Schwarmintelligenz

Sarah Götz · Philipp Fuchs
Andreas Auer · Moritz Kondmann

Einführung

Definition

Schwarmintelligenz beschreibt das kollektive Verhalten von dezentralisierten, selbstorganisierten Systemen.

Einführung

Definition

Schwarmintelligenz beschreibt das kollektive Verhalten von dezentralisierten, selbstorganisierten Systemen.

Dezentralisierung: Individuen handeln autonom und basieren Entscheidungen auf lokalen Informationen.

Selbstorganisation: Komplexes Verhalten entsteht durch einfache Regeln.

Emergenz: Zusammenspiel vieler Individuen führt zu globalem Verhalten.

Beispiele und Anwendungen

Beispiele aus der Natur:

- Ameisenkolonien: Finden kürzester Wege durch Pheromonspuren.
- Vogelschwärme: Suche nach dem besten Rastplatz.
- Fischschwärme: Suche nach Nahrung durch lokale Interaktionen.

Beispiele und Anwendungen

Beispiele aus der Natur:

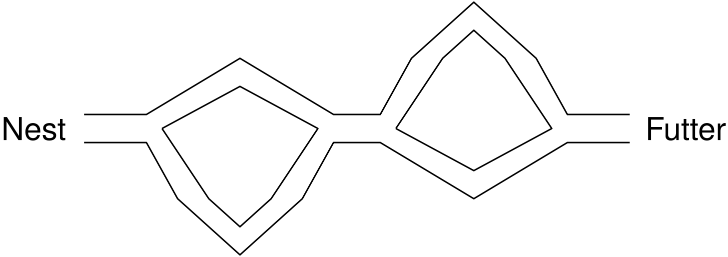
- Ameisenkolonien: Finden kürzester Wege durch Pheromonspuren.
- Vogelschwärme: Suche nach dem besten Rastplatz.
- Fischschwärme: Suche nach Nahrung durch lokale Interaktionen.

Relevanz in der Informatik:

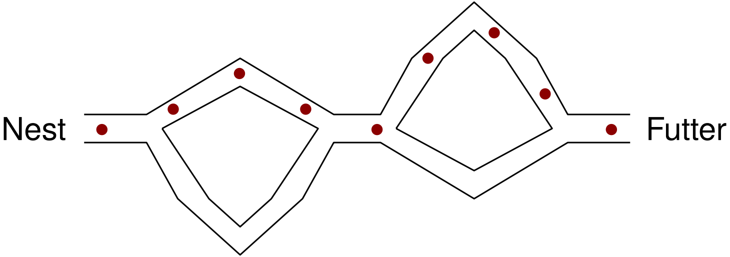
- Lösung von Optimierungsproblemen.
- Autonome Systeme und Robotik.
- Netzwerkmanagement.

Ameisenalgorithmus (ACO)

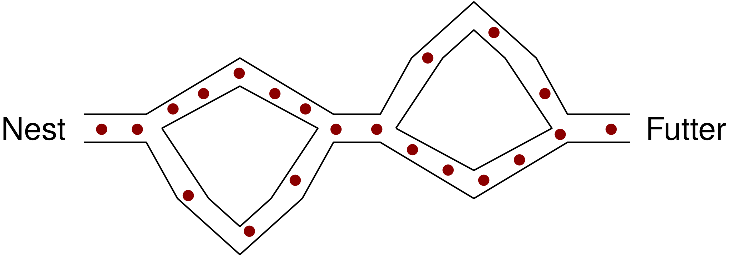
Biologische Inspiration



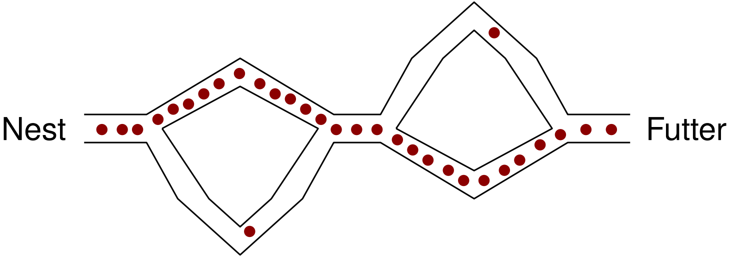
Biologische Inspiration



Biologische Inspiration



Biologische Inspiration



Handelsreisenden Problem (TSP)

Gesucht:

- Kürzeste Tour um alle Städte ausgehend von der Heimatstadt genau einmal zu besuchen und zur Heimatstadt zurückzukehren.

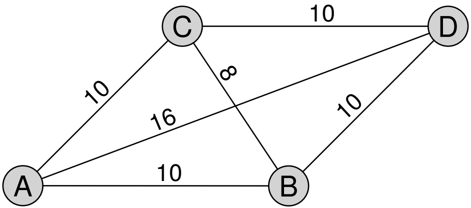
Handelsreisenden Problem (TSP)

Gesucht:

- Kürzeste Tour um alle Städte ausgehend von der Heimatstadt genau einmal zu besuchen und zur Heimatstadt zurückzukehren.

Grundidee:

- Beschreibung des TSP als vollständigen gewichteten Graphen.
- Finde den kürzesten Hamiltonschen Kreis.



Lösungsansatz: Brute Force

$$\text{Anzahl der möglichen Touren} = \frac{(n-1)!}{2}$$

Lösungsansatz: Brute Force

$$\text{Anzahl der möglichen Touren} = \frac{(n-1)!}{2}$$

- 4 Städte $\rightarrow$ 3 Touren,
- 5 Städte $\rightarrow$ 12 Touren,

Lösungsansatz: Brute Force

$$\text{Anzahl der möglichen Touren} = \frac{(n-1)!}{2}$$

- 4 Städte $\rightarrow$ 3 Touren,
- 5 Städte $\rightarrow$ 12 Touren,
- 10 Städte $\rightarrow$ 181 440 Touren,

Lösungsansatz: Brute Force

$$\text{Anzahl der möglichen Touren} = \frac{(n-1)!}{2}$$

- 4 Städte $\rightarrow$ 3 Touren,
- 5 Städte $\rightarrow$ 12 Touren,
- 10 Städte $\rightarrow$ 181 440 Touren,
- 15 Städte $\rightarrow$ 43 589 145 600 Touren,

Lösungsansatz: Brute Force

$$\text{Anzahl der möglichen Touren} = \frac{(n-1)!}{2}$$

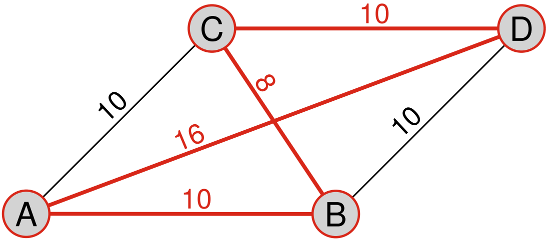
- 4 Städte → 3 Touren,
- 5 Städte → 12 Touren,
- 10 Städte → 181 440 Touren,
- 15 Städte → 43 589 145 600 Touren,
- **20 Städte → 60 822 550 204 416 000 Touren.**

Lösungsansatz: Greedy Algorithmus Nearest Neighbor

- Trifft lokal beste Entscheidung um Tour zu konstruieren.
- Findet nicht immer die global beste Lösung.

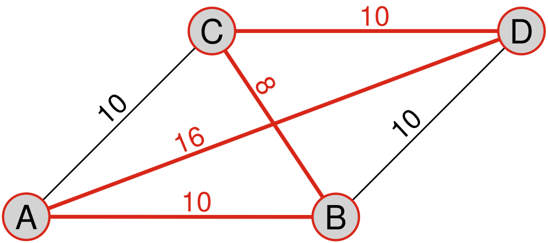
Lösungsansatz: Greedy Algorithmus Nearest Neighbor

- Trifft lokal beste Entscheidung um Tour zu konstruieren.
- Findet nicht immer die global beste Lösung.



Lösungsansatz: Greedy Algorithmus Nearest Neighbor

- Trifft lokal beste Entscheidung um Tour zu konstruieren.
- Findet nicht immer die global beste Lösung.



$$\begin{aligned} \text{Länge der gefundenen Tour} &= 44, \\ \text{Länge der optimalen Tour} &= 40. \end{aligned}$$

Lösungsansatz: ACO

- Künstliche Ameisen,
- Heuristische Informationen (z.B. Distanz),
- Historische Informationen (Pheromone).

Lösungsansatz: ACO

- Künstliche Ameisen,
- Heuristische Informationen (z.B. Distanz),
- Historische Informationen (Pheromone).

Ablauf:

1. Pheromonwerte aller Kanten initialisieren.

Lösungsansatz: ACO

- Künstliche Ameisen,
- Heuristische Informationen (z.B. Distanz),
- Historische Informationen (Pheromone).

Ablauf:

1. Pheromonwerte aller Kanten initialisieren.
2. Ameisen zufällig auf die Knoten des Graphen verteilen.

Lösungsansatz: ACO

- Künstliche Ameisen,
- Heuristische Informationen (z.B. Distanz),
- Historische Informationen (Pheromone).

Ablauf:

1. Pheromonwerte aller Kanten initialisieren.
2. Ameisen zufällig auf die Knoten des Graphen verteilen.
3. Jede Ameise konstruiert eigene, wahrscheinlichkeitsbasierte Lösung und ermittelt deren Qualität

Lösungsansatz: ACO

- Künstliche Ameisen,
- Heuristische Informationen (z.B. Distanz),
- Historische Informationen (Pheromone).

Ablauf:

1. Pheromonwerte aller Kanten initialisieren.
2. Ameisen zufällig auf die Knoten des Graphen verteilen.
3. Jede Ameise konstruiert eigene, wahrscheinlichkeitsbasierte Lösung und ermittelt deren Qualität
4. Pheromonwerte aktualisieren

Lösungsansatz: ACO

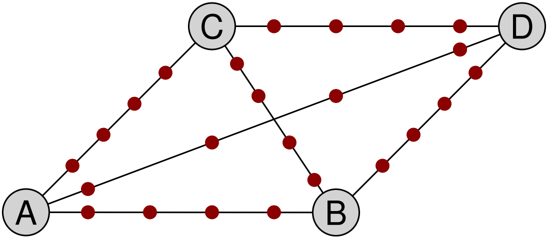
- Künstliche Ameisen,
- Heuristische Informationen (z.B. Distanz),
- Historische Informationen (Pheromone).

Ablauf:

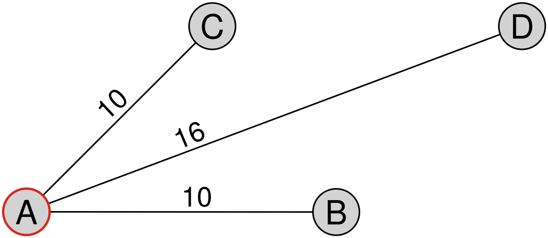
1. Pheromonwerte aller Kanten initialisieren.
2. Ameisen zufällig auf die Knoten des Graphen verteilen.
3. Jede Ameise konstruiert eigene, wahrscheinlichkeitsbasierte Lösung und ermittelt deren Qualität
4. Pheromonwerte aktualisieren

Wiederholung der Schritte 2-4 bis die Lösung konvergiert.

Initialisierung der Pheromone



Konstruktion der Lösung



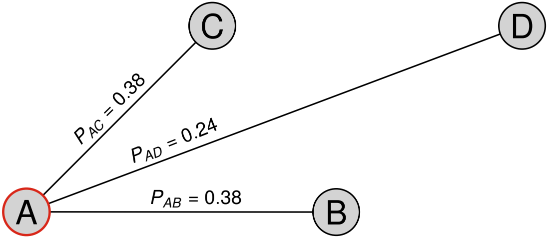
Berechnung der Wahrscheinlichkeit

Wahrscheinlichkeitsregel

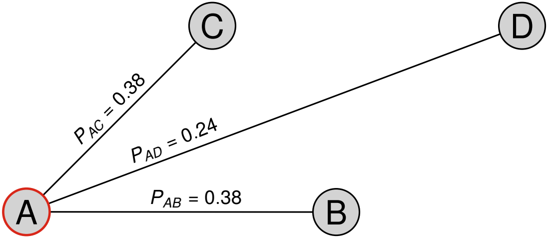
$$P_{ij} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{k \in N_i} \tau_{ik}^{\alpha} \cdot \eta_{ik}^{\beta}}$$

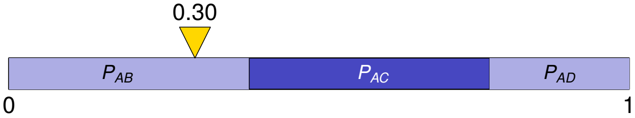
- τ_{ij}: Pheromonkonzentration,
- η_{ij}: Heuristische Information (z.B. 1/Distanz),
- α, β: Gewichtungsfaktoren,
- N_i: Menge der Nachbarknoten, die noch nicht besucht wurden.

Konstruktion der Lösung

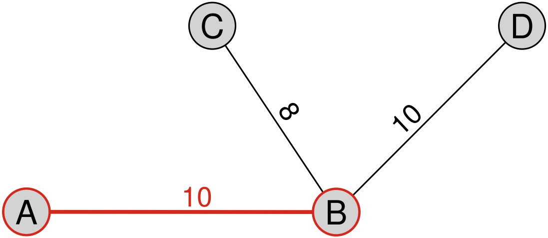


Konstruktion der Lösung

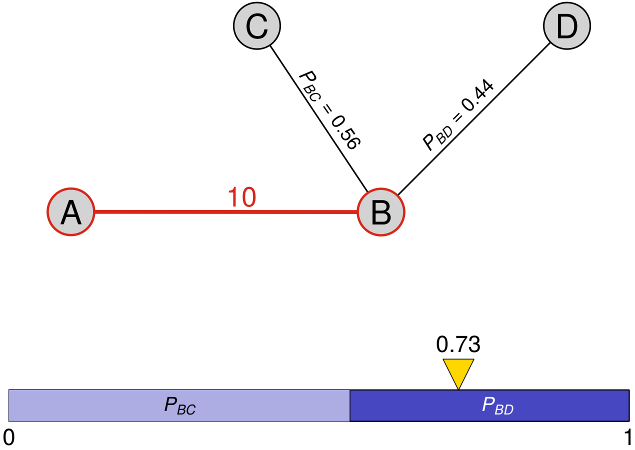




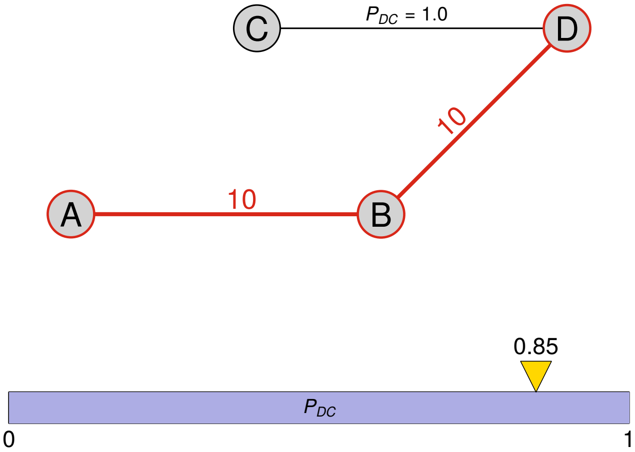
Konstruktion der Lösung



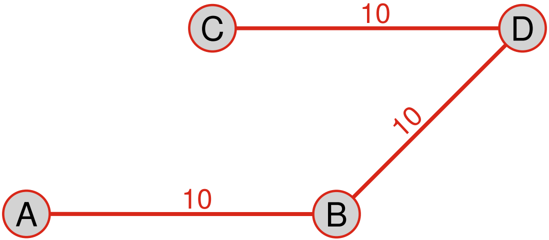
Konstruktion der Lösung



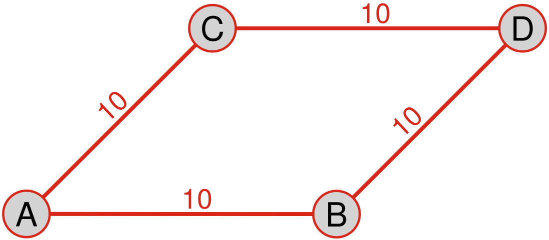
Konstruktion der Lösung



Konstruktion der Lösung



Konstruktion der Lösung



Pheromonwerte aktualisieren

Verdunstung

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

- ρ: Verdunstungsrate,
- τ_{ij}: Pheromonwert.

Pheromonwerte aktualisieren

Verdunstung

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

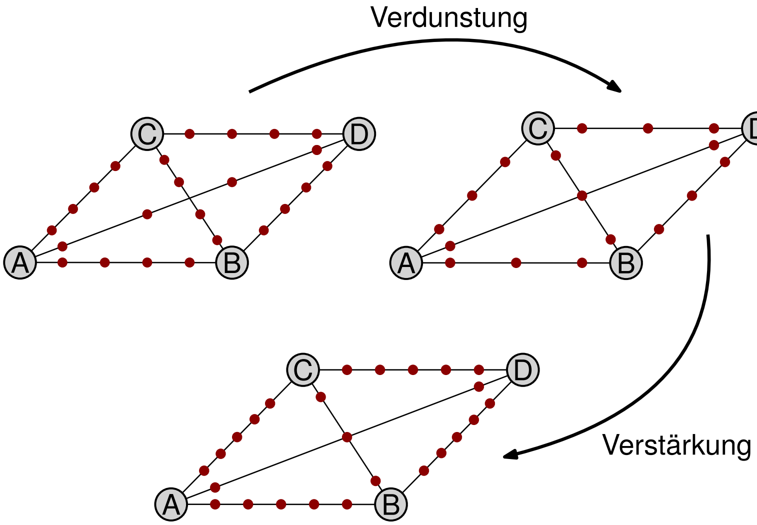
- ρ: Verdunstungsrate,
- τ_{ij}: Pheromonwert.

Verstärkung

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \frac{Q}{L_k}$$

- Q: Pheromonmenge,
- L_k: Länge der Lösung der k-ten Ameise.

Pheromonwerte aktualisieren



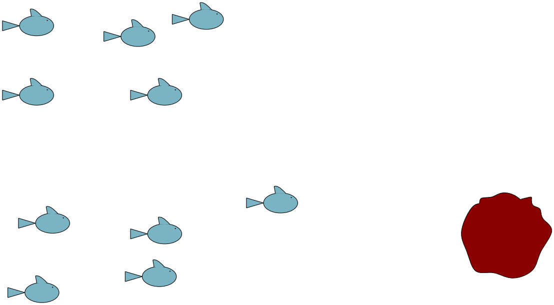
Weitere Anwendungsmöglichkeiten

Stärke des ACO: Lösungen für komplexe kombinatorische Optimierungsprobleme.

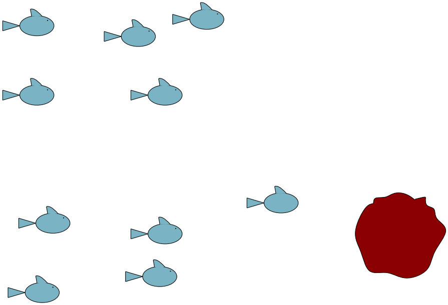
- **Logistik:** Optimierung von Lieferketten und Verkehrsfluss.
- **Netzwerk-Routing:** Datenpakete in Netzwerken effizient leiten.
- **Planung:** Zuteilung knapper Ressourcen.

Partikelschwarmoptimierung (PSO)

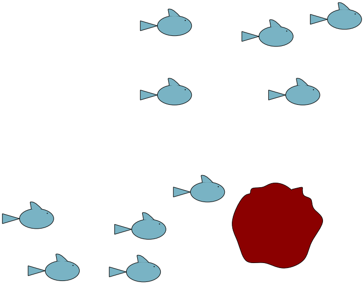
Inspiration aus der Natur



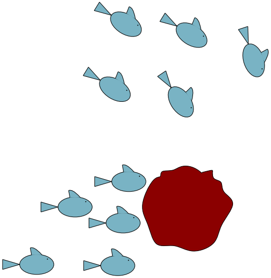
Inspiration aus der Natur



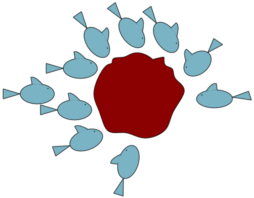
Inspiration aus der Natur



Inspiration aus der Natur



Inspiration aus der Natur



Aufbau des PSO

Hauptkomponenten

- Suchraum,
- Partikel,
- Abbruchbedingung.

Aufbau des PSO

Hauptkomponenten

- Suchraum,
- Partikel,
- Abbruchbedingung.

Partikel

- Aktuelle Position,
- Bewegungsvektor,
- Gedächtnis für persönlichen und globalen Bestwert.

Ablauf des Algorithmus

Beginn:

- Der Suchraum wird festgelegt.
- Verteilen der Partikel im Suchraum.
- Die Abbruchbedingung wird festgelegt.

Ablauf des Algorithmus

Beginn:

- Der Suchraum wird festgelegt.
- Verteilen der Partikel im Suchraum.
- Die Abbruchbedingung wird festgelegt.

Ablauf:

1. Jedes Partikel bewegt sich nach seinem Bewegungsvektor.
2. Der Schwarm tauscht untereinander die individuellen Bestwerte aus.
3. Jedes Partikel berechnet einen neuen Bewegungsvektor.

Wiederholung der Schritte 1-3 bis Abbruchbedingung erreicht ist.

Berechnung eines neuen Punktes

P_i^{t+1}

- P_i^{t+1}: Standort bei Iterationsschritt $t + 1$,

Berechnung eines neuen Punktes

$P_i^{t+1} = P_i^t$

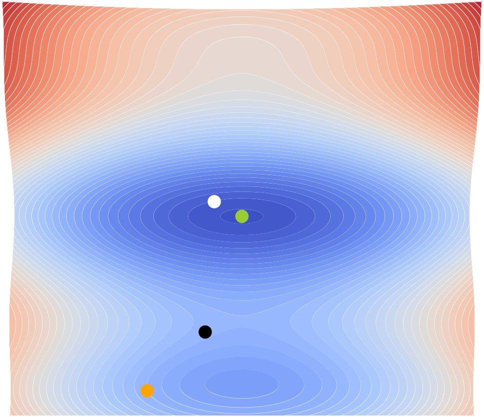
- P_i^{t+1}: Standort bei Iterationsschritt $t+1$,
- P_i^t : Standort bei Iterationsschritt t,

Berechnung eines neuen Punktes

$P_i^{t+1} = P_i^t + V_i^{t+1}$

- P_i^{t+1}: Standort bei Iterationsschritt $t+1$,
- P_i^t : Standort bei Iterationsschritt t,
- V_i^{t+1}: Bewegungsvektor für Iterationsschritt $t+1$.

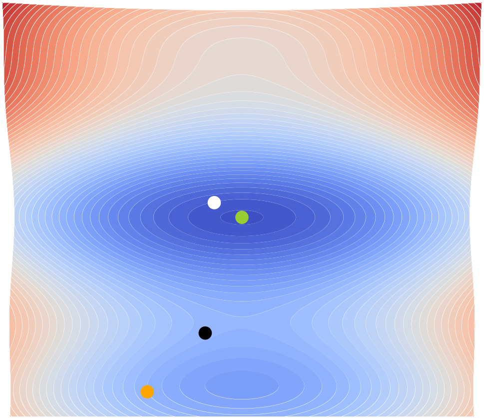
Funktionsaufbau



- Minimum,
- glob. Bestwert,
- pers. Bestwert,
- aktuelle Position.

Berechnung des Bewegungsvektors

Funktionsaufbau

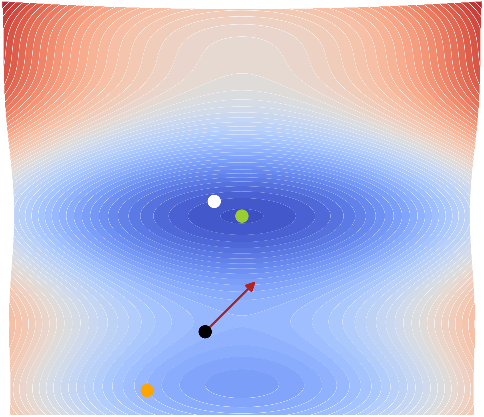


- Minimum,
- glob. Bestwert,
- pers. Bestwert,
- aktuelle Position.

Berechnung des Bewegungsvektors

V_i^{t+1}

Funktionsaufbau



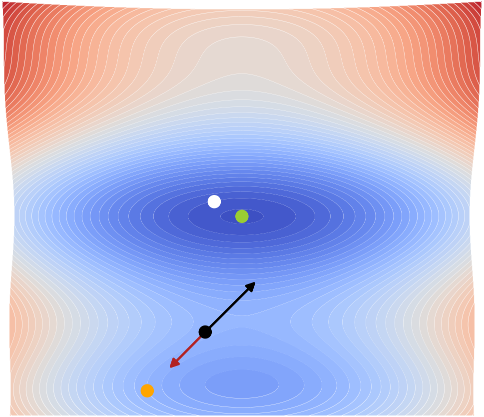
- Minimum,
- glob. Bestwert,
- pers. Bestwert,
- aktuelle Position.

- ω: Trägheitsmoment,
- V_i^t: letzte Bewegung,

Berechnung des Bewegungsvektors

$$V_i^{t+1} = \omega V_i^t$$

Funktionsaufbau



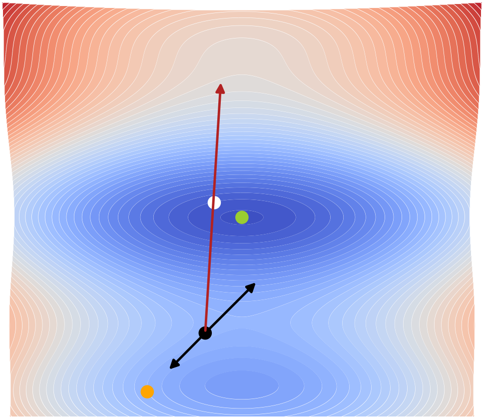
- Minimum,
- glob. Bestwert,
- pers. Bestwert,
- aktuelle Position.

- c_1: kognitive Gewichtung,
- r_1: Zufallsfaktor,
- $P_{pbest(i)^t}$: pers. Bestwert,
- P_i^t: aktuelle Position.

Berechnung des Bewegungsvektors

$$V_i^{t+1} = \omega V_i^t + c_1 r_1 \left(P_{pbest(i)^t} - P_i^t \right)$$

Funktionsaufbau



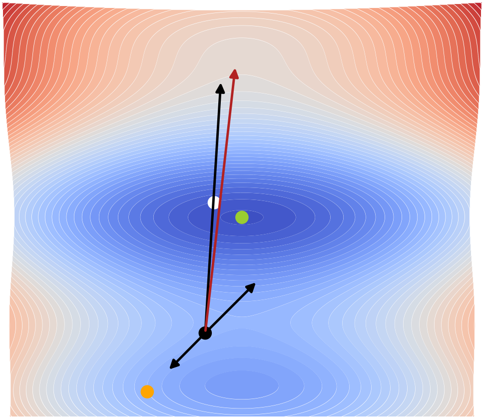
- Minimum,
- glob. Bestwert,
- pers. Bestwert,
- aktuelle Position.

- c_2: soziale Gewichtung,
- r_2: Zufallsfaktor,
- P_{gbest^t}: glob. Bestwert,
- P_i^t: aktuelle Position.

Berechnung des Bewegungsvektors

$$V_i^{t+1} = \omega V_i^t + c_1 r_1 \left(P_{pbest(i)^t - P_i^t} \right) + c_2 r_2 (P^t_{gbest^t} - P_i^t)$$

Funktionsaufbau



- Minimum,
- glob. Bestwert,
- pers. Bestwert,
- aktuelle Position.
- V_i^{t+1}: Bewegungsvektor

Berechnung des Bewegungsvektors

$$V_i^{t+1} = \omega V_i^t + c_1 r_1 \left(P_{pbest(i)^t - P_i^t} \right) + c_2 r_2 (P_{gbest^t}^t - P_i^t)$$

Berechnung des nächsten Punktes

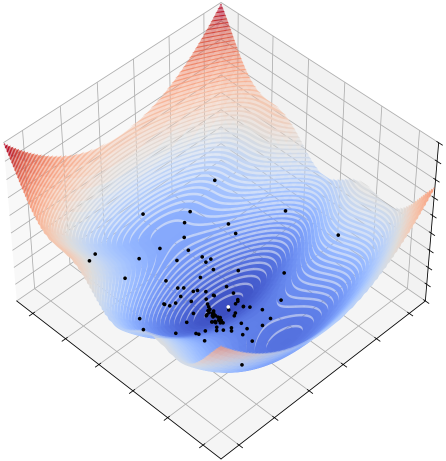
Berechnung eines neuen Punktes

$$P_i^{t+1} = P_i^t + V_i^{t+1}$$

Berechnung des Bewegungsvektors

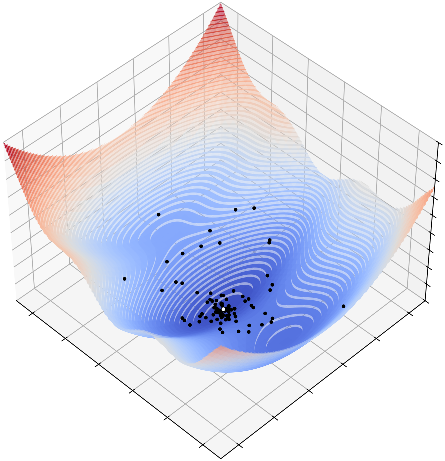
$$V_i^{t+1} = \omega V_i^t + c_1 r_1 \left(P_{pbest(i)^t - P_i^t} \right) + c_2 r_2 (P^t_{gbest^t} - P_i^t)$$

Konkretes Beispiel



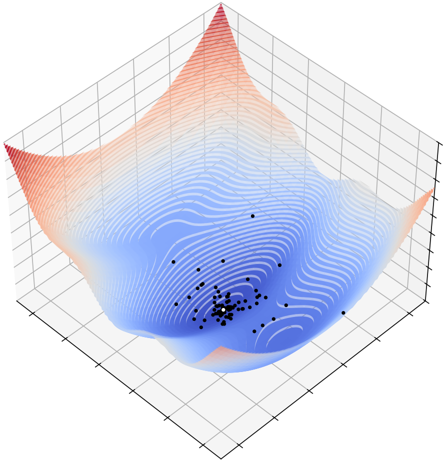
$i \in \{1, 2, \ldots, 100\} \quad t = \ 0$

Konkretes Beispiel



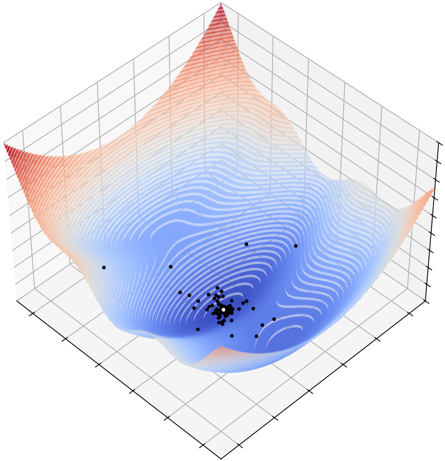
$$i \in \{1, 2, \ldots, 100\} \quad t = 10$$

Konkretes Beispiel



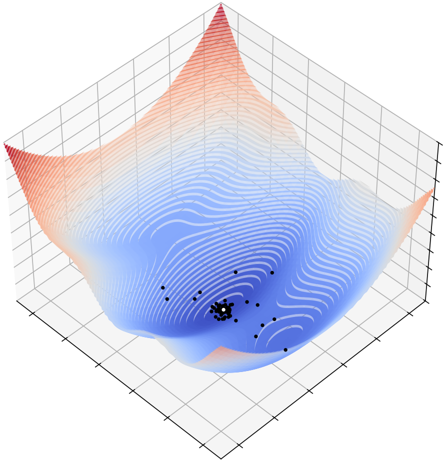
$$i \in \{1, 2, \ldots, 100\} \quad t = 20$$

Konkretes Beispiel



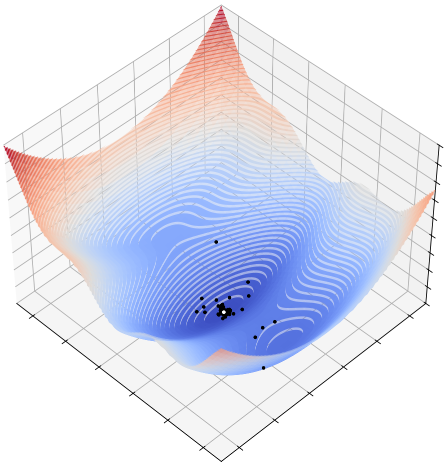
$$i \in \{1, 2, \ldots, 100\} \quad t = 30$$

Konkretes Beispiel



$$i \in \{1, 2, \ldots, 100\} \quad t = 40$$

Konkretes Beispiel



$$i \in \{1, 2, \ldots, 100\} \quad t = 50$$

Vorteile & Nachteile

Nachteile:

- Langsame Konvergenz zum optimalen Ergebnis,
- Schwache lokale Suche,
- Wenn Parameter falsch gewählt werden, besteht die Gefahr das Suchziel nicht zu erreichen.

Vorteile & Nachteile

Nachteile:

- Langsame Konvergenz zum optimalen Ergebnis,
- Schwache lokale Suche,
- Wenn Parameter falsch gewählt werden, besteht die Gefahr das Suchziel nicht zu erreichen.

Vorteile:

- Simpler Algorithmus (wenige Parameter),
- Einfache Implementierung,
- Gewährleistet Funktionalität unabhängig von der Dimension des Suchraumes.

Anwendungsbereiche - Optimierungsprobleme

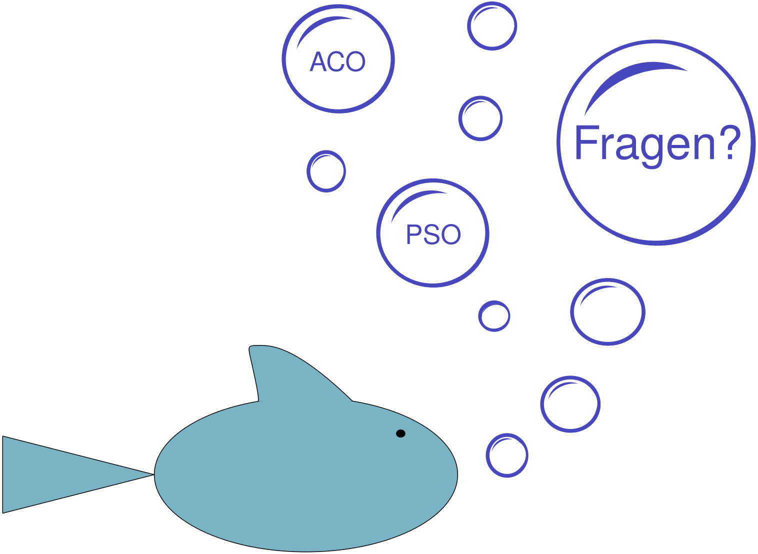
- **Navigation:** Autonome Fahrzeuge.
- **Industrie:** Berechnung der perfekten Konfiguration für Schweißnähte.
- **Produktentwicklung:** Entwicklung neuer Materialien für Luft- und Raumfahrt.

Fazit

- Schwarmintelligenz zeigt, wie wir mit einfachen Regeln komplexe Probleme lösen können.

Fazit

- Schwarmintelligenz zeigt, wie wir mit einfachen Regeln komplexe Probleme lösen können.
- ACO und PSO zeigen die praktische Umsetzung dieser Prinzipien
  - **ACO:** Diskrete Optimierung, z.B. Routenplanung.
  - **PSO:** Kontinuierliche Optimierung, z.B. laufende Anpassung der Route an aktuelle Verkehrsverhältnisse.


ACO
PSO
Fragen?