos. No MACS, apenas uma colônia τ é utilizada para a geração do conjunto de Pareto, considerando três objetivos: número de veículos, tempo total da viagem e tempo total de entrega. Inicialmente, duas informações heurísticas são utilizadas, η_{ij}^0 e η_{ij}^1, referentes ao tempo total da viagem e tempo total de entrega, respectivamente. A escolha do próximo passo é feita através da seguinte regra:

$$j = \begin{cases} \arg\max_{j \in \mathcal{N}_i^h} \{\tau_{ij}[\eta_{ij}^0]^{\lambda\beta}[\eta_{ij}^1]^{(1-\lambda)\beta}\}, & \text{se } q \leq q_0 \\ \hat{i} & \text{caso contrário;} \end{cases} \tag{5.15}$$

onde β é o parâmetro que regula a influência da trilha de feromônio, λ é calculado para cada formiga h onde $\lambda = h/m$, m representa o número total de formigas e $\hat{i}$ é dado pela seguinte regra de decisão probabilística:

$$p_{ij}^h = \frac{\tau_{ij}[\eta_{ij}^0]^{\lambda\beta}[\eta_{ij}^1]^{(1-\lambda)\beta}}{\sum_{l \in \mathcal{N}_i^h} \tau_{il}[\eta_{il}^0]^{\lambda\beta}[\eta_{il}^1]^{(1-\lambda)\beta}}, \quad \text{se } j \in \mathcal{N}_i^h \tag{5.16}$$

A cada passo dado por uma formiga, a seguinte regra de atualização local de feromônio é aplicada

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0 \tag{5.17}$$

Inicialmente τ_0 é calculado como $1/\overline{f^0}.\overline{f^1}$ usando-se a média de cada uma das funções objetivo, f^0 e f^1. Após a construção de cada solução, esta é comparada com as soluções armazenadas no conjunto de Pareto. A solução que for não-dominada é incluída e a dominada é excluída do conjunto de Pareto. Ao final de cada iteração, τ_0 é recalculado de acordo com a expressão já vista. Caso $\tau_0' > \tau_0$, então a trilha de feromônio é reinicializada para o novo valor $\tau_0 \leftarrow \tau_0'$, caso contrário, a atualização global de feromônio é realizada, utilizando no cálculo cada solução S armazenada no conjunto de Pareto, de forma que

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \frac{\rho}{f^0(S)f^1(S)} \tag{5.18}$$

Multi-objective Network ACO (MONACO): O algoritmo MONACO foi apresentado por Cardoso et al. (2003), para resolver o problema dinâmico de otimização multiobjetivo do tráfego de mensagens um uma rede. Apresentaremos uma adaptação do MONACO para o problema estático, onde a disposição dos nós na rede não é modificada.

Este algoritmo utiliza uma informação heurística $\eta_{ij} = \sum_{k=1}^{K} d_{ij}^k$ e várias matrizes de ferômonio τ^k em sua formulação, onde K é o número de objetivos. Cada formiga, considerada como uma messagem, escolhe o próximo nó da rede de acordo com a seguinte regra de decisão probabilística

$$p_{ij}^h = \frac{\eta_{ij}^{\beta} \prod_{k=1}^{K} [\tau_{ij}^k]^{\alpha_k}}{\sum_{l \in \mathcal{N}_i^h} \eta_{il}^{\beta} \prod_{k=1}^{K} [\tau_{il}^k]^{\alpha_k}} \tag{5.19}$$

Nesta equação, os parâmetros α_k e β representam a importância relativa as matrizes de feromônio e a informação heurística, respectivamente. Ao final de cada iteração, as formigas que construiram sua solução, atualizam a trilha de feromônio da seguinte forma

$$\tau_{ij}^k = (1 - \rho_k)\tau_{ij}^k + \Delta\tau_{ij}^k \quad \text{com} \quad \Delta\tau_{ij}^k = Q/f^k(s_h) \tag{5.20}$$

onde ρ_k são os parâmetros relativos à taxa de evaporação para cada objetivo k, Q é uma constante relativa à quantidade de feromônio liberada pelas formigas e s_h a solução construída pela formiga h. Nesta adaptação do MONACO, as soluções não-dominadas, que formam o conjunto de Pareto, são armazenadas em um arquivo externo.

Ant Algorithm for Bicriterion Optimization (BicriterionAnt): Iredi et al. (2001) desenvolveram alguns métodos ACO para o problema de minimização do tempo total de atraso numa única máquina incluindo custos de conversão (*Single machine total tardiness problem with change over cost*). Nesta seção, apenas um dos algoritmos propostos por Irede *et al* será apresentado, aquele que considera apenas uma colônia de formigas. Este algoritmo, chamado BicriterionAnt, utiliza duas matrizes de feromônio τ e τ' e duas informações heurísticas η e η', referentes a cada um dos objetivos.

A cada iteração, cada uma das m formigas da colônia constrói uma solução, e a escolha do próximo passo utiliza a seguinte regra de decisão probabilística

$$p_{ij}^h = \frac{\tau_{ij}^{\lambda\alpha} \tau_{ij}'^{(1-\lambda)\alpha} \eta_{ij}^{\lambda\beta} \eta_{ij}'^{(1-\lambda)\beta}}{\sum_{l \in \mathcal{N}_i^h} \tau_{il}^{\lambda\alpha} \tau_{il}'^{(1-\lambda)\alpha} \eta_{il}^{\lambda\beta} \eta_{il}'^{(1-\lambda)\beta}}, \quad \text{se } j \in \mathcal{N}_i^h \tag{5.21}$$

onde α e β são os parâmetros que regulam a influência de τ e η, respectivamente, e $\mathcal{N}_i^h$ são os vizinhos viáveis (cidades ainda não visitadas) da formiga h. Para forçar que cada formiga busque por soluções em diferentes regiões na fronteira de Pareto, λ é calculado para cada formiga $h \in \{1, ..., m\}$, via

$$\lambda_h = (h - 1)/(m - 1) \tag{5.22}$$

Assim, nos casos extremos, a formiga 1, com $\lambda = 0$, considera apenas o primeiro objetivo e a formiga m, com $\lambda = 1$, considera apenas o segundo.

Após a construção das soluções, a trilha de feromônio é evaporada, aplicando a seguinte regra em todas as arestas a_{ij}

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} \quad , \quad \tau'_{ij} = (1 - \rho)\tau'_{ij} \tag{5.23}$$

onde $\rho \in (0, 1]$ representa a taxa de evaporação do feromônio.

Em seguida, as soluções não-dominadas na atual iteração são armazenadas no conjunto de Pareto e apenas estas podem atualizar a matriz de feromônio, acrescentando uma quantidade de $\tau_{ij} \leftarrow \tau_{ij} + 1/l$ na trilha, onde l é a quantidade de formigas que podem atualizar o feromônio na atual iteração.

7. ACO Aplicada a Problemas com Variáveis Contínuas

A extensão das ideias da ACO a problemas de otimização contínua exige diversas modificações, sendo poucos os trabalhos disponíveis na literatura, alguns deles brevemente comentados a seguir. Observa-se que alguns algoritmos são apenas inspirados em ACO, sendo conceitualmente diferentes do ACO original aplicado a problemas discretos. Dentre estes, as principais classes são *simulação direta* e *discretização do domínio*. Recentemente alguns artigos foram publicados focando nos conceitos principais de um algoritmo ACO, sendo classificados como *amostra probabilística*. A seguir, será apresentada uma breve descrição dos principais algoritmos destas classes.

Simulação Direta

Continuous ACO (CACO):

Foi o primeiro algoritmo proposto por Bilchev e Parmee (1995). Este algoritmo estabelece uma vizinhança ao ninho através de representação por finitos vetores direção. O procedimento de busca local é realizado usando estes vetores e usa algoritmos genéticos para executar a busca global (busca por diferentes ninhos).

Pachycondyla API*calis* (API):

Assim como no CACO, no algoritmo API (Monmarché et al., 2000) as formigas movem-se na vizinhança do ninho. Cerca de 20-30% das formigas exploram esta vizinhança de acordo com uma amplitude, enquanto as demais formigas exploram outras regiões aleatoriamente.

Continuous Interacting Ant Colony (CIAC): Proposto por Dréo e Siarry (2004), no algoritmo CIAC a interação entre os agentes ocorre de forma heterárquica onde um indivíduo influencia os demais por fluxos de informação. Esta comunicação considera o número de indivíduos em cada lado do canal de comunicação que é dividida em duas partes: (a) Trilha: distribuição pontual do feromônio. As formigas movem-se para o centro de gravidade destes pontos (semelhante ao algoritmo PSO). (b) Inter-indivíduo: comunicação P2P. Se a mensagem incluir melhora na função objetivo, o receptor migra para a região.

Discretização do Domínio

Adaptive Ant Colony Algorithm (AACA): No algoritmo AACA (Li e Wu, 2003) as soluções candidatas são representadas por vetores binários. Após completar a rota, este vetor é convertido e a atualização dos valores de feromônio é feita por um método de auto-ajuste, baseado nos valores da função objetivo.

Amostra Probabilística

Extended ACO (EACO ou ACO*): O algoritmo EACO, proposto por Socha (2004), é uma extensão direta do ACO estruturado para trabalhar com problemas mistos. As formigas usam função densidade de probabilidade (FDP) para uma escolha probabilística dos componentes da solução. O feromônio é atualizado por modificações nas FDP[4] Entretanto, por possuir somente um máximo, uma única FDP normal não é capaz de descrever uma situação onde duas áreas disjuntas em um espaço de busca são promissoras. Para resolver este problema, é usado um conjunto de FDP normais, sendo cada FDP deste conjunto definida por

$$P_{jk}(x_j) = g(x_j, \mu_k, \sigma_k) = \frac{1}{\sigma_k\sqrt{2\pi}} e^{-\frac{(x_j-\mu_k)^2}{2\sigma_k^2}} \tag{5.24}$$

[4] Principais Características da FDP: A função normal (gaussiana) é uma das funções mais aplicadas como FDP para estimar distribuições. É definida por média e desvio padrão e permite a geração de números aleatórios a partir de uma distribuição uniforme.

onde μ_k é a média e σ_k é o desvio padrão da k-ésima FDP.
Continuous Ant Colony System (CACS): É uma extensão direta da variante ACS. Assim como na EACO, no algoritmo CACS (Pourtakdoust e Nobahari, 2004) as formigas usam FDP normal para a construção de soluções candidatas. Por não usar um conjunto de FDP, não é capaz de trabalhar com múltiplos mínimos. Porém foi estruturado para trabalhar com um menor número de parâmetros.
Aggregation Pheromone System (APS): Considerado uma generalização do EACO, o APS (Tsutsui, 2004) usa FDP para agregar feromônio ao conjunto solução. As soluções são ordenadas e a os valores de feromônio são atualizados em uma elite através de uma FDP. O total de feromônio emitido pela elite é igual a um valor constante. Assim como no EACO, o APS trabalha com a decomposição de sub-FDP, porém, através de uma abordagem mais complexa (via decomposição de Cholesky).
Direct ACO (DACO): Proposto por Kong e Tian (2006), a DACO usa dois tipos de feromônio, um para a média e outro para o desvio padrão. Estes valores são usados para construir as soluções candidatas sendo cada variável do problema é associada a uma respectiva FDP. A atualização é feita considerando-se distintamente a média μ e o desvio padrão σ. Sendo a evaporação dada por

$$\vec{\mu}(t) = (1-\rho)\vec{\mu}(t-1) \qquad \vec{\sigma}(t) = (1-\rho)\vec{\sigma}(t-1) \tag{5.25}$$

onde $\rho \in [0,1]$ a taxa de evaporação, e a intensificação é dada por

$$\vec{\mu}(t) = \vec{\mu}(t) + \vec{\rho} \qquad \vec{\sigma}(t) = \vec{\sigma}(t) + \vec{\rho}|\vec{x} - \vec{\mu}(t-1)| \tag{5.26}$$

onde $\vec{x}$ é a melhor solução global.
Continuous RBAS (CRBAS):

Proposto por Capriles et al. (2006), o algoritmo CRBAS é uma extensão direta da variante *Rank Based Ant System* (RBAS) aplicado a problemas de minimização de peso de treliças. Segue a implementação e metodologia proposta pelo EACO, sendo a construção de soluções candidatas baseada em conjuntos de FDP. O processo de escolha de FDP e atualização do feromônio seguem o mesmo mecanismo apresentado pelo RBAS original, aplicado a problemas discretos.

8. Conclusões

Este capítulo apresentou a metaheurística ACO, inspirada no comportamento de uma colônia de formigas, originalmente proposta para problemas formulados em grafos. Discutiu-se sua extensão aos casos de múltiplos objetivos, problemas com restrições e variáveis contínuas. É importante lembrar que a ACO continua em desenvolvimento e sua aplicação em outras situações, como otimização em dois níveis e programação automática (*ant programming*), pode ser encontrada na literatura, bem como sua implementação em arquiteturas paralelas e distribuídas.