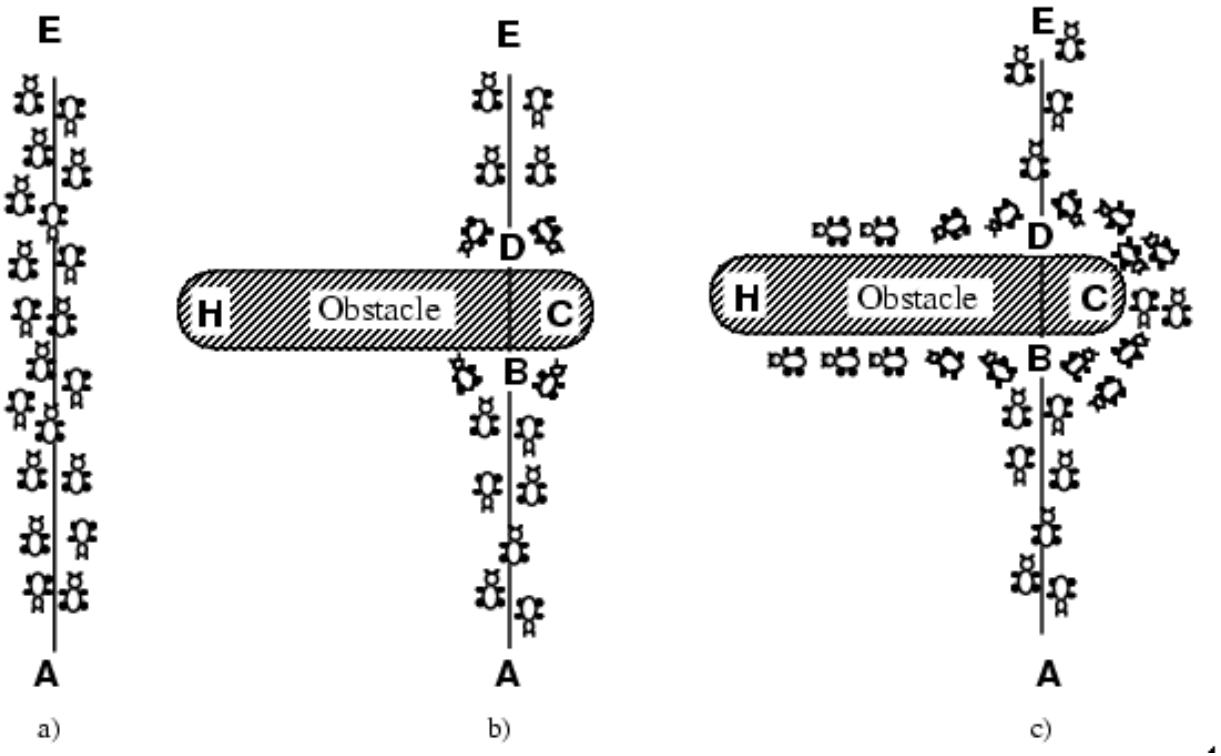
Ant-System: Algoritmos de otimização baseado em colônias de formigas artificiais

Prof. Ademir A. Constantino
Departamento de Informática
Universidade Estadual de Maringá
www.din.uem.br/~ademir

Ant System:

Inspiração nas formigas naturais



a) as formigas seguem um caminho entre A e E.

b) um obstáculo é colocado, então as formigas escolhem um dos dois caminhos com probabilidades iguais.

c) a intensidade de feromônio no menor caminho será maior.

Ant System:

Inspiração nas formigas naturais

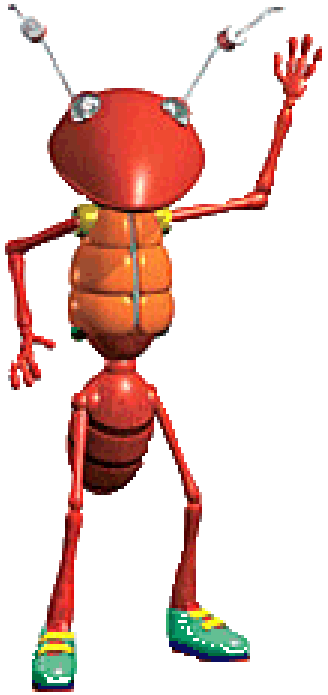
- *Ant System*: nome dado ao paradigma computacional que faz uma analogia com o comportamento de colônias de formigas naturais.

- Algoritmos baseados em *Ant System*, denominados de *Ant Algorithms*, simulam colônias de formigas usando colônias de formigas artificiais como ferramenta de otimização.

- Diferenças com as formigas reais:
  * As formigas artificiais tem alguma memória;
  * Elas não são completamente cegas;
  * Elas vivem num ambiente onde o tempo é discreto;
  * O "feromônio" depositado no caminho é representado por um número.
  * O feromônio é depositado somente depois de ter completado o caminho.

Ant System:
Inspiração nas formigas naturais

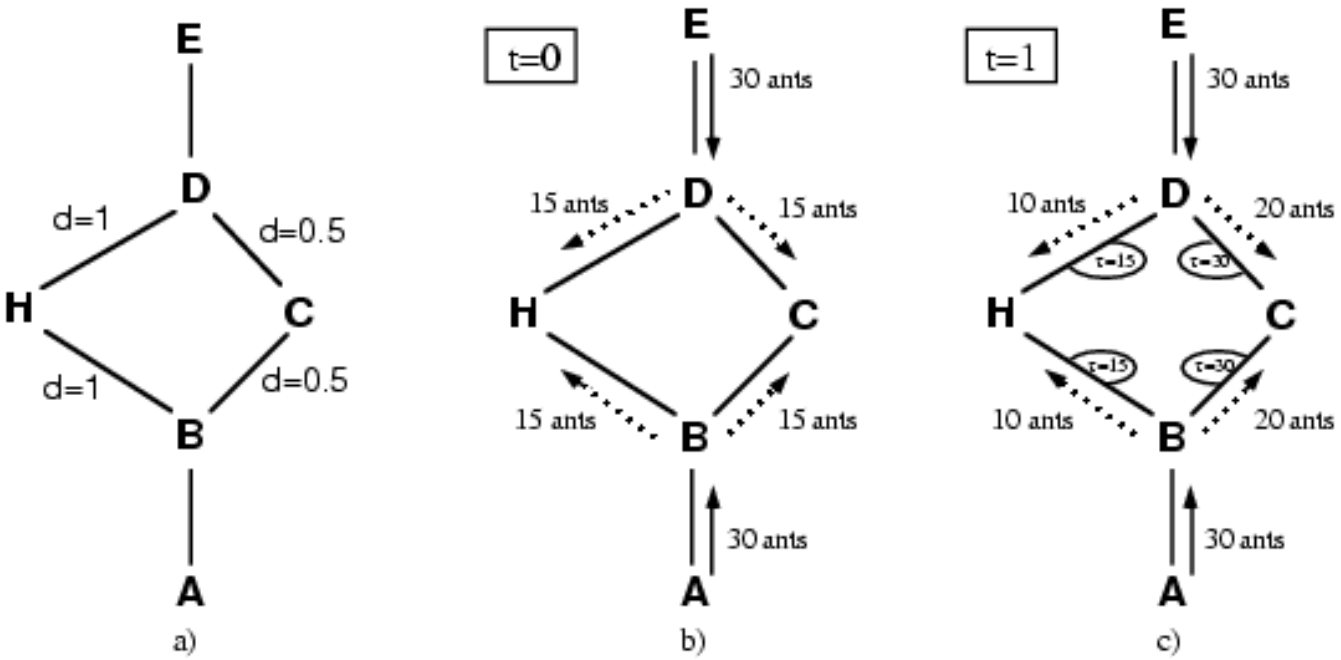
- Tempo discretizado
- Memória
- Feromônio é um número
- Feromônio é atualizado a posteriori.
- Qualidade da Solução
- Visão (distância)



Formiga REAL

Formiga ARTIFICIAL

Ant System:
Exemplo com formigas artificias



a) O grafo com as distâncias.

b) No tempo t=0 não existe "feromônio" nas arestas, então as formigas escolhem um dos dois caminhos com probabilidades iguais.

c) No tempo t=1 o "feromônio" τ é maior nas aresta menores, portanto, esta arestas são mais preferidas pelas formigas.

Princípios

- *Ant System* surgiu do trabalho de Dorigo (1996). Um algoritmo baseado em *Ant System* é composto por vários agentes (chamados de formigas artificiais) que se interagem na troca de informações com o objetivo de resolver o problema.
- *Ant System* foi inspirado em colônias reais de formigas as quais depositam uma substância química (chamada de feromônio) no chão para guiar as demais formigas no caminho entre o ninho e a fonte de alimento. Essa substância influencia a escolha: se há uma grande quantidade de feromônio sobre um particular caminho, então haverá uma grande probabilidade de que uma formiga selecionar esse caminho.

Princípios (cont)

- O feromônio depositado pelas formigas se evapora com o tempo, portanto, o caminho menor entre o ninho e a comida terá uma concentração maior de feromônio porque as formigas poderão trafegar mais rapidamente.
- Mais tarde sugiu a meta-heurística ACO (*Ant Colony Optimization technique*) como um esquema para unificar a maioria das aplicações de algoritmos baseados em formigas para problemas de otimização combinatória.

Princípios (cont)

- É um paradigma construtivo e de melhoramento simultaneamente.
- O algoritmo é baseado em população: um conjunto de soluções são construídas simultaneamente;
- Possui comportamento auto-catalítico: trata-se de um processo que auto-reforço que causa convergência muito rápida.

Idéia geral do algoritmo para o PCV

- Cada formiga executa as seguintes tarefas:
- a) Em cada iteração t uma formiga escolhe um vértice (cidade) ainda não visitado usando uma regra probabilística;
- b) A probabilidade de escolha de uma aresta (caminho) (i,j) é uma composição da quantidade feromônio τ_{ij} na aresta (i,j) com o inverso do custo da aresta d_{ij} (distância entre os vértices i e j);
- c) **Depois** da formiga ter completado o caminho, ela deposita uma quantidade $\Delta\tau_{ij}$ de feromônio em cada aresta (i,j) do caminho;
- e) Para a evaporação do feromônio em cada aresta (i,j) é considerado uma taxa de evaporação $\rho<1$: $\tau_{ij} = \rho\ \tau_{ij} + \Delta\tau_{ij.}$

O Algoritmo para o PCV

- Considere um conjunto de n cidades, o objetivo PCV é encontrar o menor caminho (*tour*) passando por cada cidade pelo menos uma vez e retornando à cidade de origem.

- Seja $b_i(t)$ (i=1, 2, ..., n) o número de formigas na cidade i no tempo t, portanto, $m = \sum_{i=1}^{n} b_i(t)$, é o número total de formigas.

- Depois de uma formiga ter encontrado um *tour*, uma substância denominada de **resíduo** (feromônio) é depositada nas arestas visitadas. A quantidade de resíduo na aresta (i, j) no tempo t é representada por $\tau_{ij}(t)$.

O Algoritmo para o PCV

- Cada *iteração* é definida por $(t+n)$. A cada iteração a intensidade de resíduo na aresta é atualizado por:

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij} \qquad (1)$$

sendo

ρ um coeficiente de persistência tal que $(1-\rho)$ representa a taxa de evaporação do resíduo entre os tempos t e $t+n$.

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (2)$$

é a quantidade da substância a ser depositada na aresta (i,j) pela k-ésima formiga .

O Algoritmo para o PCV

- Quantidade de resíduo depositado na aresta (modelo "*ant-cycle*").

$$\Delta \tau_{ij}^{k} = \begin{cases} Q / L_k \text{ se a k - ésima formiga usou a aresta } (i, j) \\ \qquad \text{em seu tour (entre o tempo } t \text{ e } t + n). \\ 0 \text{ caso contrário} \end{cases} \qquad (3)$$

onde Q é uma constante e L_k é o comprimento do tour da k-ésima formiga.

O Algoritmo para o PCV

- Os vértices visitados pela *k*-ésima formiga são colocados em uma lista **tabu**$_k$. Define-se por *visibilidade* $\eta_{ij} = 1/d_{ij}$. Assim, a probabilidade de transição de uma cidade *i* para a cidade *j* pela *k*-ésima formiga é definida como:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in allowed_k} [\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}} & \text{se } j \in allowed_k \\ 0 & \text{caso contrário} \end{cases} \qquad (4)$$

sendo $allowed_k$ o conjunto de cidades que ainda não foram visitadas pela *k*-ésima formiga e $\alpha\beta$ significam a importância relativa do resíduo e a atratividade.

Algoritmo: Resumo

- O algoritmo é construtivo
- Inicialização: t=0 e cada formiga é posicionada em uma cidade diferente.
- Em todo tempo t cada formiga escolhe probabilisticamente a próxima cidade que ela estará no tempo t+1;
- Uma *iteração* do algoritmo corresponde a m movimentos realizados pelas m formigas no intervalo (t, t+1);
- Em n iterações (denominado de *ciclo*) cada formiga terá completado um *tour*.

O algoritmo

1. Initialize:
   Set t:=0 {t is the time counter}
   Set NC:=0 {NC is the cycles counter}
   For every edge (i,j) set an initial value $\tau_{ij}(t)=c$ for trail intensity and $\Delta\tau_{ij}=0$
   Place the m ants on the n nodes

2. Set s:=1 {s is the tabu list index}
   For k:=1 to m do
   Place the starting town of the k-th ant in $\mathbf{tabu}_k(s)$

3. Repeat until tabu list is full {this step will be repeated (n-1) times}
   Set s:=s+1
   For k:=1 to m do
   Choose the town j to move to, with probability $p_{ij}^k(t)$ given by equation (4)
   {at time t the k-th ant is on town i=$\mathbf{tabu}_k(s-1)$}
   Move the k-th ant to the town j
   Insert town j in $\mathbf{tabu}_k(s)$

O algoritmo (cont.)

4. For k:=1 to m do
   Move the k-th ant from $\mathbf{tabu_k}(n)$ to $\mathbf{tabu_k}(1)$
   Compute the length L_k of the tour described by the k-th ant
   Update the shortest tour found
   For every edge (i,j)
   For k:=1 to m do

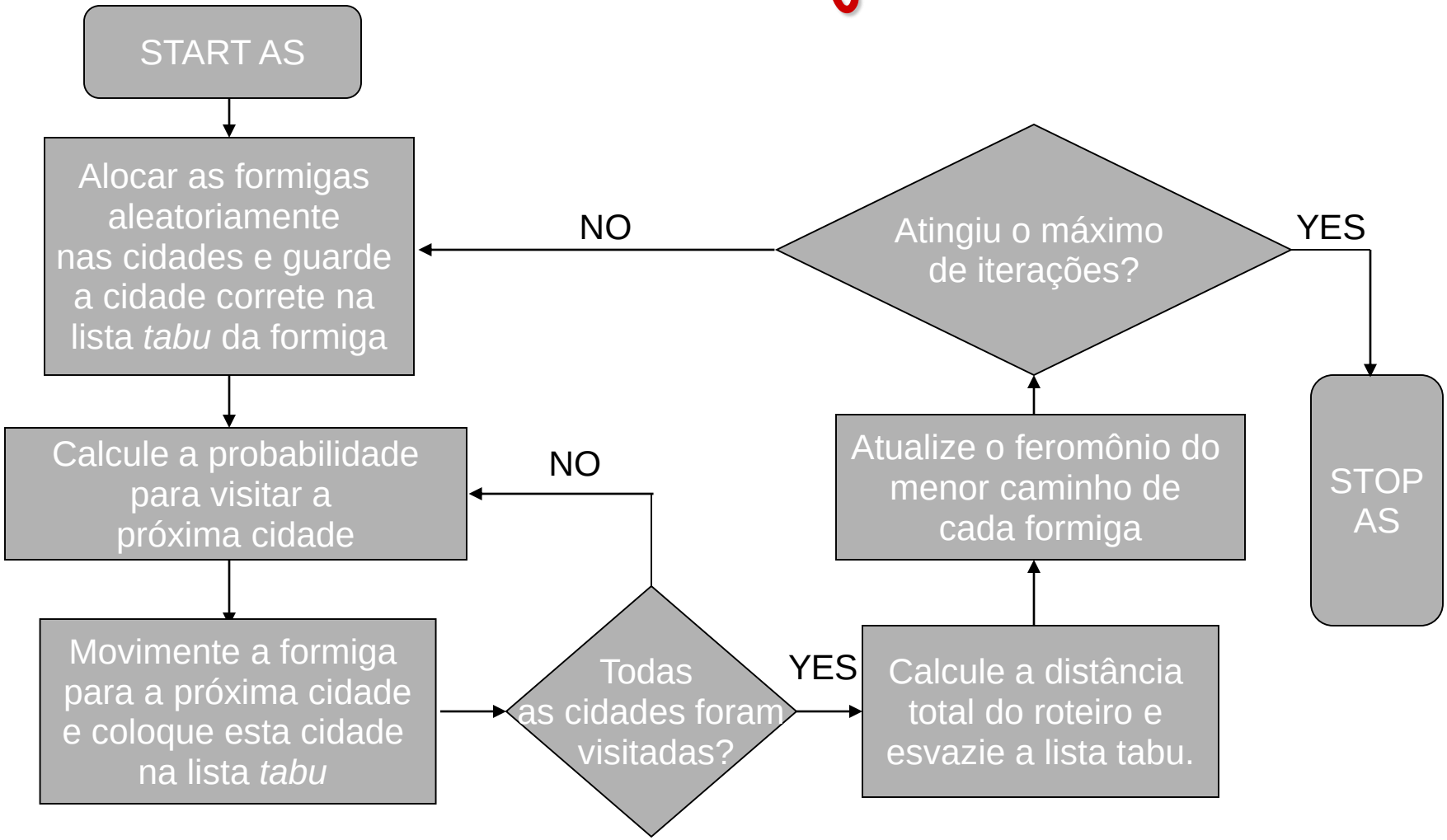
$$\Delta\tau_{i,j}^{k} = \begin{cases} \frac{Q}{L_k} & \text{if } (i,j) \in \text{tour described by } \mathbf{tabu_k} \\ 0 & \text{otherwise} \end{cases}$$

$$\Delta\tau_{ij} := \Delta\tau_{ij} + \Delta\tau_{ij}^{k};$$

5. For every edge (i,j) compute $\tau_{ij}(t+n)$ according to equation $\tau_{ij}(t+n)=\rho \cdot \tau_{ij}(t)+\Delta\tau_{ij}$
   Set t:=t+n
   Set NC:=NC+1
   For every edge (i,j) set $\Delta\tau_{ij}:=0$

6. If (NC < NC_{MAX}) and (not stagnation behavior)
   then
   Empty all tabu lists
   Goto step 2
   else
   Print shortest tour
   Stop

Fluxograma do AS para o Prob. Caixeiro Viajante



Outras Propostas para o Resíduo

Modelo "*ant-density*"

$$\Delta\tau_{ij}^{k} = \begin{cases} Q & \text{se a-ésima formiga usou a aresta } (i,j) \\ & \text{em seu tour (entre o tempo } t \text{ e } t+n). \\ 0 & \text{caso contrário} \end{cases}$$

Modelo "*ant-quantity*"

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/d_{ij} & \text{se a-ésima formiga usou a aresta } (i,j) \\ & \text{em seu tour (entre o tempo } t \text{ e } t+n). \\ 0 & \text{caso contrário} \end{cases}$$

Parâmetros testados por Dorigo (1996)

α	β	ρ	Q
0,5	1	0,3	1
1	2	0,5	100
2	5	0,7	10.000
5		0,9	

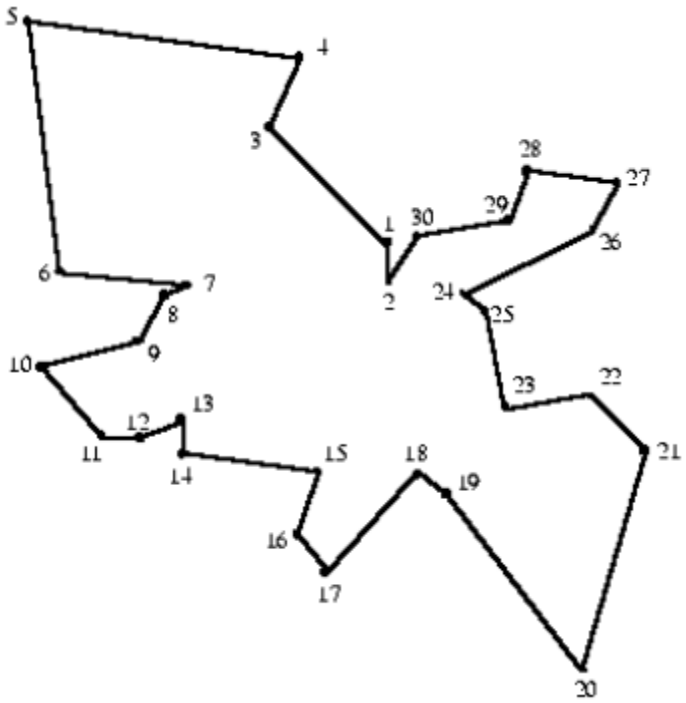
Resultados

Problema com 30 cidades (*Oliver30*)

Table I. Comparison among ant-quantity, ant-density, and ant-cycle. Averages over 10 trials.

	Best parameter set	Average result	Best result
ant-density	$\alpha=1$, $\beta=5$, $\rho=0.99$	426.740	424.635
ant-quantity	$\alpha=1$, $\beta=5$, $\rho=0.99$	427.315	426.255
ant-cycle	$\alpha=1$, $\beta=5$, $\rho=0.5$	424.250	423.741

Resultados



a) O grafo com 30 vértices (*Oliver30*).

b)α=1 β=5 ρ=0.5, Q=100.

c) Melhor solução obtida com 342 ciclos (iterações).

Resultados

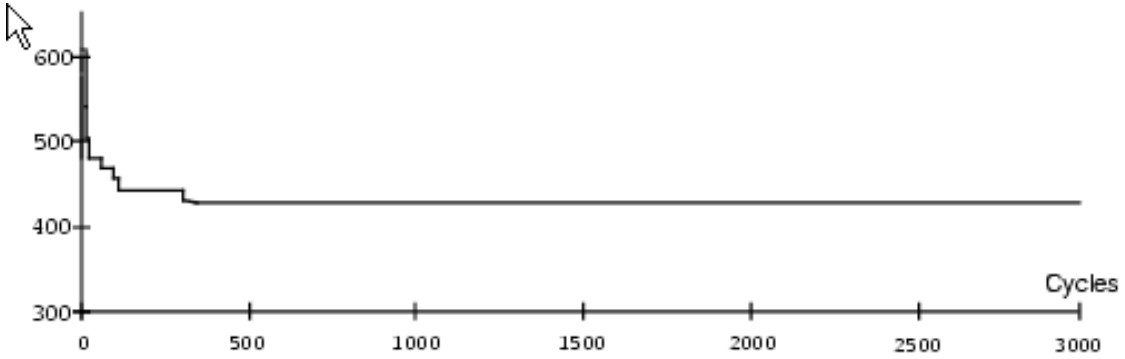


Fig. 3. Evolution of best tour length (Oliver30). Typical run.

Melhor solução encontrada com 342 ciclos

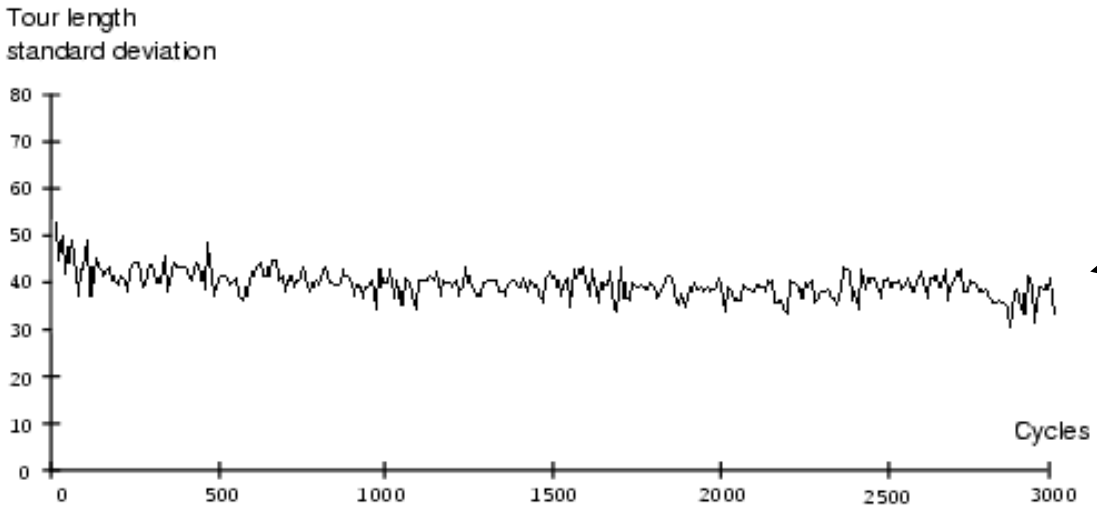


Fig. 4. Evolution of the standard deviation of the population's tour lengths (Oliver30). Typical run.

O algoritmo continua pesquisando por soluções diferentes.

Resultado

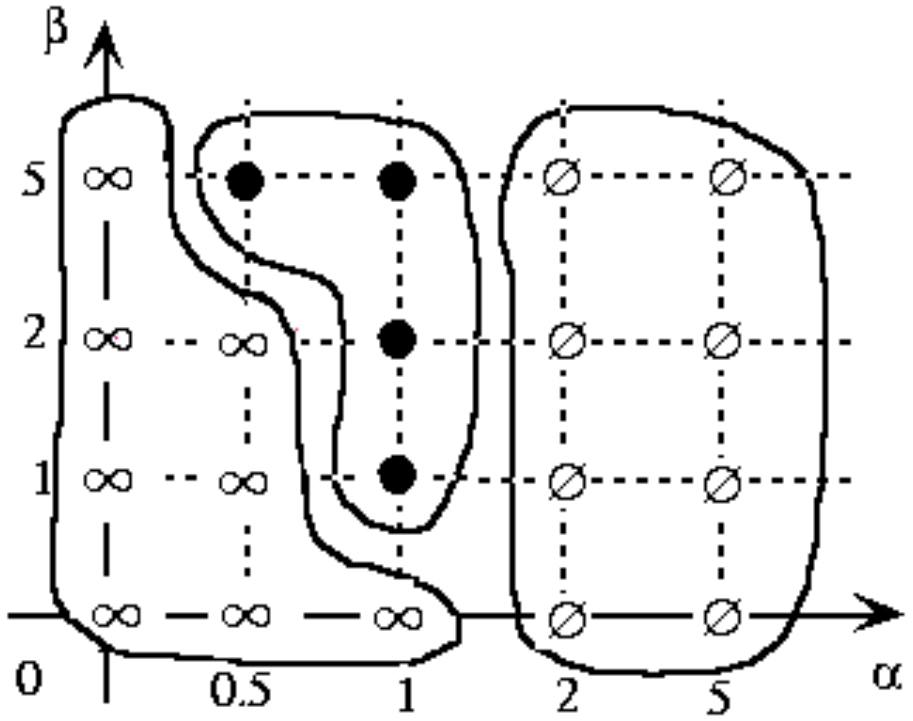


Fig. 8. *Ant-cycle* behavior for different combinations of α-β parameters.

- ● - The algorithm finds the best known solution without entering the stagnation behavior.
- ∞ - The algorithm doesn't find good solutions without entering the stagnation behavior.
- ∅ - The algorithm doesn't find good solutions and enters the stagnation behavior.

Exercício

- Responda as questões abaixo com base no artigo do Dorigo *et all* (1996), indicando a posição (página, parágrafo, ...) no texto onde está a resposta.

1. Descubra no artigo (Dorigo, 1996) o número de formigas utilizadas para resolver o problema.
2. Onde (qual cidade) cada formiga inicia seu *tour*?
3. Encontre uma justificativa para que o modelo "*ant-cycle*" tenha encontrado a melhor solução.
4. Explique por que ρ=0,5 foi o melhor valor para o algoritmo.
5. Qual foi o melhor valor de Q?
6. O que significa estagnação da solução (*stagnation behavior*)?
7. O que significa a estratégia elitista?

Histórico

Table 1. List of applications of ACO algorithms to static combinatorial optimization problems. Classification by application and chronologically ordered.

Problem name	Authors	Year	Main references	Algorithm name
Traveling salesman	Dorigo, Maniezzo & Colorni	1991	[33, 40, 41]	AS
	Gambardella & Dorigo	1995	[49]	Ant-Q
	Dorigo & Gambardella	1996	[37, 38, 50]	ACS & ACS-3-opt
	Stützle & Hoos	1997	[97, 98]	$\mathcal{MM}$AS
	Bullnheimer, Hartl & Strauss	1997	[12]	AS_{rank}
Quadratic assignment	Maniezzo, Colorni & Dorigo	1994	[77]	AS-QAP
	Gambardella, Taillard & Dorigo	1997	[53, 54]	HAS-QAP[a]
	Stützle & Hoos	1998	[99]	$\mathcal{MM}$AS-QAP
	Maniezzo & Colorni	1998	[76]	AS-QAP[b]
	Maniezzo	1998	[75]	ANTS-QAP
Job-shop scheduling	Colorni, Dorigo & Maniezzo	1994	[20]	AS-JSP
Vehicle routing	Bullnheimer, Hartl & Strauss	1996	[15, 11, 13]	AS-VRP
	Gambardella, Taillard & Agazzi	1999	[52]	HAS-VRP
Sequential ordering	Gambardella & Dorigo	1997	[51]	HAS-SOP
Graph coloring	Costa & Hertz	1997	[22]	ANTCOL
Shortest common supersequence	Michel & Middendorf	1998	[78, 79]	AS-SCS

[a] HAS-QAP is an ant algorithm which does not follow all the aspects of the ACO meta-heuristic.
[b] This is a variant of the original AS-QAP.

Bibliografia

- Página do Dorigo:
  * Dorigo, M. Maniezzo, V. Colorni, A. (1996).**The Ant System: Optimization by a colony of cooperating agents** (1996). *IEEE Transactions on Systems, Man, and Cybernetics Part B*. V. 6 (1), pp. 1-13.

  * http://iridia.ulb.ac.be/~mdorigo/ACO/ACO.html


Dúvidas??