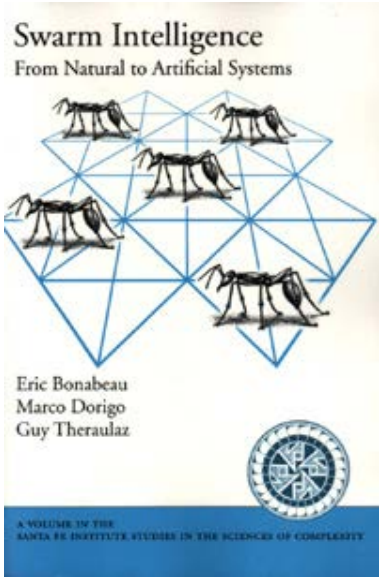
Verfahren der Schwarm Intelligenz

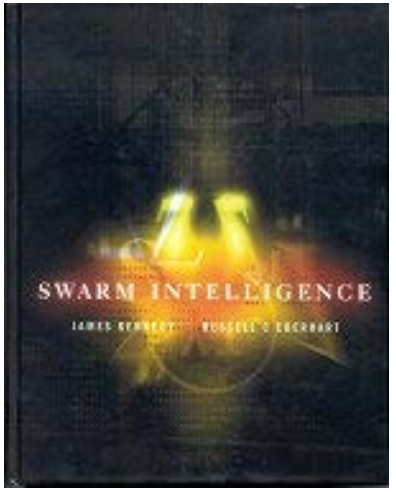
Martin Middendorf
Schwarmintelligenz und Komplexe Systeme
middendorf@informatik.uni-leipzig.de

Literatur

1. **Eric Bonabeau, Marco Dorigo, Guy Theraulaz: Swarm Intelligence,**
   Oxford University Press, 1999.



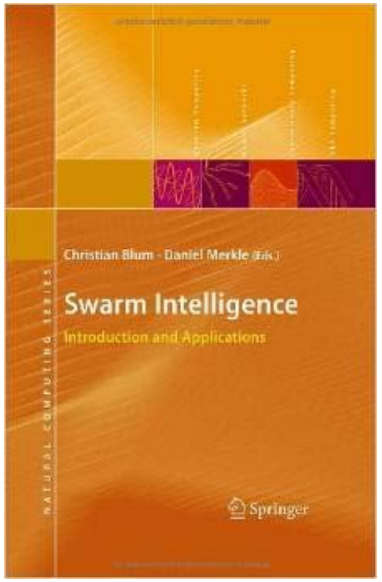
2. **James Kennedy, Russell C. Eberhart, Shi Yuhui: Swarm Intelligence,**
   Morgan Kaufmann, 2001.



Literatur

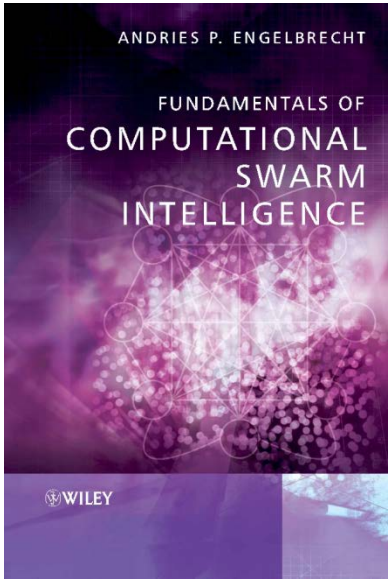
3. **Christian Blum, Daniel Merkle: Swarm Intelligence - Introduction and Applications,**

   Springer, 2008.

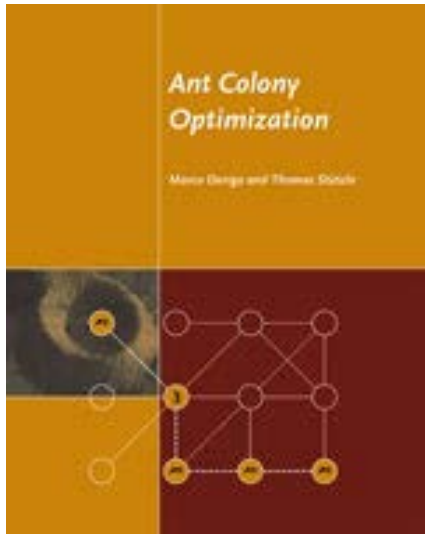


Literatur

4. **Andries P. Engelbrecht: Fundamentals of Computational Swarm Intelligence**
   Wiley, 2006

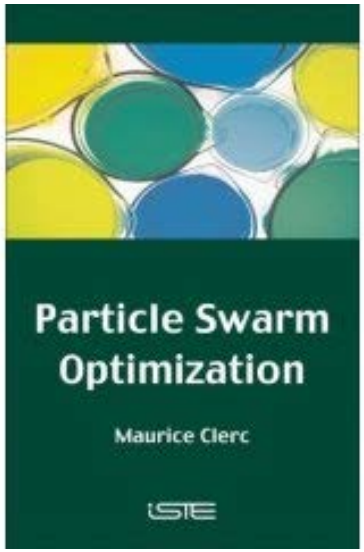


5. **Marco Dorigo, Thomas Stützle: Ant Colony Optimization**
   MIT Press, 2004

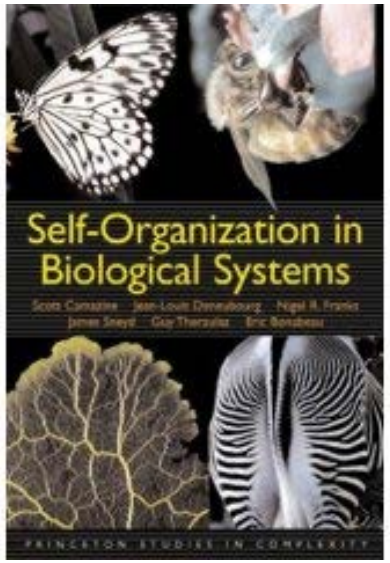


Literatur

6. **Maurice Clerc: Particle Swarm Optimization, ISTE, 2006.**

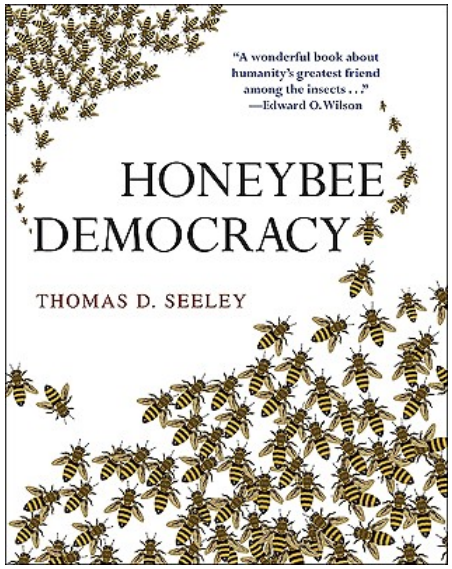


7. **Scott Camazine, Jean-Louis Deneubourg, Nigel R. Franks James Sneyd, Guy Theraulaz, Eric Bonabeau: Self-Organization in Biological Systems, Princeton University Press, 2003.**

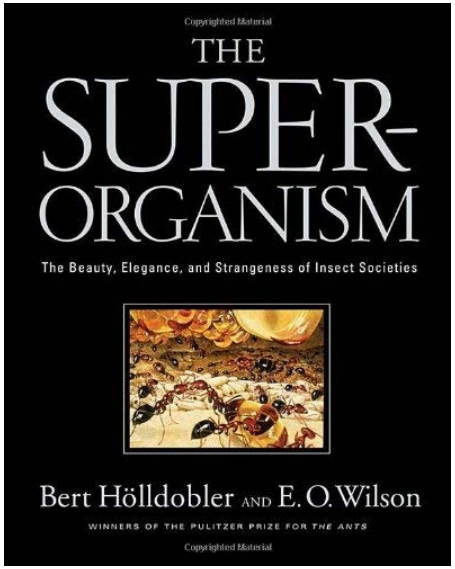


Literatur

8. **Tom Seeley: Honeybee Democracy, Princeton University Press, 2010**
   deutsch: Bienendemokratie, Fischer Verlag, 2014



9. **Bert Hölldobler, Edward O. Wilson: The Superorganism: The Beauty, Elegance, and Strangeness of Insect Societies, W. W. Norton & Company, 2008**

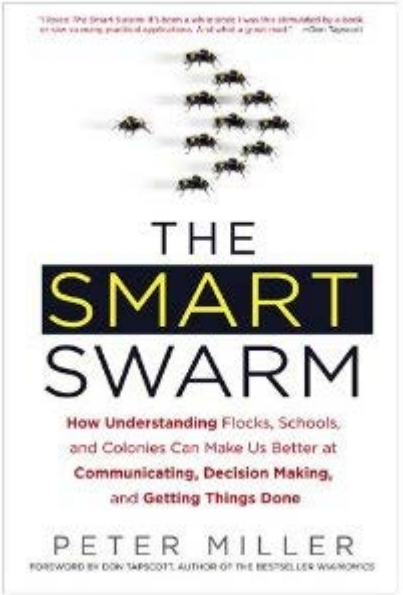


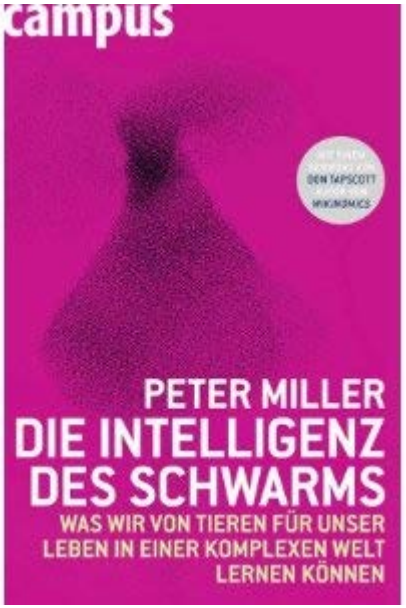
Literatur

In den letzen Jahren findet sich das Thema Schwarmintelligenz zunehmend in der populärwissenschaftlichen Literatur und sogar in Romanen

10. Peter Miller: The Smart Swarm: How Understanding Flocks, Schools, and Colonies Can Make Us Better at Communicating, Decision Making, and Getting Things Done, Avery, 2010.

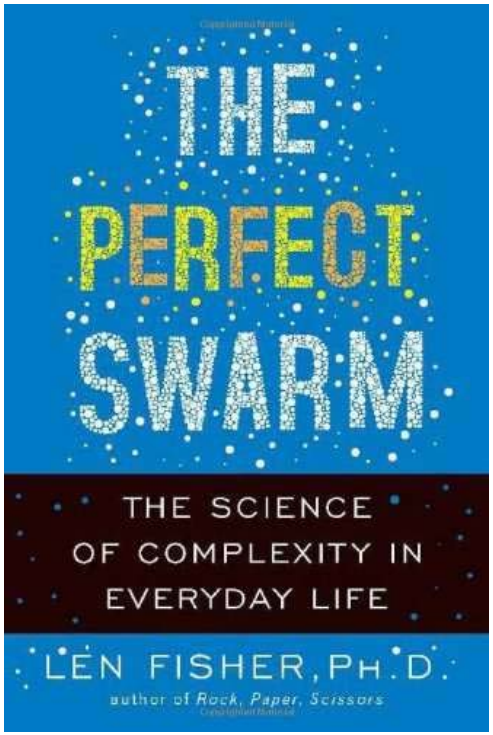
deutsche Ausgabe: Die Intelligenz des Schwarms: Was wir von Tieren für unser Leben in einer komplexen Welt lernen können, Campus, 2010.





Literatur

11. Len Fisher: The Perfect Swarm: The Science of Complexity in Everyday Life, Basic Books, 2009



Die Vorlesung beruht zu großen Teilen auf Originalliteratur.

Copyrighted Material

SCHÄTZING

SCHWARM-INTELLIGENZ.

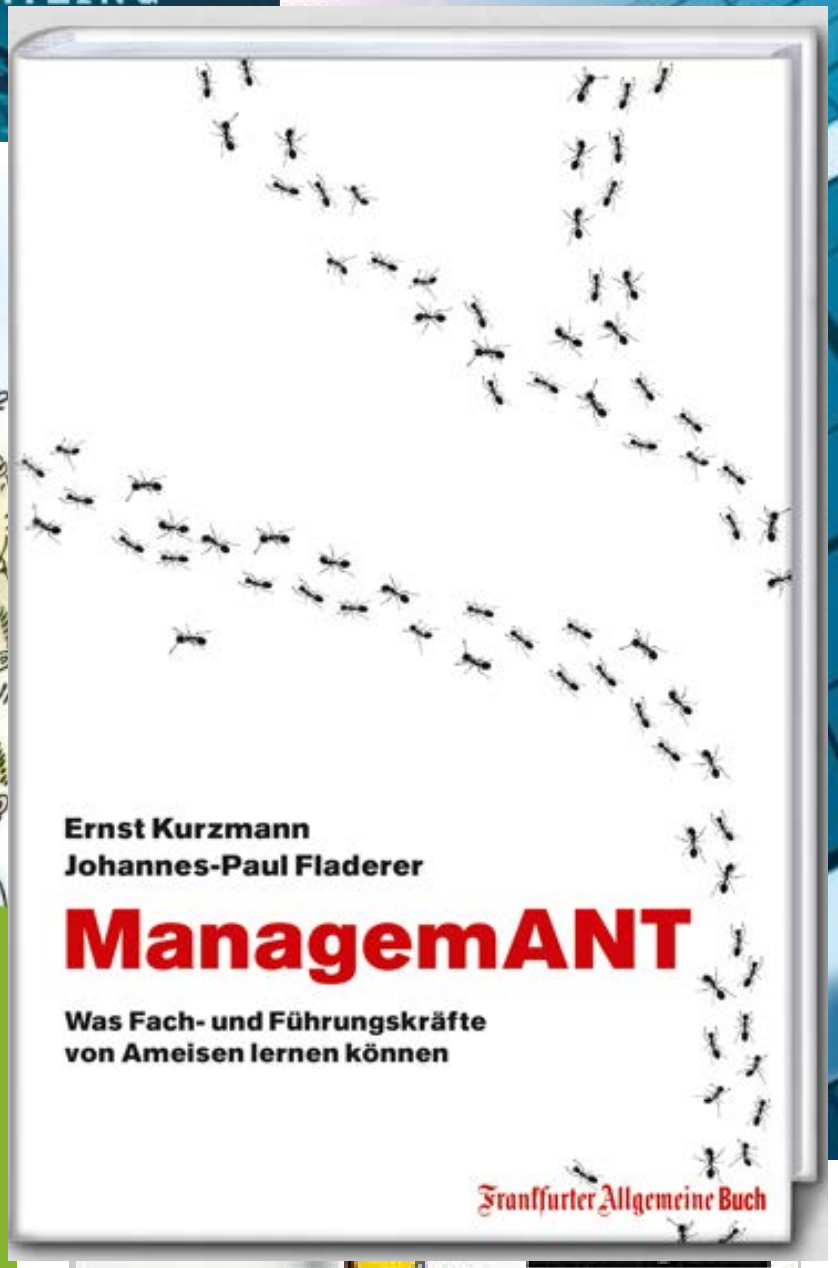
WWW.JET-TANKSTELLEN.DE

JET

SCHLAUER IST DAS.

Warum Gruppen klüger sind als Einzelne

GOLDMANN



Guttenberg im Netz der Schwarmintelligenz

Der Herr Dr. und die Schwarmintelligenz des Internets

„Schwarmintelligenz“ im Kampf gegen Plagiate

Schwarmintelligenz bringt Guttenberg in Bedrängnis

Guttenberg promoviert die Schwarmintelligenz

Der Schwarm rupft fremde Federn - GuttenPlag und VroniPlag gegen Wissenschafts-Posing

Guttenberg: die mobbende Schwarmintelligenz hat gesiegt

„GUTTENBERG IST VON EINEM SCHWARM VERSCHLUCKT WORDEN“

"Schwarmkontrolle" brachte Minister zu Fall

Piraten-Parteitag

An der Grenze der Schwarmintelligenz

Quelle: FR online

Die Schwarmintelligenz soll Entscheidungen fällen, die Funktionäre dann bloß verbreiten.

Quelle: Stern

Piraten-Konzept: "Wir wollen Schwarmintelligenz statt Köpfe"

DER PIRATENSCHWARM HAT SEINE INTELLIGENZ NOCH NICHT BEWIESEN

Viel Schwarm und wenig Intelligenz – so scheint derzeit das Motto der selbsternannten Partei der „Schwarmintelligenz“ zu lauten.

Schwarmintelligenz, der Prinz und die Erscheinung

Die Grünen bejubeln ihr Spitzenkandidaten-Duo. Sie feiern den Auftritt als Beweis für ihre Schwarmintelligenz.

Quelle: Die Welt

Urwahl der Spitzenkandidaten: Grüne Schwarmintelligenz

Die Überraschung für viele war, dass die Mitglieder tatsächlich eine Art "Schwarm-Intelligenz" bewiesen,

Schwarmintelligenz nutzen: Die nächste große Sache!

Alternativen zum Biosprit: Schwarmintelligenz an der Zapfsäule

Schwarmintelligenz der Schwarzfahrer

Die „Schwarmintelligenz“ scheitert am grauen Unternehmensalltag

Schwarmintelligenz im Supermarkt

Selbstfahrende Autos sollen Schwarmintelligenz nutzen

Schwarmintelligenz auf der Balkanroute

Schwarmintelligenz ausgehebelt: Wikipedia-Artikel für Pharmakonzern geschönt?

Hat die Schwarmintelligenz immer recht?

Die Schwarmintelligenz lebt

Schwarmintelligenz bringt Unternehmen nach vorne

„Die Schwarmintelligenz der Gesellschaft ist einem politischen Diktat überlegen“, stellte der FDP-Generalsekretär klar.

Schwarmintelligenz: Keiner weiß so viel wie alle

Schwarmintelligenz fürs Bürgerzentrum

Fußballer sind schwarmintelligent

Wir-Denken: Auf dem Weg zur Schwarm-Intelligenz

CDU setzt auf die Schwarmintelligenz

Schwarmintelligenz für die Entwicklung ländlicher Räume.

Schwarmintelligenz prognostiziert Wahlausgang

Was Industrie 4.0 mit Schwarmintelligenz zu tun hat

Wie weit reicht die Schwarmintelligenz im Banking?

Schwarm-Intelligenz und Rudel-Blödsinn

Trendforschung: Internet ist Auslöser der "Schwarm-Intelligenz"

Warum Ameisen die besseren Manager sind

Klimaschutz mit Schwarmintelligenz

Die Schwarmintelligenz sagt: "Bohr' in der Nase!"

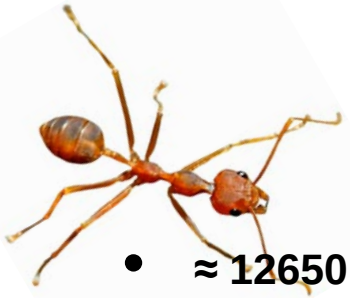
Schwarmintelligenz auch beim Menschen?

Was Manager von Bienen lernen können

Überblick

1. Einführung
2. Ant Colony Optimization
3. Particle Swarm Optimization
4. Task Selection
5. Ant Clustering

Ameisen



- ≈ 12650 Arten sind bekannt
- Koloniegrößen von ca. 50 bis mehrere Millionen Individuen (abhängig von der Art)
- Lebensdauer der Königin bis zu 30 Jahren


Ameisen sind als **eusoziale** Insekten gekennzeichnet durch

i. Zusammenarbeit bei der Brutpflege
ii. überlappende und zusammen lebende Generationen sowie
iii. Aufteilung in reproduktive (Königin, Männchen) und nicht reproduktive Individuen (Arbeiterinnen)

Arbeiterinnen sind in **Kasten** aufgeteilt




Ameisen

Aufgaben in einer Ameisenkolonie

- Reproduktion und Brutpflege

- Nestbau & -unterhalt
  - Blattnester der Weberameisen
  - Höhlen

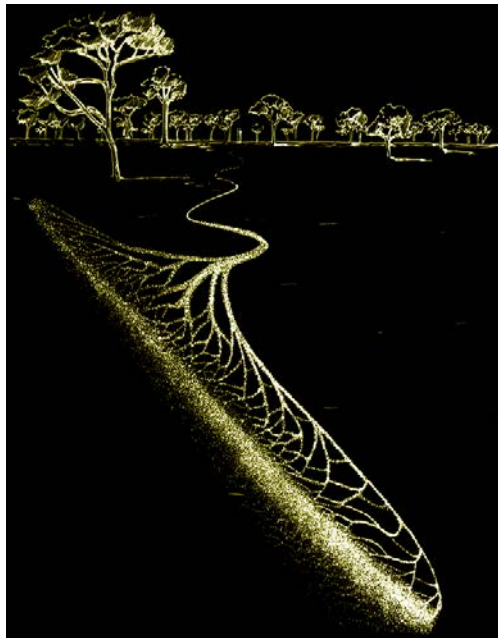
- Verteidigung (Königin, Nest, Territorium, Nahrung)
  - Türsteher
  - Steine-werfende Ameisen
- Versorgung mit Nahrung
  - Beutezüge der Treiberameisen

Ameisen

Aufgaben in einer Ameisenkolonie

Photo: A. Wild

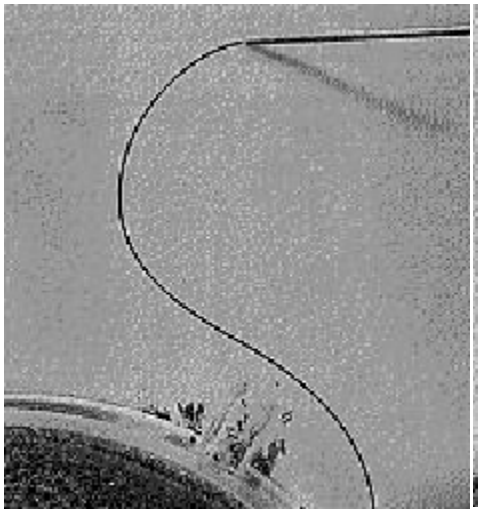
- Reproduktion und Brutpflege
- Nestbau & -unterhalt
  - Blattnester der Weberameisen
  - Höhlen
- Verteidigung (Königin, Nest, Territorium, Nahrung)
  - Türsteher
  - Steine-werfende Ameisen
- Versorgung mit Nahrung
  - Beutezüge der Treiberameisen

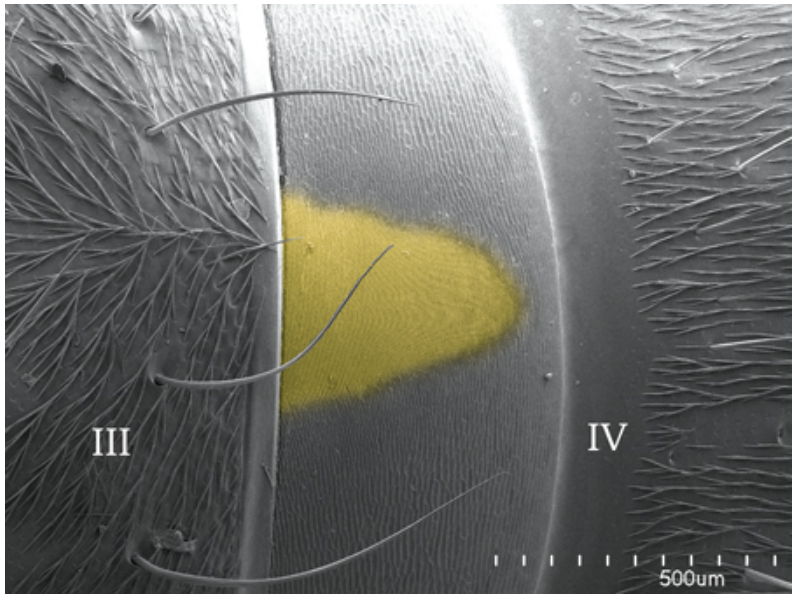


Ameisen

Kommunikation

- Different pheromones
  - Alarm pheromones
  - Trail pheromones
  - Queen and brood pheromones
- Sound
  - Drumming
  - Stridulation
- Tactile
- Visual
- Multimodal signals

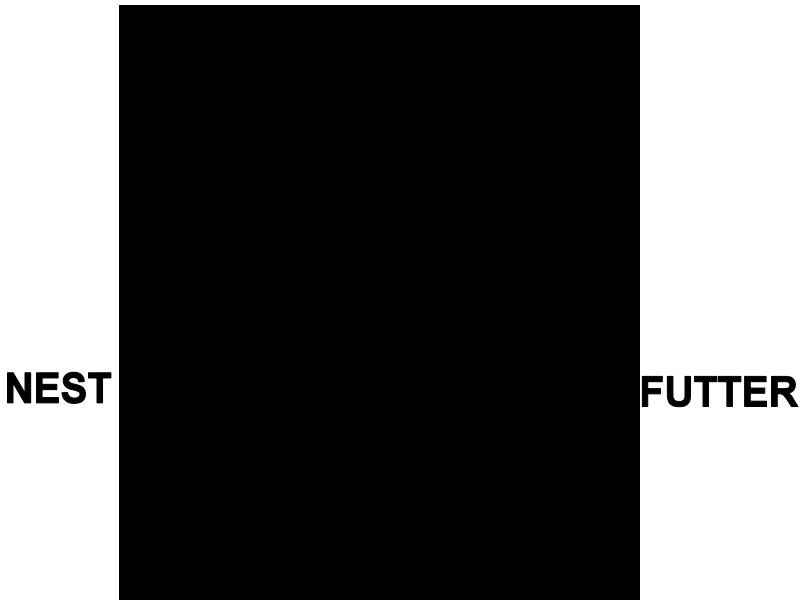




Phtoto: R. A. Keller/AMNH

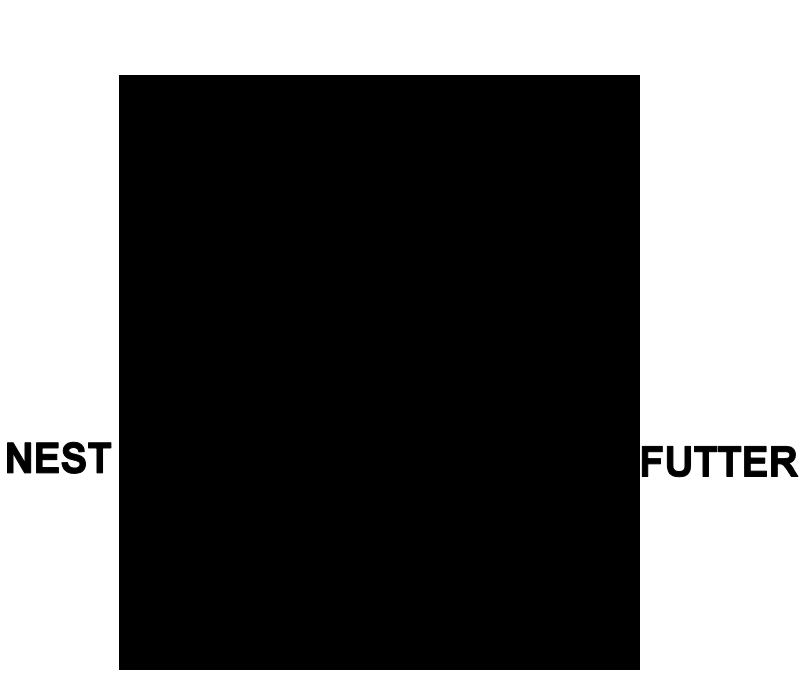
Ein Experiment zum Verhalten bei der Nahrungssuche

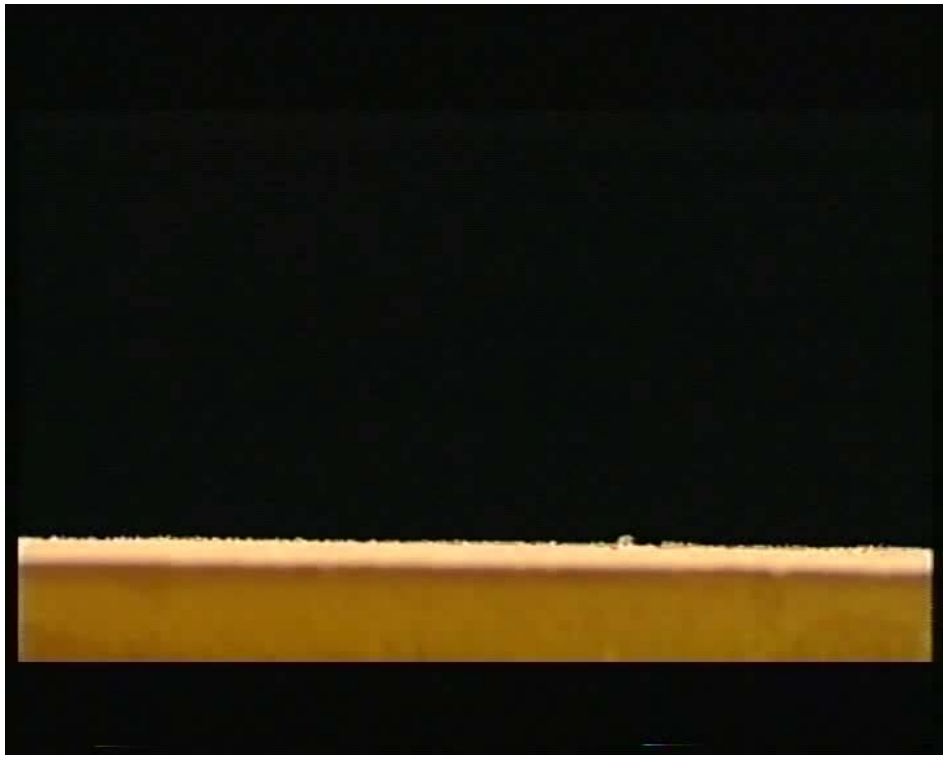
[Goss et al. 89, Deneubourg, Theraulaz et al. 90]



Ein Experiment zum Verhalten bei der Nahrungssuche

[Goss et al. 89, Deneubourg. Theraulaz et al. 90]

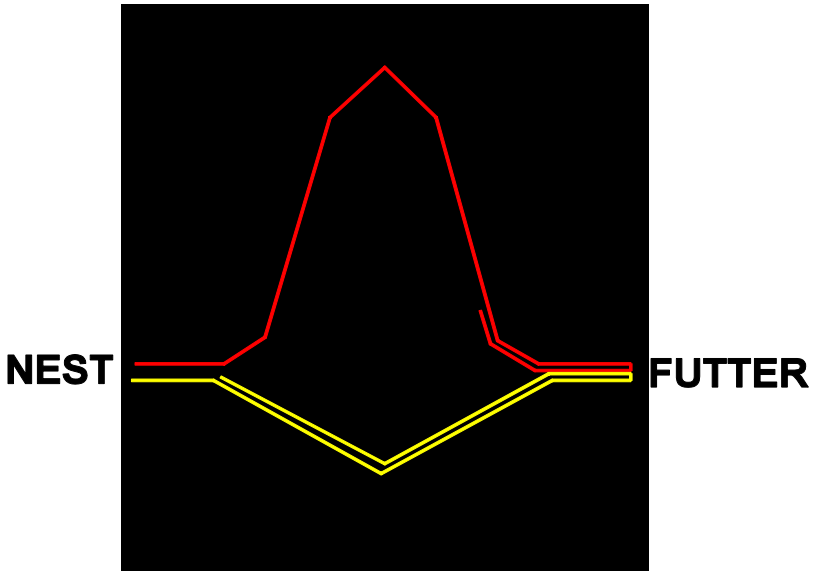




Video Theraulaz et al.

Ein Experiment zum Verhalten bei der Nahrungssuche

[Goss et al. 89, Deneubourg, Theraulaz et al. 90]



Ant Colony Optimization

Simple discrete model:

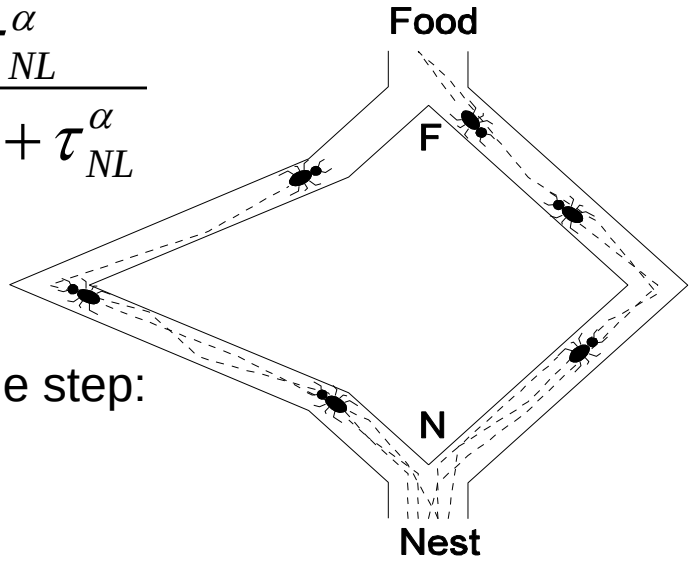
Ants walk in both directions (and place one unit of pheromone every time step):

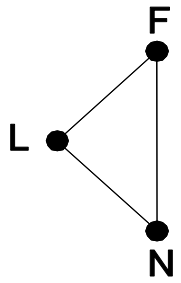
crossing from N directly to F (or vice versa) takes one time step

crossing from N via L to F (or vice versa) takes two time steps

Probability p_{NY} for ant at node N to choose the branch to Yε{N,L} is:

$$p_{NF} = \frac{\tau_{NF}^{\alpha}}{\tau_{NF}^{\alpha} + \tau_{NL}^{\alpha}} \qquad p_{NL} = \frac{\tau_{NL}^{\alpha}}{\tau_{NF}^{\alpha} + \tau_{NL}^{\alpha}}$$





Pheromone evaporation after every time step:

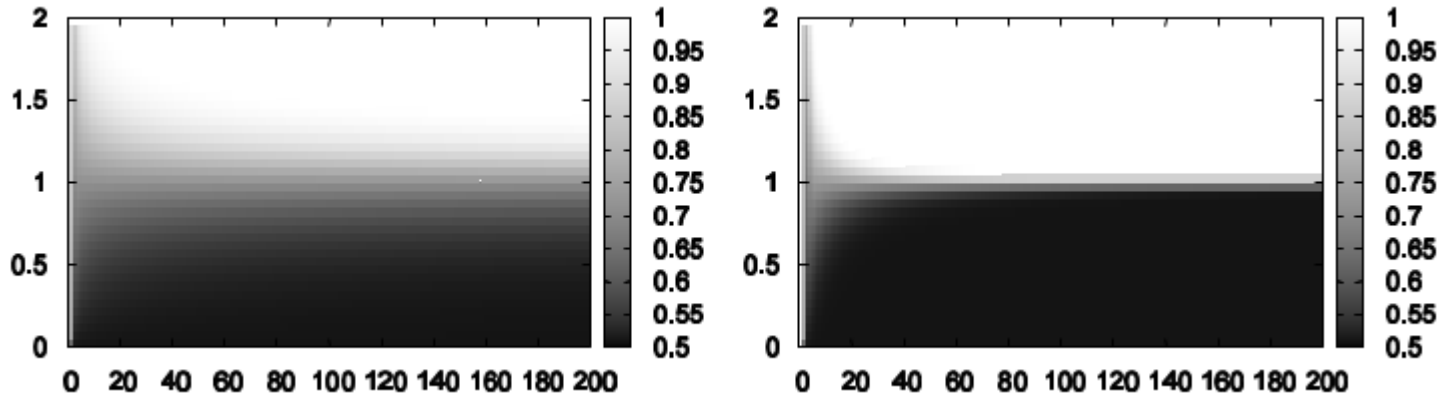
$$\tau_{XY} = (1-\rho) \cdot \tau_{XY}$$

where ρ Є[0,1) is the evaporation rate and XY ε{NF,NL,LF}

Ant Colony Optimization

Change of probability to chose NF from N over time (time steps 1 – 200)

- for different values of parameter αЄ[0,2]



ρ=0.1 ρ=0.5

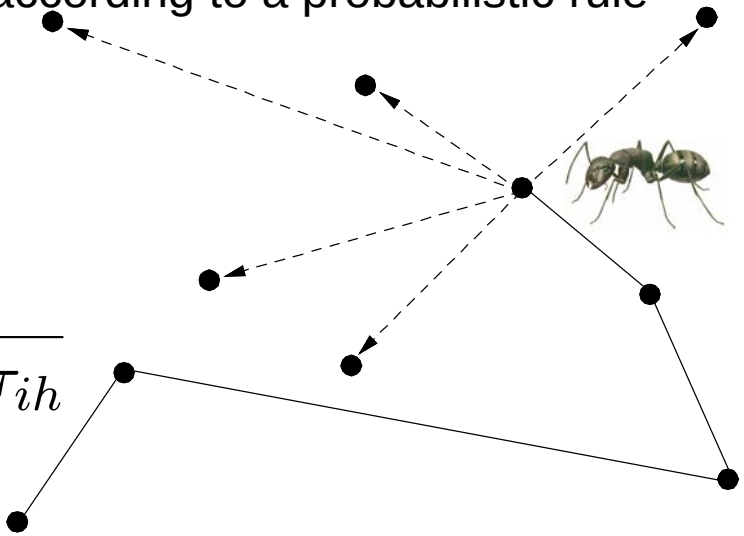
dark: low probability , bright: high probability

Ant Colony Optimization

Application to Shortest Path Problems: [Dorigo et al.,91]

Example: Traveling Salesperson Problem

Idea: An (artificial) ant chooses a tour randomly
- the ant remembers the cities it has already visited and chooses the next city according to a probabilistic rule



Probabilistic Rule:

$$p_{ij} := \frac{\tau_{ij}}{\sum_{h \in \mathcal{S}} \tau_{ih}}$$

where p_{ij} = probability that an ant on city i selects city j as next one

τ_{ij} = amount of pheromone on edge (i,j)

S = set of cities not yet visited by the ant

Ant Colony Optimization

Pheromone Update:

After all ants have constructed a solution the pheromone values are updated

i) **Evaporation**

$$\tau_{ij} = (1-\rho)\cdot\tau_{ij} \qquad 0 < \rho < 1$$

ii) **Deposit**

$$\tau_{ij} = \tau_{ij} + \Delta_{ij}$$

where the solution of ant k is S_k, the quality of S_k is $f(S_k)$ and

$$\Delta_{ij} = \sum_{\substack{k=1,\ldots,m \\ (ij)\varepsilon S_k}} f(S_k)$$

For TSP: $f(S_k)$:=1/(length of tour found by ant k)

Remark: in most ant algorithms only the ant that found the best solution is allowed to add pheromone

Ant Colony Optimization

Heuristic

Use of heuristic information can improve the choice of an ant

→ **extended probabilistic rule**: $p_{ij} := \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{h \in \mathcal{S}} \tau_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}}$

τ_{ij}= Pheromone value for edge (i,j)

η_{ij} = Heuristic value for edge (i,j)

α = Influence of phermone

β = Influence of heuristic

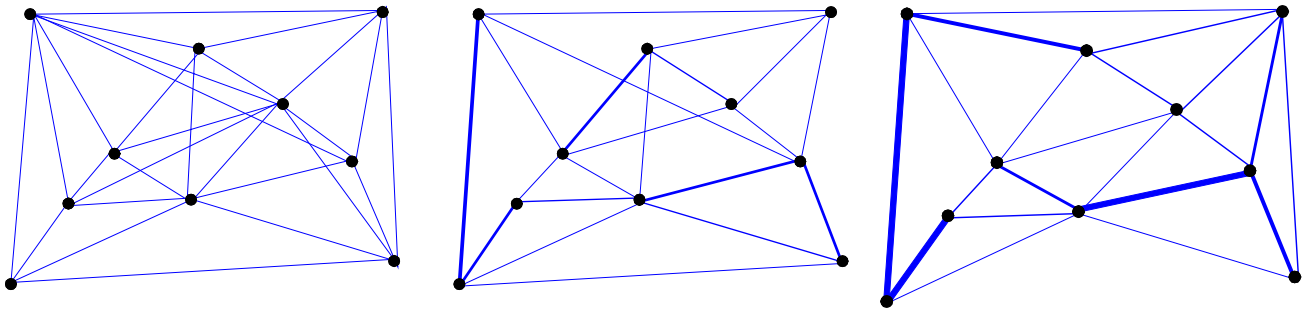
S = set of cities not visited by the ant

For TSP: η_{ij}:=1/distance(i,j)

Ant Colony Optimization

Loop
For k := 1 to *m* * each of m ants builds a tour*
    Place ant k on a randomly chosen start node
    For step := 1 to *n* * each ant adds a node to its path *
        Choose next node to move by applying the **probabilistic rule**
    End-for
End-for
Update pheromone information
Until End_condition

Example: change of pheromome values over time



Ant Colony Optimization

Results: TSP (from the „classical“ paper of Dorigo et al. 1997)

Problem name	ACS	GA	EP	SA	Optimum
Eil50	**425**	428	426	443	**425**
(50-city problem)	(427.96)	(N/A)	**(427.86)**	(N/A)	(N/A)
	[1,830]	[25,000]	[100,000]	[68,512]	
Eil75	**535**	545	542	580	**535**
(75-city problem)	**(542.37)**	(N/A)	(549.18)	(N/A)	(N/A)
	[3,480]	[80,000]	[325,000]	[173,250]	
KroA100	**21,282**	21,761	N/A	N/A	**21,282**
(100-city problem)	**(21,285.44)**	(N/A)	(N/A)	(N/A)	(N/A)
	[4,820]	[103,000]	[N/A]	[N/A]	

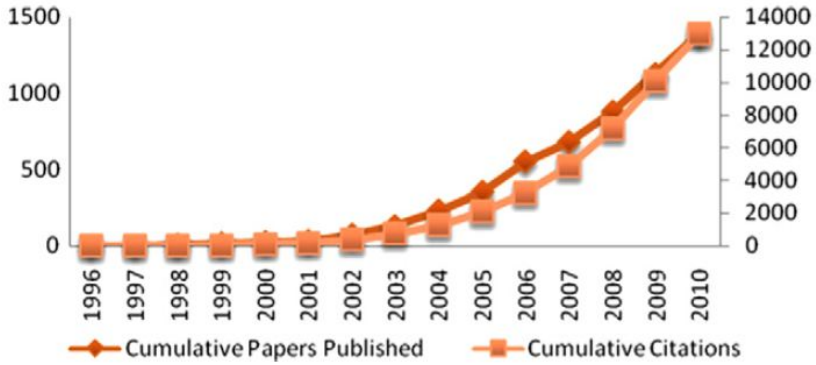
Comparison of ant algorithm (ant colony system, ACS) with other metaheuristics:

- genetic algorithm (GA)
- evolutionary programming (EP)
- simulated annealing (SA)

Ant Colony Optimization

Citations of the literature on ACO in different subject areas during 1996–2010.

Subject	Citation
Computer Science, Artificial Intelligence (266)	4637
Computer Science, Theory & Methods (269)	3965
Operations Research & Management Science (271)	2623
Automation & Control Systems (120)	2398
Computer Science, Cybernetics (27)	2183
Computer Science, Interdisciplinary Applications (185)	1230
Management (59)	1208
Engineering, Electrical & Electronic (226)	1126
Engineering, Industrial (113)	831
Computer Science, Information Systems (75)	531
Engineering, Manufacturing (111)	483
Mathematics, Applied (67)	332
Engineering, Multidisciplinary (79)	316
Engineering, Civil (66)	280
Water Resources (21)	185
Computer Science, Software Engineering (43)	184
Engineering, Chemical (17)	153
Telecommunications (55)	145
Computer Science, Hardware & Architecture (33)	129
Statistics & Probability (13)	127
Construction & Building Technology (19)	96
Chemistry, Analytical (18)	94
Energy & Fuels (14)	87
Instruments & Instrumentation (19)	78
Robotics (14)	75
Engineering, Mechanical (17)	69
Mathematics, Interdisciplinary Applications (26)	66
Chemistry, Multidisciplinary (19)	61
Environmental Sciences (13)	46
Transportation Science & Technology (16)	45
Physics, Applied (13)	42
Mechanics (26)	41
Optics (12)	26
Materials Science, Multidisciplinary (11)	14



[Deng,Lin,2012]

Ant Colony Optimization

1	Bauer et al. (1999)	$1\|\sum T_i$ e $1\|\sum w_i \cdot T_i$
2	den Besten et al. (2000)	$1\|\sum w_i \cdot T_i$
3	Merkle and Middendorf (2000)	$1\|\sum w_i \cdot T_i$
4	Merkle and Middendorf (2003)	$1\|\sum w_i \cdot T_i$
5	Holthaus and Rajendran (2005)	$1\|\sum w_i \cdot T_i$
6	Cheng et al. (2009)	$1\|\sum T_i$
7	Gagn et al. (2002)	$1/s_{ij}/\sum T_i$
8	Liao and Juan (2007)	$1/s_{ij}/\sum w_i \cdot T_i$
9	Anghinolfi et al. (2008)	$1/s_{ij}/\sum w_i \cdot T_i$
10	Kashan and Karimi (2008)	1/batch, incompatible, $s_{ij}/\sum w_i \cdot T_i$
11	Thiruvady et al. (2009)	$1/s_{ij}/M$
11	Xu et al. (2012)	$1/r_j,s_j$, batch/C_{max}
12	Cheng et al. (2008)	1/batch/C_{max}
13	Sankar et al. (2005)	$P_m/s_{ij}/C_{max}$
14	Behnamian et al. (2009)	$P_m/s_{ij}/C_{max}$
15	Gatica et al. (2010)	$P_m\|T_{max}$
16	Raghavan and Venkataramana (2006)	P_m/batch, incompatible/$\sum w_i \cdot T_i$
17	Li et al. (2008)	P_m/A_{ij},batch, incompatible/$\sum w_i \cdot T_i$
18	Li et al. (2009)	$P_m/A_{ij},Q_i$,batch, incompatible/$\sum w_i \cdot T_i$
19	Zhou et al. (2007)	$R_m\|\sum w_i \cdot T_i$
20	Monch (2008)	$R_m\|\sum w_i \cdot T_i$
21	Arnaout et al. (2008)	$R_m/s_{ijk}/C_{max}$
22	Arnaout et al. (2009)	$R_m/s_{ijk}/C_{max}$
23	Ying and Liao (2003)	F_m/prmu/M
24	Rajendran and Ziegler (2004)	F_m/prmu/M
25	Ahmadizar et al. (2007)	F_m/prmu/M
26	Chen et al. (2008)	F_m/prmu/M
27	Zhou and Qingshan (2009)	F_m/prmu/M
28	Ying and Lin (2007)	$F_m\|M$
29	Shyu et al. (2004)	F_2/nwt,setup/M
30	Yan-hai et al. (2005)	$F_m\|$prmu,rush,M
31	T'kindt et al. (2002)	$F_2\|\sum C_i$ e $F_2\|$Lex(C_{max}, $\sum Ci$)
32	Li et al. (2011)	$F_m/s_{ij}/\sum C_i$
33	Tavares Neto and Godinho Filho (2011)	F_m/prmu, budget/$(1-\delta)\cdot M+\delta \cdot OC$
34	Rajendran and Ziegler (2004)	F_m/prmu/M e F_m/prmu/$\sum F_i$
35	Rajendran and Ziegler (2005)	F_m/prmu/$\sum F_i$
36	Gajpal and Rajendran (2006)	F_m/prmu/CTV
37	Li and Zhang (2006)	$F_2\|\alpha \cdot \sum C_i+(1-\alpha)\cdot M$
38	Pasia et al. (2006)	$F_m\|\sum T_i,M$
39	Yagmahan and Yenisey (2008)	F_m/prmu/$\sum C_i,M,\sum F_i$,idle
40	Al-Anzi and Allahverdi (2009)	F_2/prmu/$u\cdot M+v\cdot\sum Ci$
41	Lin et al. (2008)	F_m/prmu/$u\cdot M+v\cdot\sum C_i$
42	Marimuthu et al. (2009)	F_m/batch/M e F_m/batch/$\sum F$
43	Huang and Yang (2009)	$F_m/r_j,S_{ij}/\alpha\sum_{i=1}^{m}$ MП$_i+\beta\sum_{i=1}^{m}\sum_{j=1}^{B_j}$
44	Colorni et al. (1994)	$J_{n,m}\|M$
45	Zhang et al. (2010)	$J_{n,m}\|M$
46	Shih-Pang et al. (2011)	$J_{n,m}\|M$
47	Yoshikawa and Terai (2006)	$J_{n,m}\|M$
48	Udomsakdigool and Kachitvichyanukul (2008)	$J_{n,m}\|M$
49	Heinonen and Pettersson (2007)	$J_{10,10}\|M$
50	Zhuo et al. (2007)	$J_{10,10}\|M$
51	Huang and Liao (2008)	$J_{10,10}\|M$
52	Eswaramurthy and Tamilarasi (2009)	$J_{n,m}\|M$
53	Mirabi (2011)	$J_{n,m}/s_{ij}/M$
54	Huang and Yang (2008)	$J_{n,m}$/window/$\sum(u\cdot E_l+v\cdot T_i)$

Ant Colony Optimization

Population based ACO (P-ACO): [Guntsch,Middendorf,2002]

Observation: Standard ACO algorithm transfers pheromone information from one iteration to the next

Idea: - transfer a set of solutions (=population) from one iteration to the next

- in each iteration use the population of solutions to build up the pheromone information that is then used by the ants

Which solutions should be included into the population (2 methods)?

1. enter the best solution from the last iteration, and remove the oldest solution in the population
   - population can then be stored in a first-in-first-out queue
2. enter the best solution from the last iteration, and remove the worst solution in the population

For the first k iterations no solution is removed from the population

Ant Colony Optimization

Assumption: pheromone information is stored in a pheromone matrix

Building the Pheromone Matrix:

- Let $\tau_0 > 0, \Delta > 0$ be two parameters
- Let $\tau_{ij} = \tau_0$ for i, j$\in$[1,n]
- For each solution in population: if edge (i,j) is in the solution do

$$\tau_{ij} := \tau_{ij} + \Delta$$

Observations:

1. For each pheromone value: $\tau_0 \leq \tau_{ij} \leq \tau_0 + k \cdot \Delta$
2. There exist exactly k+1 different possible pheromone values
3. In every generation (except from the first k generations): one solution enters the population and one solution leaves the population

   → the difference between the pheromone matrices of two consecutive iterations is: pherome for one solution is removed and pheromone for solution is added

Ant Colony Optimization

Some differences between P-ACO and standard ACO:

- **Speed**:
  - pheromone update takes $O(n)$ time for P-ACO (if pheromone matrix is transferred from one iteration to the next) – compared to $O(n^2)$ for standard ACO
  - no multiplication operations are necessary for pheromone update in P-ACO

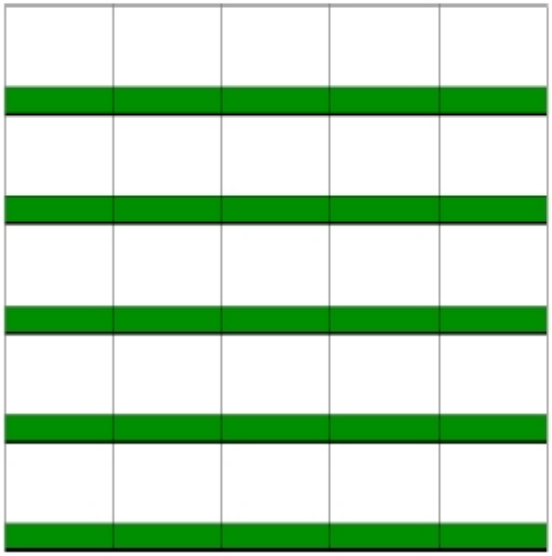
- **Agility**: old and perhaps obsolete information is completely removed in P-ACO after k iterations from the pheromone matrix

- **Analysis**: understanding of optimization behaviour of P-ACO might be easier due to the finite number of different possible pheromone matrices

  → this number is independent from the total number of iterations

Population Based ACO: Update

Example ($k = 3$): initial matrix, empty queue

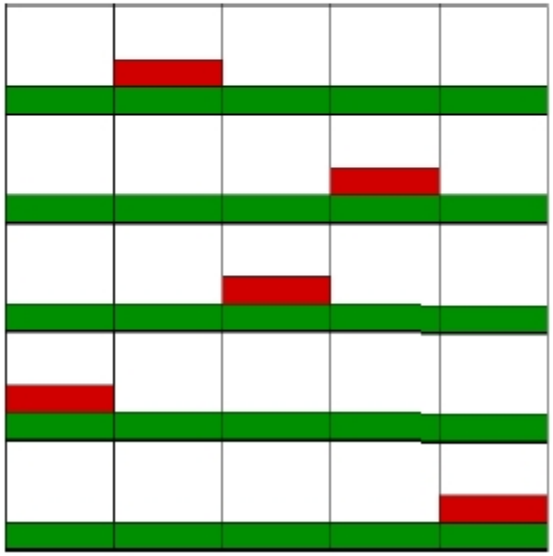


FIFO–Queue

(empty)
(empty)
(empty)

Population Based ACO: Update

Example ($k = 3$): positive update with (2,4,3,1,5)

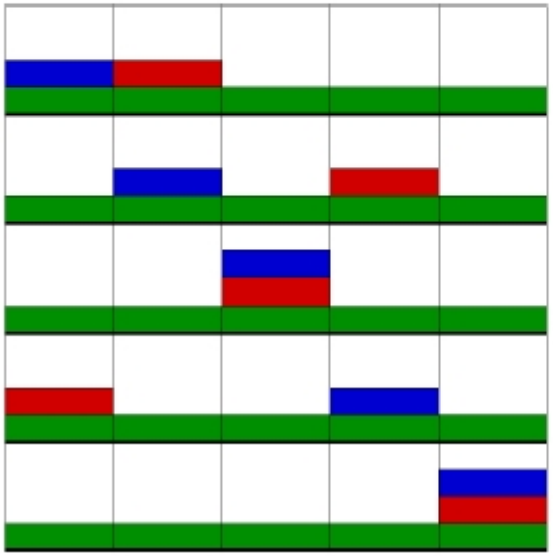


FIFO–Queue

(2,4,3,1,5)
(empty)
(empty)

Population Based ACO: Update

Example $(k = 3)$: positive update with (1,2,3,4,5)

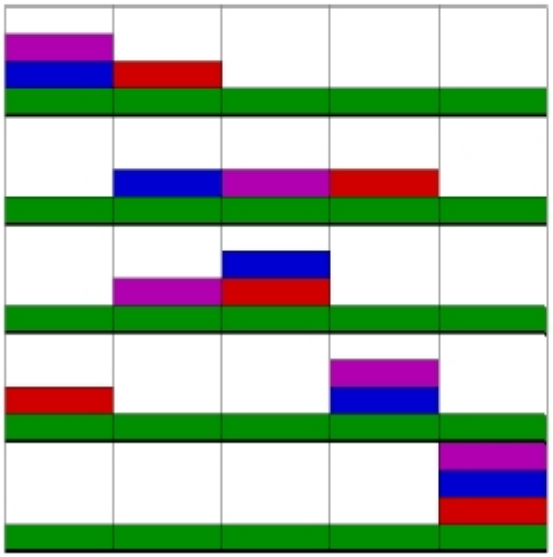


FIFO–Queue

(1,2,3,4,5)
(2,4,3,1,5)
(empty)

Population Based ACO: Update

Example $(k = 3)$: positive update with (1,3,2,4,5)

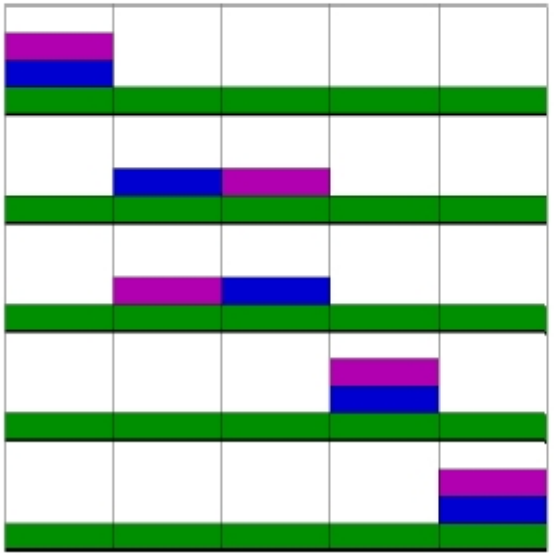


FIFO–Queue

(1,3,2,4,5)
(1,2,3,4,5)
(2,4,3,1,5)

Population Based ACO: Update

Example ($k = 3$): negative update with (2,4,3,1,5)

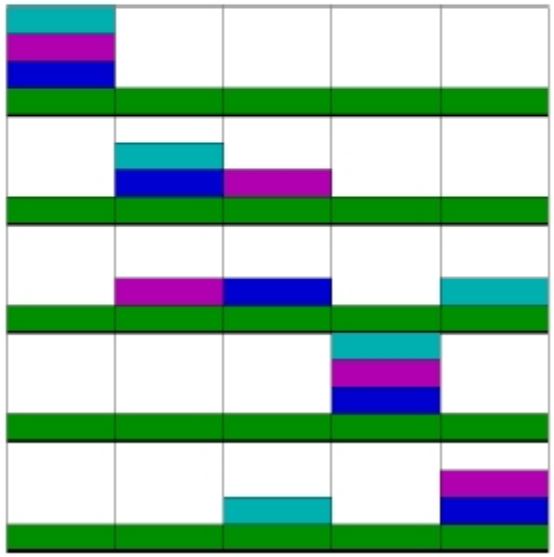


FIFO–Queue

(empty)
(1,3,2,4,5)
(1,2,3,4,5)

Population Based ACO: Update

Example ($k = 3$): positive update with (1,2,5,4,3)



FIFO–Queue

(1,2,5,4,3)
(1,3,2,4,5)
(1,2,3,4,5)

Population based ACO

Advantages of Population based ACO:

- **good performance: compared to standard ACO on TSP and QAP**

- **fast pheromone update**
  - O(n) for PACO (if population has constant size)
    O(n^2) for standard ACO

[Oliveira,Hussin,Stützle,Roli,Dorigo,2011]:

- Population based ACO with local search is a **state of the art** ACO
- Comparison with a state of the art standard ACO: MMAS

TSP:

	P-ACO		MMAS	
Instance	%cst	%ph	%cst	%ph
d198	95.64	**4.09**	43.32	**55.81**
eil101	93.63	**5.87**	47.14	**52.56**
kroA200	94.77	**4.86**	42.70	**56.98**
rd400	94.06	**5.06**	28.71	**70.03**
pcb1173	94.53	**5.02**	11.11	**88.21**
u2319	92.45	**4.96**	9.04	**88.47**

QAP:

	P-ACO		MMAS	
Instance	%cst	%ph	%cst	%ph
kra30a	99.70	**0.30**	90.46	**9.09**
wil50	99.61	**0.38**	90.08	**9.92**
tai80b	99.74	**0.26**	90.51	**9.49**
tai100b	99.76	**0.24**	90.52	**9.46**
es300	99.69	**0.31**	88.92	**11.06**

%cst = percentage of time used for solution construction

%ph = percentage of time used for pheromone update

Population based ACO

TSP suite: methods and algorithms for evaluating and comparing TSP solvers [Weise et al.,2014]

Conclusion of the authors “Prior to these experiments using the TSP suite, some of us would expect an MA (Memetic Algorithm) to be the best hybrid EC (Evolutionary Computation) method for the TSP. The results, however, tell us that P-ACO is the method of choice”.

ACO and Simple Problems

Idee: Untersuche Ameisenalgorithmen auf einfachen Problemen um ein besseres Verständnis der Wirkungsweise zu bekommen

Einige Fragen:

Wie verändert sich die Pheromonmatrix im Laufe eines Algorithmus?

Wie können die Ameisen die Pheromoninformation am besten nutzen?

Welche Entscheidungen treffen die Ameisen tatsächlich?

Spielt die Entscheidungsreihenfolge der Ameisen eine Rolle?

Im Folgenden:

Betrachte einfache Permutationsprobleme und Ameisen, die keine Heuristik verwenden

Annahme: Die Summe der Pheromonwerte in jeder Zeile und in jeder Spalte ist jeweils 1

ACO and Simple Problems

Problem: Minimum Cost Permutation

Given: Cost matrix $C=(c_{ij})$ with $i,j \in \{1,\ldots,n\}$

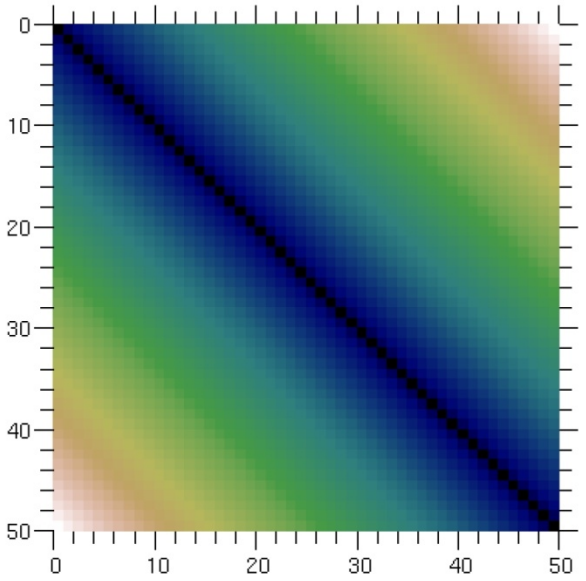
Find: Permutation π of $\{1,\ldots,n\}$ such that the following sum is minimum

$$\sum_{i=1}^{n} c_{i,\pi(j)}$$

Here: Consider the following cost matrix $C_1=(c_{ij})$:

- $c_{ij} = |i - j|+1$
- optimal solution is easy to see:

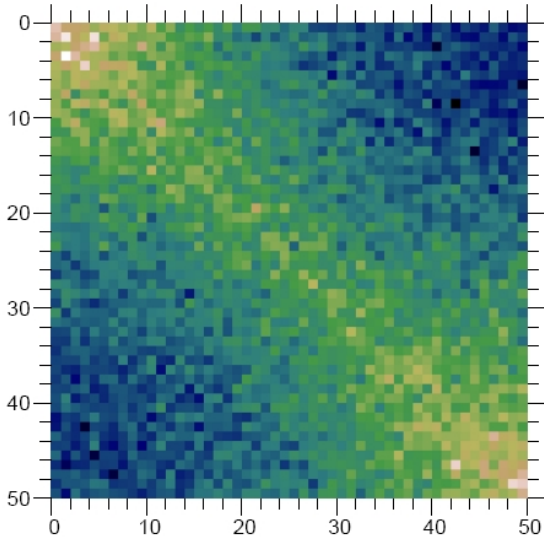
  place item i on place i, cost=50



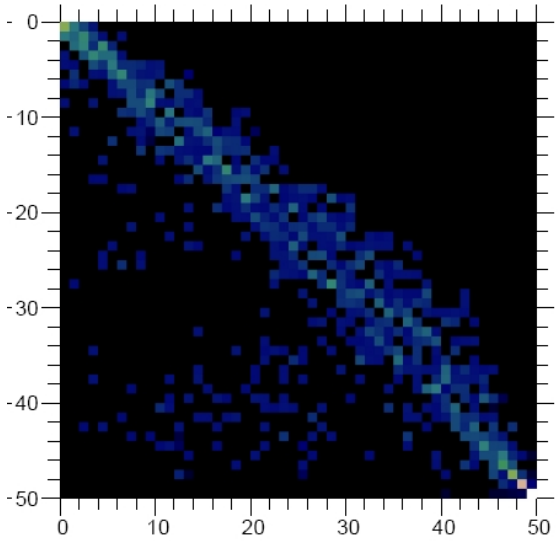
- light color: high cost
- dark color: low cost

ACO and Simple Problems

Changes of pheromone values during a run of the ACO algorithm for standard (local) pheromone evaluation:



after 500 iterations



after 4000 iterations

Observation: at the end of the run (4000 iterations) some higher pheromone values occur in the bottom left corner of the matrix

→ this indicates a fundamental problem of the standard (local) pheromone evaluation method

ACO and Simple Problems

Let σ_{ij} be the probability that an ant decides to put item j on place i

Ideally ants should make a decision with the same probability as the corresponding phermone value suggests, i.e. ideally it holds that

$$\sigma_{ij} = \frac{\tau_{ij}}{\sum_{j=1}^{n} \tau_{ij}}$$

Definition: $\sigma_{ij} - \tau_{ij}$ is called **selection bias**

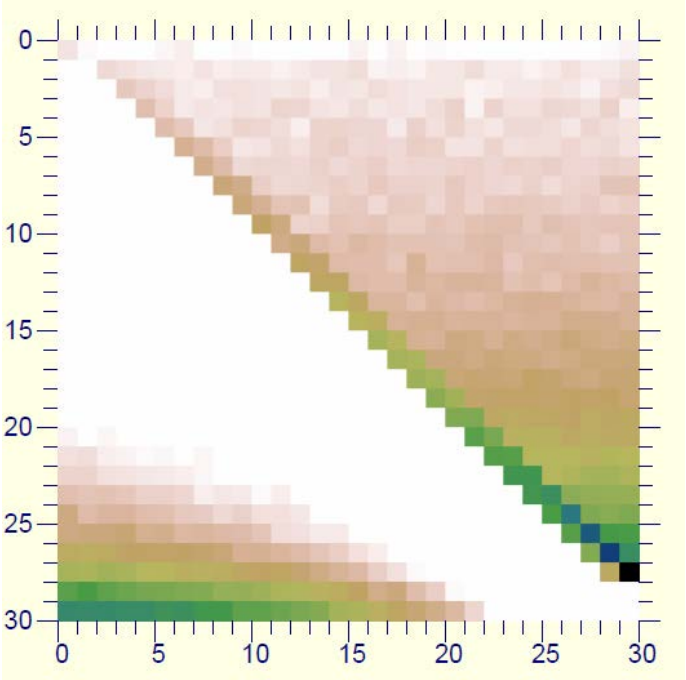
(positive selection bias if >0, otherwise negative selection bias)

Experiment: compare the frequencies of decisions of the ants with the expected values that would result „ideally“ from the pheromone values

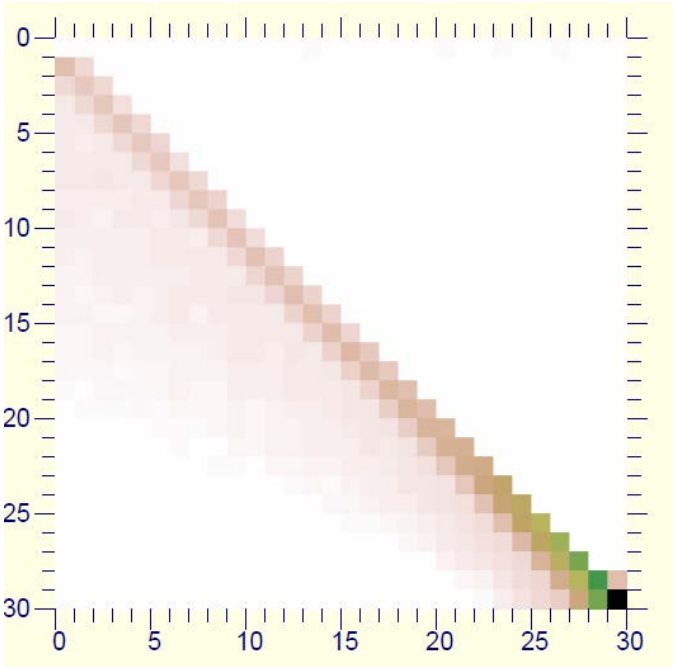
- consider the following fixed pheromone matrix: all pheromone values along the diagonal are equal and higher than all the other pheromone values (which are also equal)
- use100,000 ants for only one iteration (i.e., no pheromone evaporation, no pheromone update), no heuristic
- local (standard) pheromone evaluation

ACO and Simple Problems

Result: Selection frequencies do not correspond to pheromone values



positive selection bias



negative selection bias

dark colours = high values

Observation: no selection bias occurs in the first row

Reason: every item is selectable in the first row $\Rightarrow$

$$\sigma_{1j} - \tau_{1j} = 0 \qquad \text{for } j=1,\ldots,n$$

ACO and Simple Problems

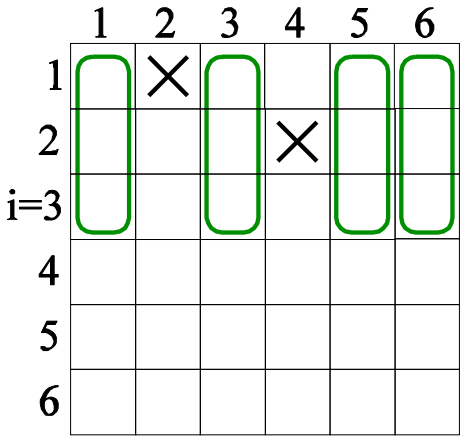
For this test problem (with cost matrix C_1):

an ant should not „forget“ an item with a high pheromone value if the ant has not chosen the item

Idea: for each ant decision use the sum of all pheromone values from the top row to the current row of the ant (instead of using only the local pheromone values in the current row)

Summation evaluation: [Merkle,Middendorf,2000]

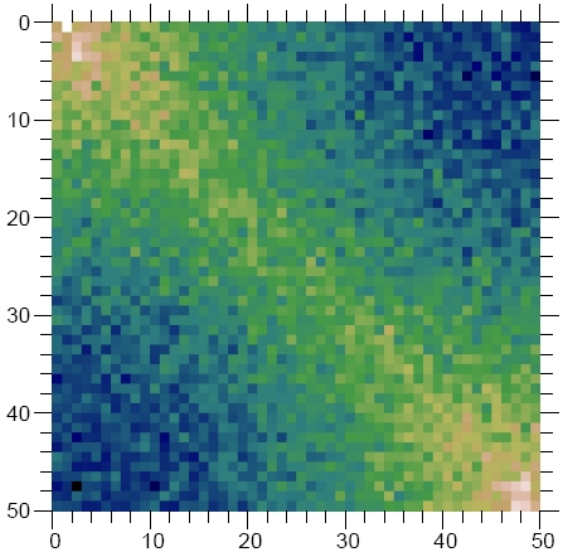
$$\tau^{\star}_{ij} := \sum_{h=1} \tau_{hj}$$



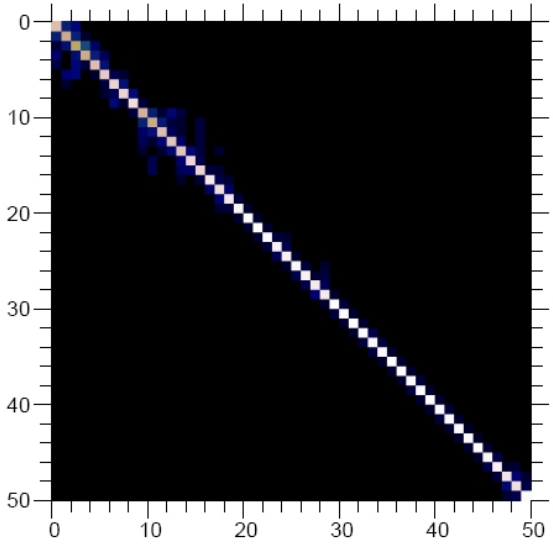
$$p_{ij} = \frac{\tau^{\star}_{ij}\eta_{ij}}{\sum_{k\in\mathcal{S}} \tau^{\star}_{ik}\eta_{ik}}$$

ACO and Simple Problems

Changes of pheromone values during a run of the ACO algorithm for (global) summation pheromone evaluation:



after 500 iterations



after 4000 iterations

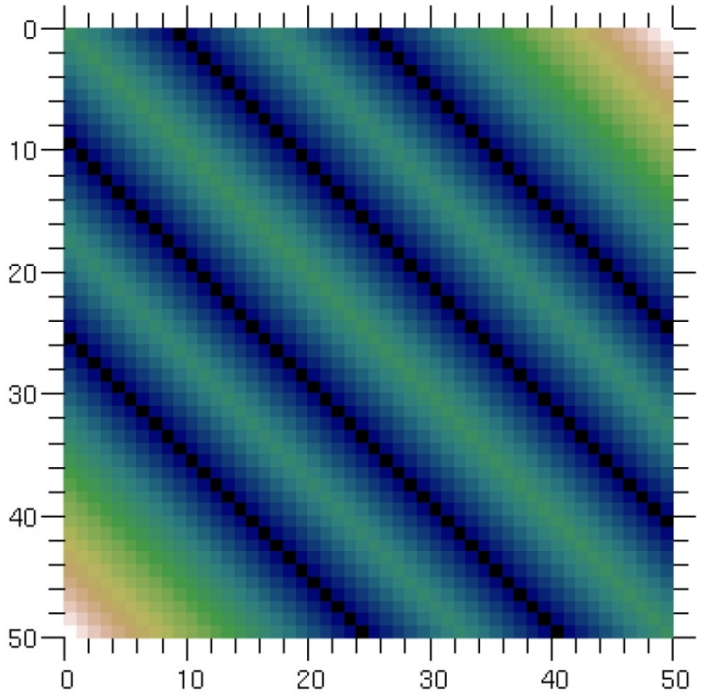
Observations:

- selection bias is much smaller than for standard (local) evaluation
- there are no high pheromone values far away from the diagonal

ACO and Simple Problems

Problem: Minimum Cost Permutation

Cost matrix C_2

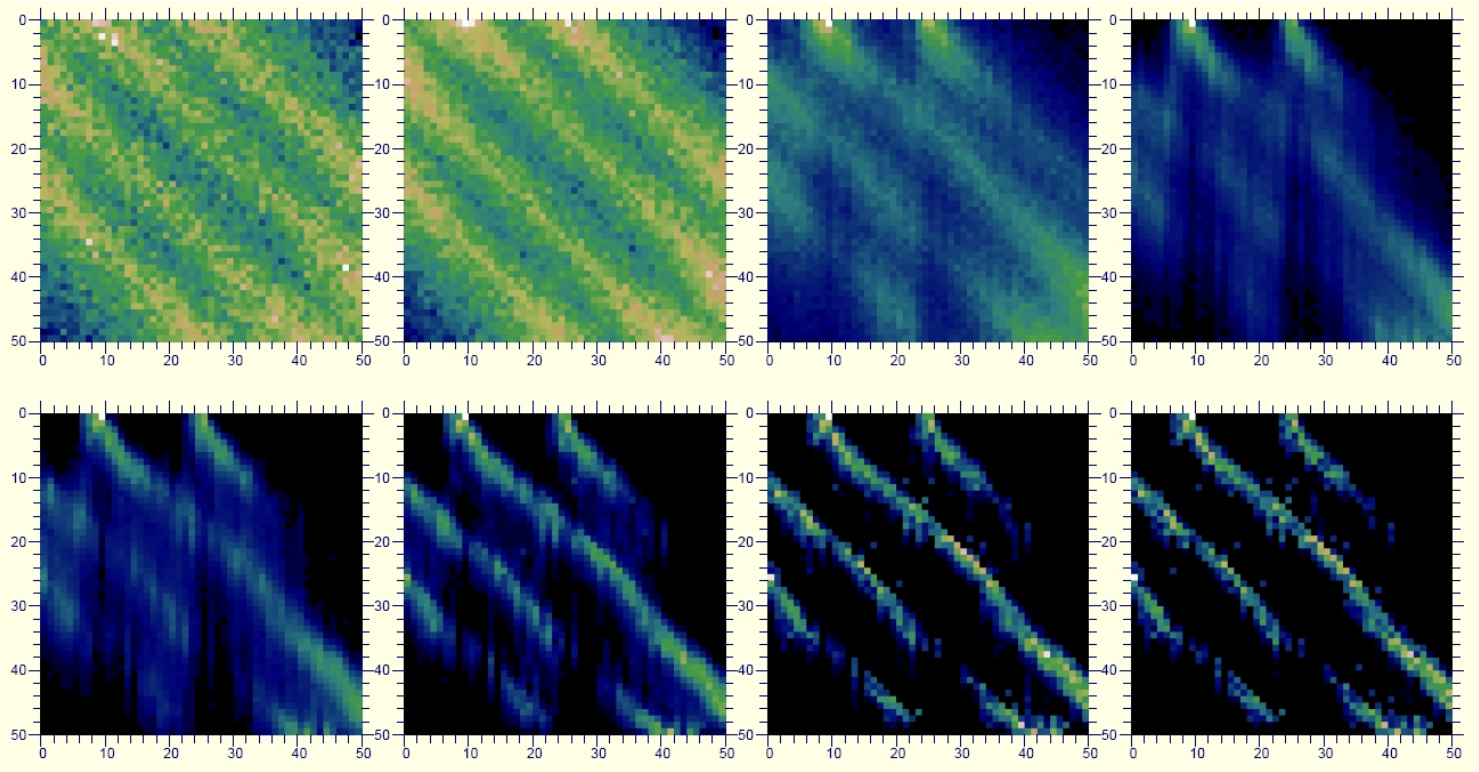


- light color: high cost
- dark color: low cost

Observation: Minimum Cost Permutation problem with C_2 has several optimal solutions

ACO and Simple Problems

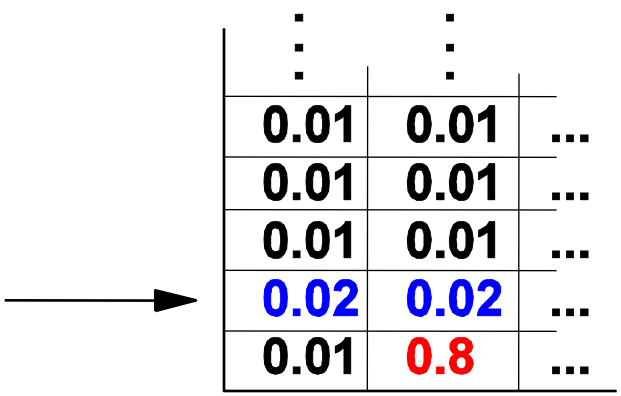
Changes of pheromone matrix with summation evaluation:



brighter colours correspond to more pheromone

ACO and Simple Problems

Consider the following situation: items 1 and 2 have the same pheromone value in the second last row of a pheromone matrix



Which item should be chosen?

→ 0.02 is relatively more pheromone with respect to the sum of the pheromone values in the last two rows for column 1 (0.03) than for column 2 (0.82)

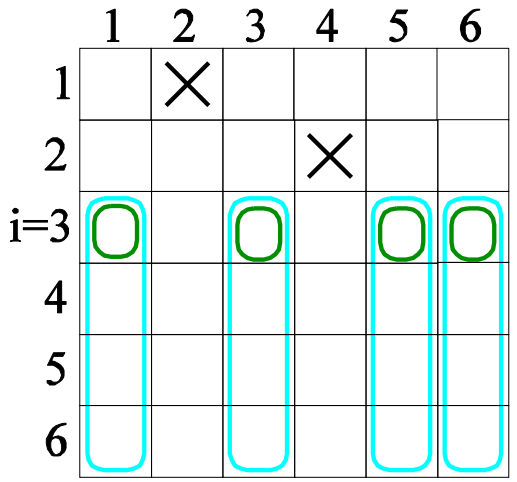
→ item 1 should be chosen

Idea: Normalize each pheromone value relative to the total amount of pheromone in the rest of its column

ACO and Simple Problems

Relative pheromone evaluation method: [Merkle,Middendorf,2002]

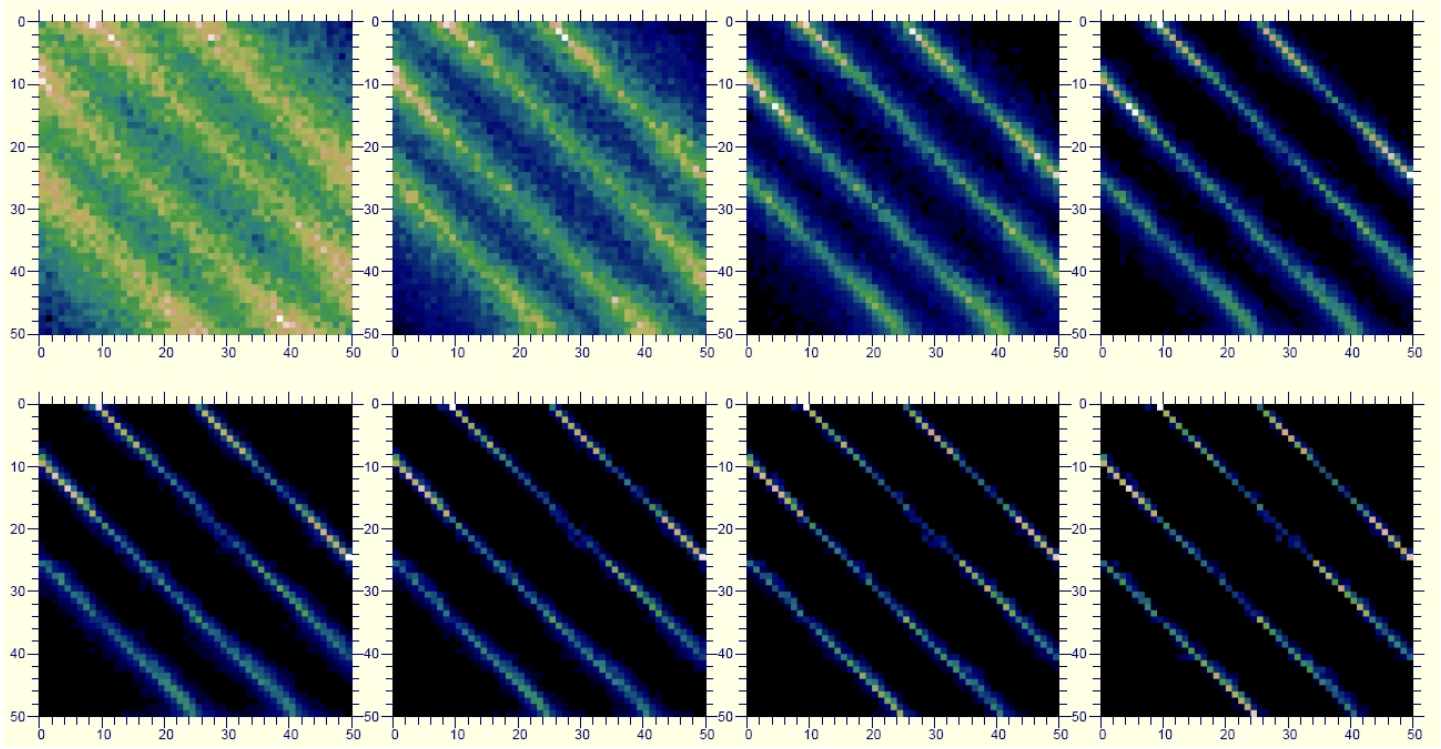
$$\tau_{ij}^{\star\star} := \frac{1}{\sum_{h=i}^{n} \tau_{hj}} \cdot \tau_{ij}$$



$$p_{ij} = \frac{\tau_{ij}^{\star\star}\eta_{ij}}{\sum_{k\in\mathcal{S}} \tau_{ik}^{\star\star}\eta_{ik}}$$

ACO and Simple Problems

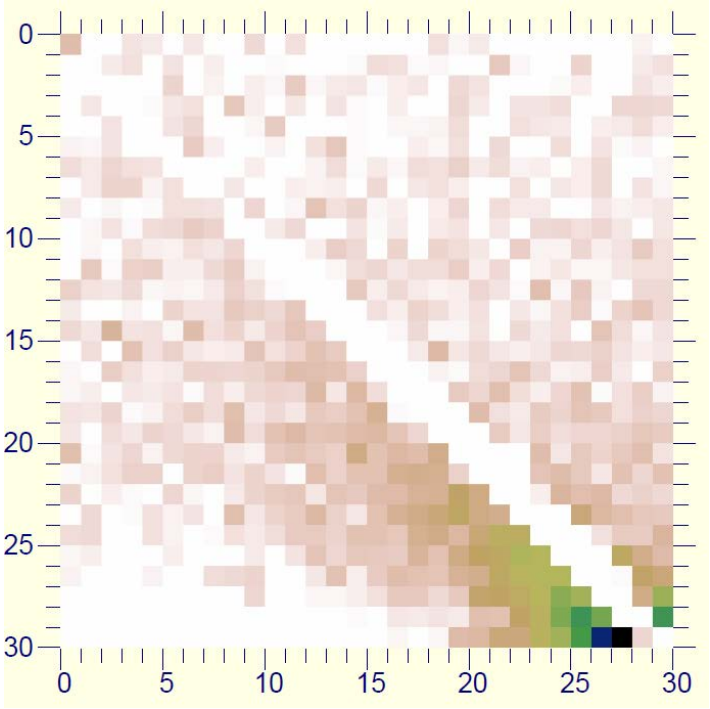
Changes of pheromone matrix with relative pheromone evaluation method



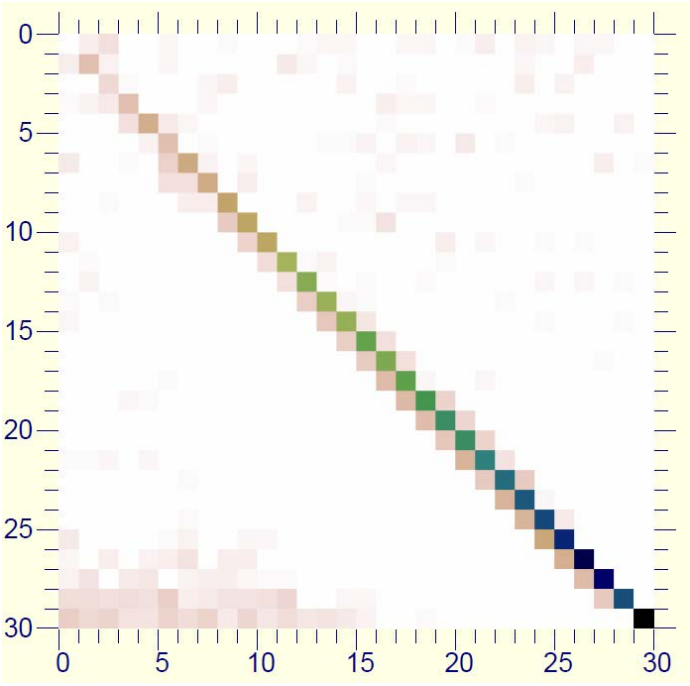
brighter colours correspond to more pheromone

ACO and Simple Problems

Same experiment as on slide 45 but with relative pheromone evaluation



positive selection bias



negative selection bias

Comparison of selection bias:	standard local evaluation	vs. relativ evaluation
Sum of absolute bias values	349572	72474
Maximum positive bias	22518	3704
Maximum negative bias	-4328	-1687

ACO and Simple Problems

Modified Version of Relative Pheromone Evaluation:

Problem: If amount of pheromone added during update is equal in each place, then places with corresponding small sum of pheromone values in the rest of their column receive larger relative advantage to be selected

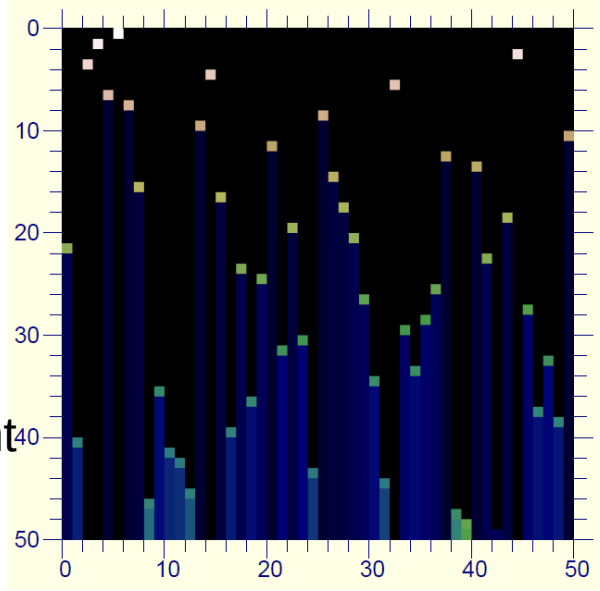
Requirements: Add so much pheromone that

- each updated value $\tau_{ij}^{\star\star}$ is increased by the same value
- each column receives the same amount of pheromone

→ add pheromone not only on places corresponding to the solution itself but also on all places below in the same column

Example: Resulting amount of pheromone update for a pheromone matrix where all pheromone values are equal (selected items can be seen by the bright squares)

(brighter colour = more pherome added)



ACO and Simple Problems

Single Machine Total Earliness Multiple Due Date Problem

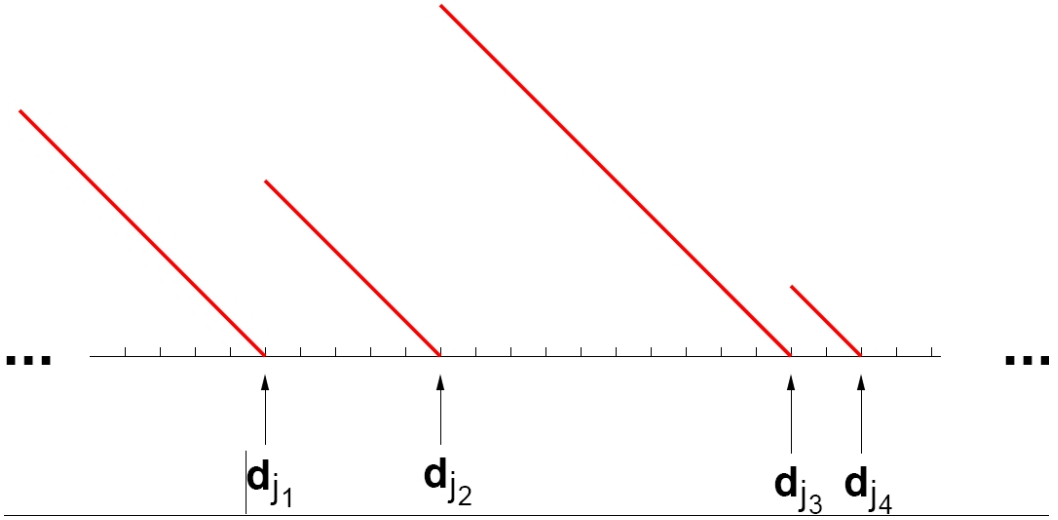
Given: n Jobs with processing times p_j and every job j has n_j due dates $d_{j,1}>\ldots>d_{j,n_j}$

Find: A non-preemptive one machine schedule, that minimizes

$$E = \sum_{j=1}^{n} \{min_{k=1,\ldots,n_j}\{max\{0, d_{j_k} - C_j\}\}$$

where C_j is the completion time of job j.

Example: costs for job j



ACO and Simple Problems

Test Instances and Parameters:

Processing times p_j taken randomly from [1,10]

Due dates $d_{j,1},\ldots,d_{j,nj}$ taken randomly from $[1, \sum_{j=1}^{n} p_j]$

Number of ants m=10, α=1, ρ=0.01, influence of heuristic {ß=0,β=1}

Number of due dates {2,3,5,10}

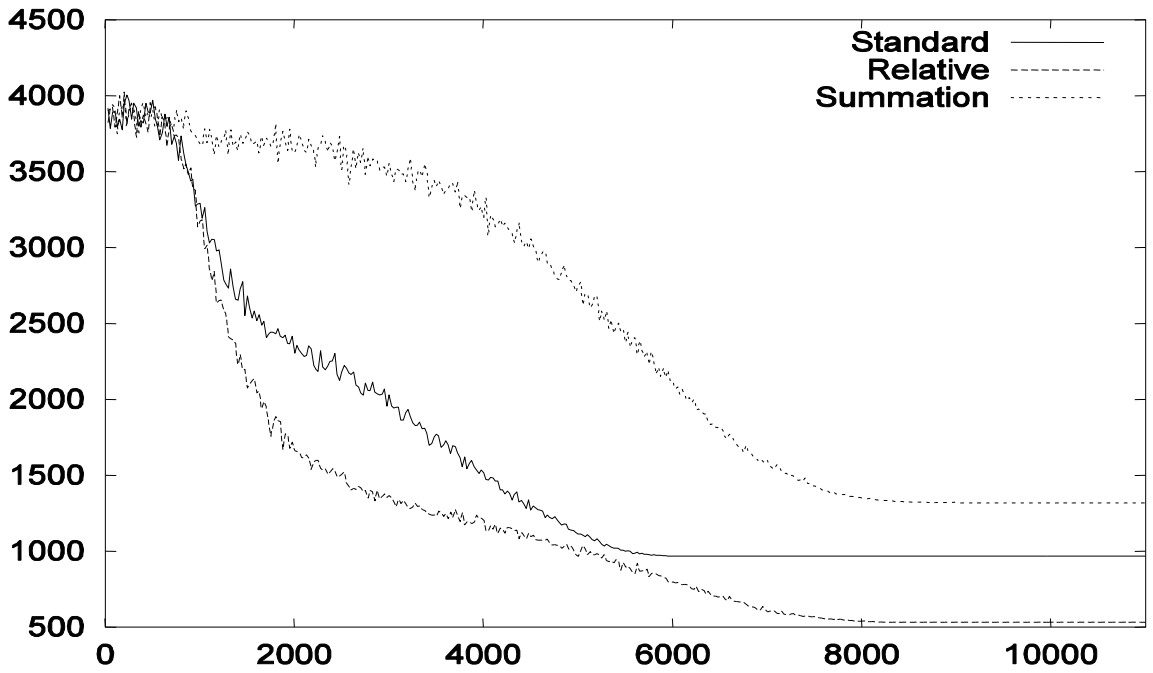
Heuristic: prefer jobs that will cause small earliness costs when scheduled

$$\eta_{ij} = \frac{1}{\min_{k=1\ldots n_j}\{\max\{1, d_{j_k} - (t + p_j)\}\}}$$

where t is the completion time of the so far chosen partial schedule

ACO and Simple Problems

Results for ß=0:

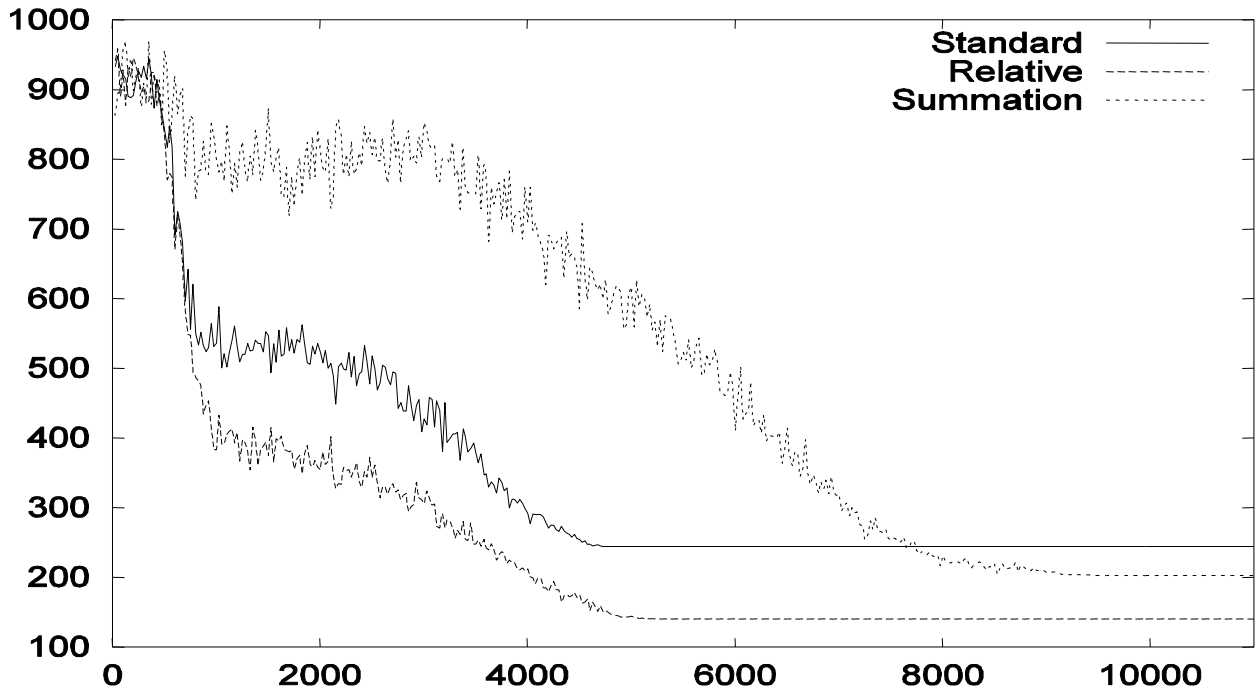


\# DD	Standard	Summation	Relative	Relative (Var.)
2	3782.5 (5025)	4281.0 (8314)	1710.0 (6994)	3097.7 (5300)
3	2441.1 (5068)	3496.4 (8433)	1182.7 (7001)	2087.2 (5368)
5	1793.2 (5082)	2420.8 (8266)	862.4 (7363)	1547.4 (5675)
10	964.2 (5537)	1317.0 (8527)	512.9 (7453)	781.2 (5731)

in brackets: iteration when best value was found

ACO and Simple Problems

Results for ß=1:



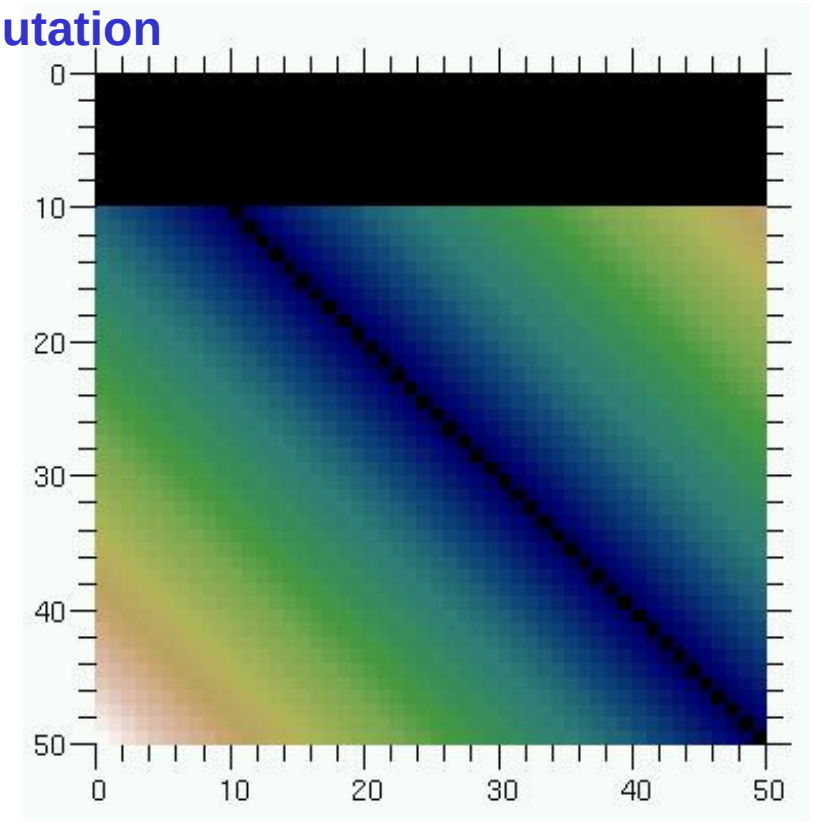
\# DD	Standard	Summation	Relative	Relative (Var.)
2	755.1 (3261)	534.1 (5062)	460.2 (3292)	498.7 (3260)
3	555.0 (2783)	550.5 (5933)	341.7 (3453)	389.3 (3277)
5	353.6 (3207)	271.6 (6579)	211.3 (3979)	254.4 (3730)
10	236.9 (3864)	199.2 (7622)	137.7 (4401)	158.4 (3833)

in brackets: iteration when best value was found

ACO and Simple Problems

Problem: Minimum Cost Permutation

Cost matrix C_3



light color: high cost; dark color: low cost

Observation: every solution that has item j on place j for $j=11,\ldots,50$ is optimal

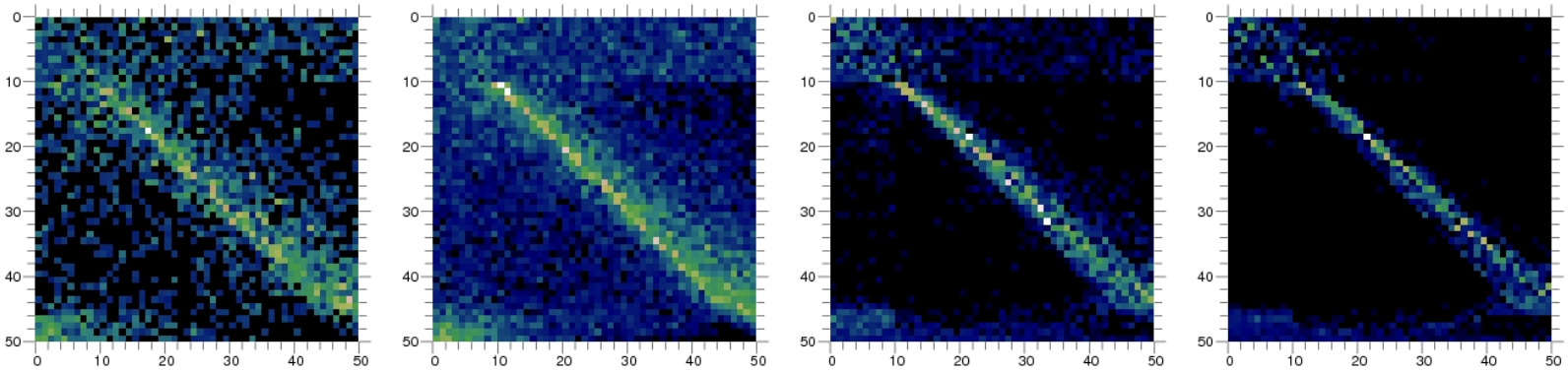
Question: Which decision should the ants make first

ACO and Simple Problems

Forward ants:

→ Decision sequence: π(1),π(2),…,π(50)

Changes of pheromone matrix with relative pheromone evaluation method



brighter colours correspond to more pheromone

Solution quality is poor

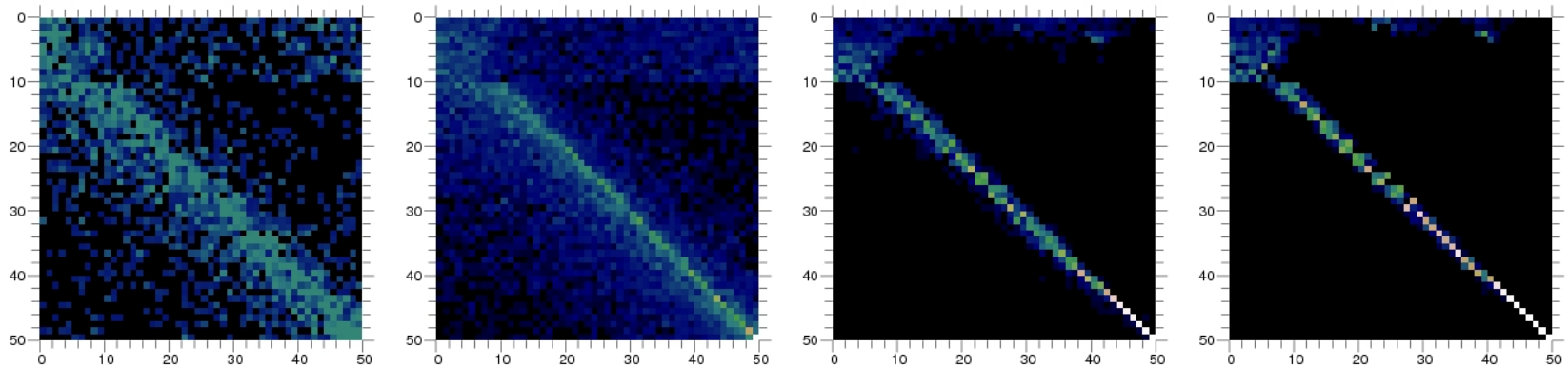
Observation: the pheromone matrix „converges“ from the top rows

ACO and Simple Problems

Backward ants:

→ Decision sequence: π(50),π(49),…,π(1)

Changes of pheromone matrix with relative pheromone evaluation method



brighter colours correspond to more pheromone

Solution quality is moderate

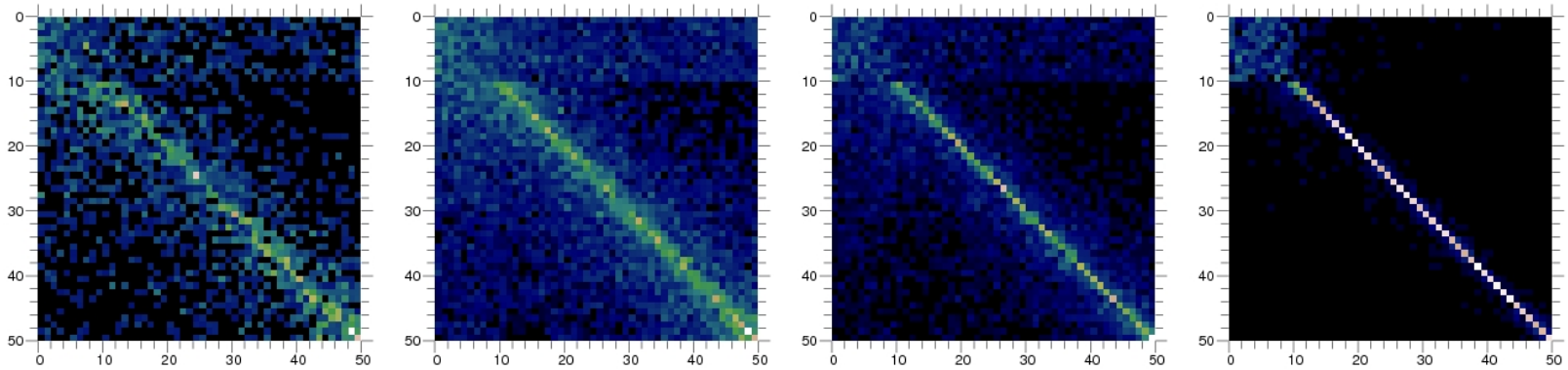
Observation: the pheromone matrix „converges“ from the bottom rows

ACO and Simple Problems

Random ants:

→ Decision sequence of each ant is randomly chosen, i.e.
for each ant k exists a permutation σ_k and the decision sequence is $\pi(\sigma_k(1)),\pi(\sigma_k(2)),\ldots,\pi(\sigma_k(5)0)$

Changes of pheromone matrix with relative pheromone evaluation method



brighter colours correspond to more pheromone

Solution quality is good

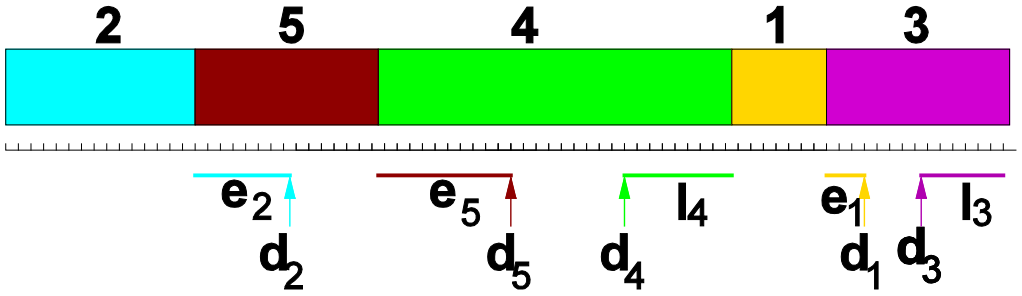
ACO and Simple Problems

Single Machine Total Weighted Deviation Problem :

Given: n Jobs with processing times p_i, due dates d_i, lateness weights l_i, and earliness weights e_i.

Find: Permutation of the jobs with minimal sum of weighted due date violations.

Example: Schedule for an instance with 5 jobs

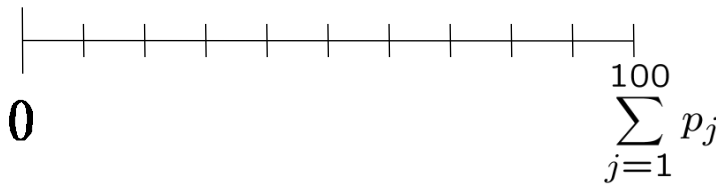


ACO and Simple Problems

Test Instances:

Processing times p_i randomly from in [10,100]

Due dates d_i randomly from: $[(TF-0.2)\cdot\sum_{j=1}^{100} p_j, (TF+0.2)\cdot\sum_{j=1}^{100} p_j]$



with TF={**0.2**,**0.4**,**0.6**,**0.8**}

Earliness weights e_i randomly from [1e,2e], tardiness weights t_i randomly from [1t,2t]

15 instances for each (e,t) $\in$ {(5,1),(3,1),(1,1),(1,3),(1,5)}

ACO Parameters: number of ants m=10, evaporation ρ=0.01,
every run was stopped after 20000 generations

ACO and Simple Problems

Notation:

F-A = forward ants

ΣF-A = forward ants with summation pheromone evaluation

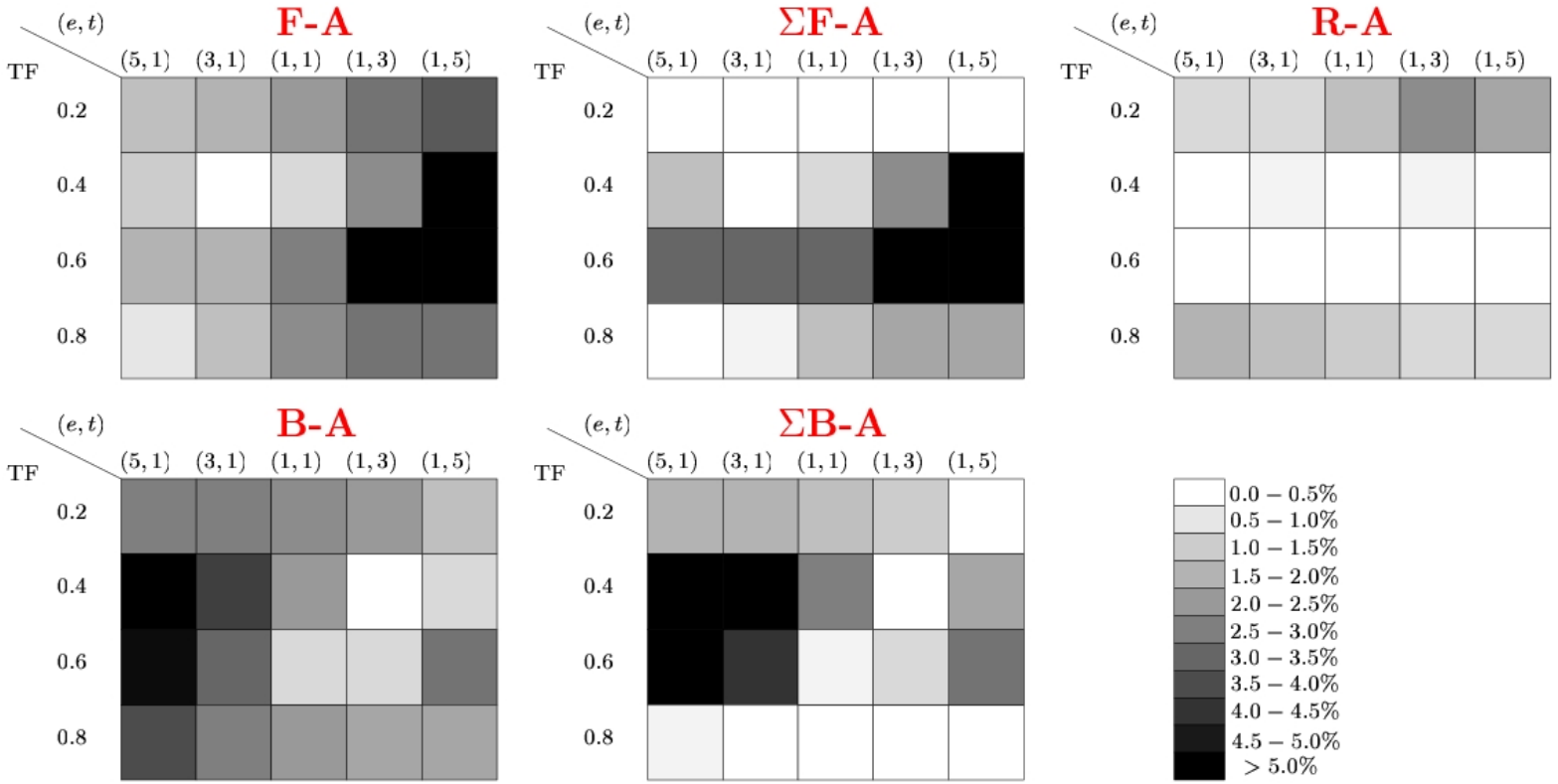
B-A = backward ants

ΣB-A = backward ants with summation pheromone evaluation

R-A = random ants

ACO and Simple Problems

Results:



For each test parameter combination (TF,(e,t)): relativ deviation (in percent) of solution quality with respect to the solution quality of the best type of ants

Example: for (TF,(e,t))=(0.8,(1,5)) the ΣB-A ants are best

ACO and Simple Problems

Problem: For many problems (e.g. many scheduling problems) good heuristics exist only when the ants construct prefixes (as the forward ants do) or suffixes (as the backward ants do) of a permutation

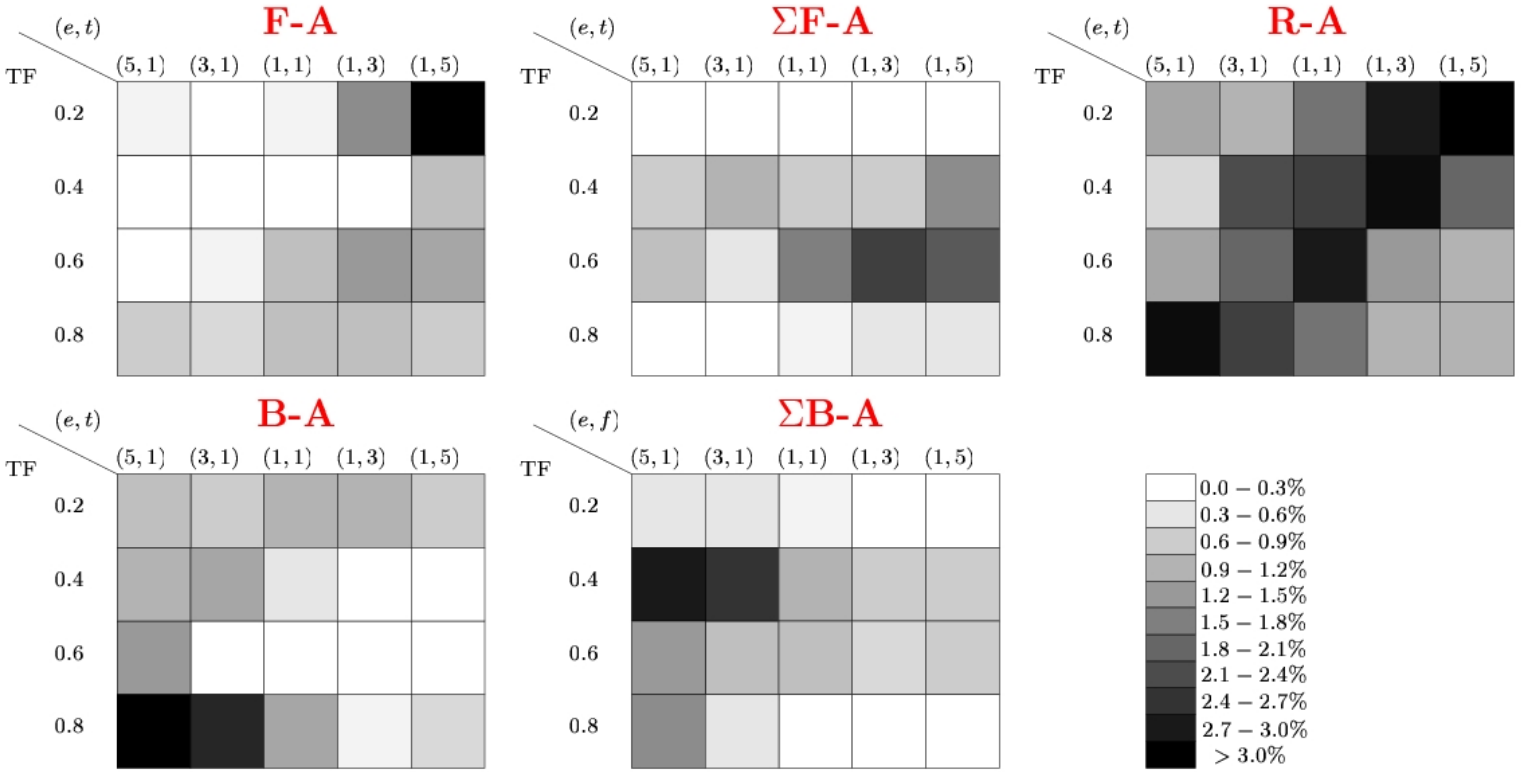
Idea: use random ants **(**without heuristic) and forward or backward ants (with heuristic) together

… but using them together in the same iteration could be a problem when one type of ants is weaker than the other

→ use random ants and forward or backward ants alternately in different iterations

ACO and Simple Problems

Results: now -> every second generation has random ants



For each test parameter combination (TF,(e,t)): relativ deviation (in percent) of solution quality with respect to the solution quality of the best type of ants

For every parameter combination: the two type ants algorithms are better than all single type ants algorithms