Programmierkurs

Vorlesung 11

M.Sc. Laslo Hunhold

Department Mathematik/Informatik
Abteilung Informatik
Universität zu Köln

1. Februar 2021

Letzte Vorlesung

- Interval Scheduling
- Minimize Maximum Lateness

Interval Partitioning

Problem

Interval Scheduling

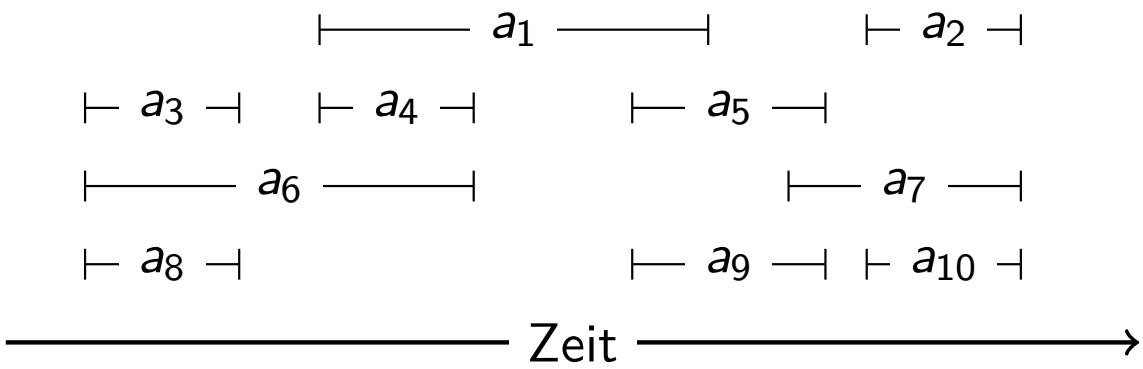
- Verfügen über *eine* Ressource
- Haben mehrere Anfragen
- Anfragen werden entweder angenommen oder abgelehnt

Was wenn wir mehr als eine Ressource gleichzeitig haben?

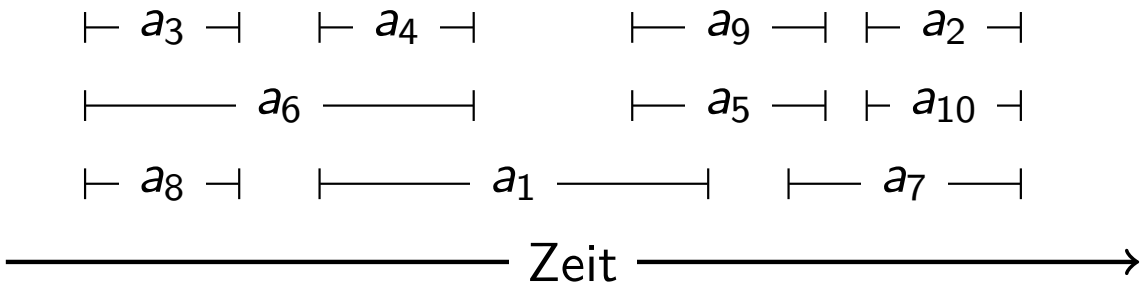
Interval Partitioning

- Mehr als eine Anfrage kann gleichzeitig bedient werden
- Wie viele Ressourcen braucht man, um alle Anfragen kompatibel (also pro Ressource überschneidungsfrei) zu erfüllen?

Beispiel



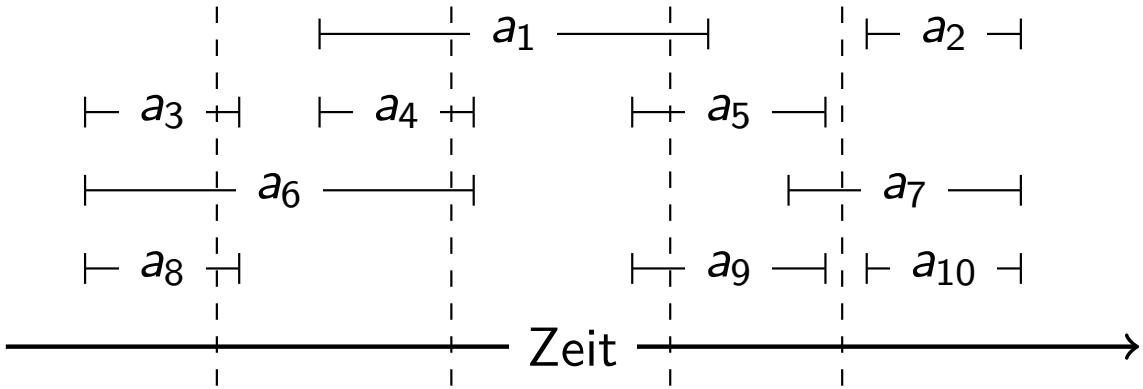
Naiver Ansatz: Brauchen 4 Ressourcen, um alle Anfragen zu bedienen (Eine Ressource pro Zeile)



Optimale Lösung (nach Umsortierung): Bedienung aller Anfragen mit 3 Ressourcen

Tiefe

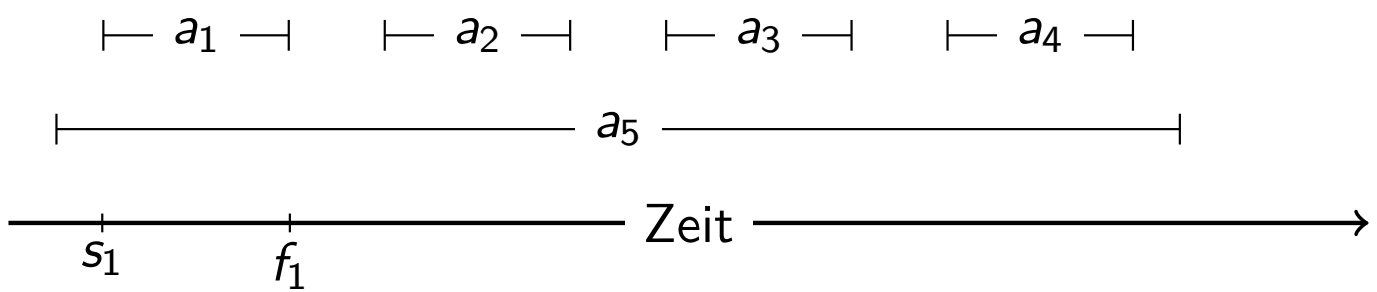
Wie bestimmt man die optimale Anzahl von Ressourcen?



- Schneide Intervallmenge an jeder Stelle vertikal
- Bestimme jeweils Anzahl der getroffenen Intervalle
- Maximum der Werte ist die „Tiefe“ der Intervallmenge

Formalisierung

- Sei $A := \{a_1, \ldots, a_n\} := \{(s_1, f_1), \ldots, (s_n, f_n)\}$ die Menge der Anfragen a_i mit Startzeitpunkt s_i und Endzeitpunkt f_i
- $r :=$ Tiefe$(A) := \max_t |\{a \in A \mid t \in a\}|$ (Anzahl der notwendigen Ressourcen, um alle Anfragen zu erfüllen)
- Sei $S := \{(a_{l_1}, \ell_{l_1}), \ldots, (a_{l_k}, \ell_{l_k})\} \subset A \times \{1, \ldots, r\}$ die Lösungsmenge der bedienten Anfragen (a_k, ℓ_k) mit der Anfrage a_k und dem „Label“ (also der Ressourcenzuweisung) $\ell_k \in \{1, \ldots, r\}$
- S kompatibel $:\Leftrightarrow \forall_{(a,\ell)\neq(\tilde{a},\tilde{\ell})\in S}$: ($a$ überlappt $\tilde{a} \rightarrow \ell \neq \tilde{\ell}$)
- Ziel: Bestimme r und Zuweisung $S \subset A \times \{1, \ldots, r\}$ kompatibel



Lösung (Greedy-Interval-Partitioning-Algorithmus)

input : Menge der Anfragen $A := \{a_1, \ldots, a_n\} := \{(s_1, f_1), \ldots, (s_n, f_n)\}$
output: Tiefe r von A
Menge $S \subseteq A \times \{1, \ldots, r\}$ der Anfragen mit zugewiesenem Label

$S \leftarrow \emptyset$;
$r \leftarrow (A = \emptyset)\,?\,0 : 1$;
while $A \neq \emptyset$ **do**
  $a \leftarrow \arg\min_{a_k \in A} s_k$; /* wähle a mit frühestem Startzeitpunkt */
  $A \leftarrow A \setminus \{a\}$; /* entferne a aus A */
  $F \leftarrow \{1, \ldots, r\}$; /* Menge der für a erlaubten Label */
  foreach $(\tilde{a}, \tilde{\ell}) \in S$ **do**
    if $\tilde{a}$ *überlappt* a **then**
      $F \leftarrow F \setminus \{\tilde{\ell}\}$; /* Label von $\tilde{a}$ ist für a verboten */
    end
  end
  if $F = \emptyset$ **then**
    $r \leftarrow r + 1$; /* brauchen eine weitere Ressource */
    $S \leftarrow S \cup \{(a, r)\}$; /* weise Anfrage neuer Ressource zu */
  else
    $\ell \leftarrow$ waehle_aus(F); /* wähle eines der erlaubten Label */
    $S \leftarrow S \cup \{(a, \ell)\}$; /* füge (a, ℓ) zu S hinzu */
  end
end
return r,S;

Bemerkungen

- `waehle_aus()` kann eine beliebige Auswahlfunktion sein
- Proposition: S ist kompatibel
- Beweis: Folgt per Konstruktion, weil inkompatible Label ausgenommen werden. □
- Proposition: $r = \mathit{Tiefe}(A)$
- Beweis: Folgt per Konstruktion, weil die Anzahl der Label so lange erhöht wird, bis jede Überschneidung konfliktfrei auf unterschiedliche Ressourcen gelegt wurde. □

Stable Matching

Problem

- n Männer und n Frauen
- Alle sollen jeweils verlobt werden
- Jeder Mann hat eine streng absteigende Präferenzliste für jede Frau
- Jede Frau hat eine streng absteigende Präferenzliste für jeden Mann
- Wie bildet man die Paarungen optimal?

Formalisierung

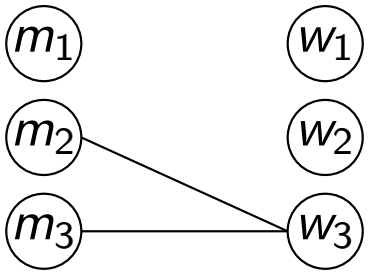
- Sei $M := \{m_1, \ldots, m_n\}$ die Menge der Männer
- Sei $W := \{w_1, \ldots, w_n\}$ die Menge der Frauen
- Sei für jeden Mann $m \in M$ der Vektor $L_W(m) \in W^n$ seine (absteigende) Präferenzliste (z.B. $(w_3, w_6, w_1, \ldots, w_9)$)
- Sei für jede Frau $w \in W$ der Vektor $L_M(w) \in M^n$ ihre (absteigende) Präferenzliste (z.B. $(m_1, m_5, m_8, \ldots, m_4)$)
- Sei $S \subset M \times W$ eine Tupelmenge, die die Verlobungen ausdrückt

$$(m, w) \in S \Leftrightarrow m \text{ und } w \text{ sind verlobt}$$

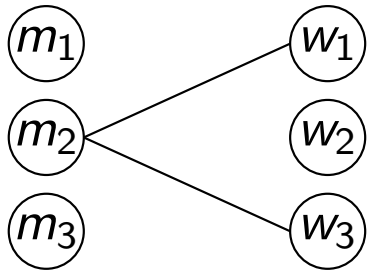
- S „Matching“ $:\Leftrightarrow \forall_{(m,w)\neq(\tilde{m},\tilde{w})\in S} : (m \neq \tilde{m} \wedge w \neq \tilde{w})$
  (i.e. kein Mann und keine Frau kommt in mehr als einem Matching vor)
- S „perfektes Matching“ $:\Leftrightarrow S$ Matching $\wedge\ |S| = n$
  (i.e. jeder Mann und jede Frau ist genau einmal verlobt)

Beispiele (Paarmengen)

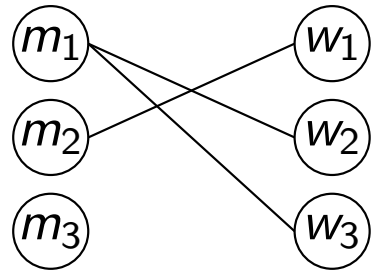
- Keine Matchings S



$\{(m_2, w_3), (m_3, w_3)\}$

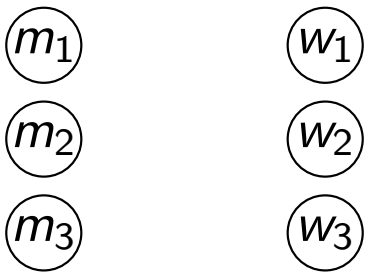


$\{(m_2, w_1), (m_2, w_3)\}$



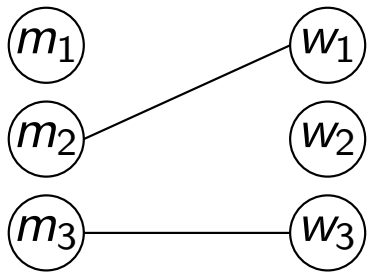
$\{(m_1, w_2), (m_1, w_3), (m_2, w_1)\}$

- Matchings S



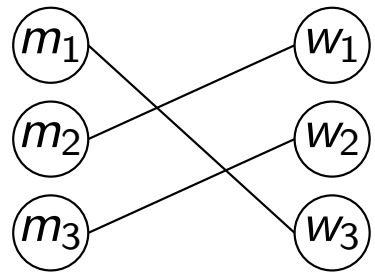
$\emptyset$

nicht perfekt



$\{(m_2, w_1), (m_3, w_3)\}$

nicht perfekt

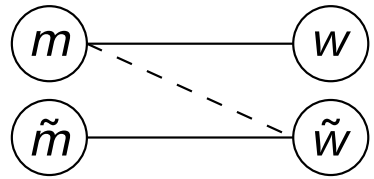


$\{(m_1, w_3), (m_2, w_1), (m_3, w_2)\}$

perfekt

Stabilität

- Beispiel
  - $M = \{m, \tilde{m}\}$, $W = \{w, \tilde{w}\}$
  - m bevorzugt $\tilde{w}$ gegenüber w
  - $\tilde{w}$ bevorzugt m gegenüber $\tilde{m}$
  - Sei $S = \{(m, w), (\tilde{m}, \tilde{w})\}$



- Das Paar $(m, \tilde{w})$ bezeichnen wir als „Instabilität“ bezüglich S
- Ein Matching ohne Instabilitäten bezeichnen wir als „stabiles Matching“
- Ziel des Stable Matching-Problems: Finde ein *perfektes* und *stabiles* Matching S

Lösung (Gale-Shapley-Algorithmus)

input : Menge der Männer $M := \{m_1, \ldots, m_n\}$
Menge der Frauen $W := \{w_1, \ldots, w_n\}$
Präferenzlisten $L_W(m_1), \ldots, L_W(m_n)$ der Männer
Präferenzlisten $L_M(w_1), \ldots, L_W(w_n)$ der Frauen
output: Paarmenge $S \subset M \times W$

$S \leftarrow \emptyset$;
while $M \neq \emptyset$ **do**
  $m \leftarrow$ waehle_aus(M);
  $w \leftarrow$ naechste_aus($L_W(m)$); /* nächste Frau aus Liste von m */
  if $\exists_{\tilde{m} \in M} : (\tilde{m}, w) \in S$ **then**
    if *w bevorzugt m gegenüber $\tilde{m}$* **then**
      $S \leftarrow S \setminus \{(\tilde{m}, w)\}$; /* w kündigt Verlobung mit $\tilde{m}$ auf */
      $M \leftarrow M \cup \{\tilde{m}\}$; /* $\tilde{m}$ wird wieder frei */
      $S \leftarrow S \cup \{(m, w)\}$; /* m verlobt sich mit w */
      $M \leftarrow M \setminus \{m\}$; /* m ist nicht mehr frei */
    end
  else
    $S \leftarrow S \cup \{(m, w)\}$; /* m verlobt sich mit w */
    $M \leftarrow M \setminus \{m\}$; /* m ist nicht mehr frei */
  end
end
return S;

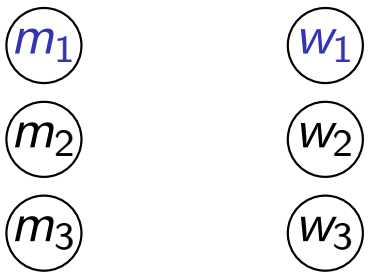
Bemerkungen

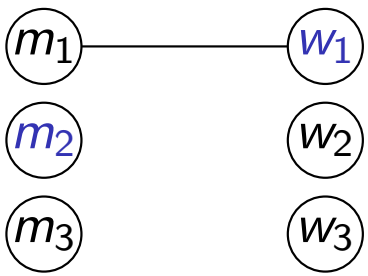
- `waehle_aus()` kann eine beliebige Auswahlfunktion sein
- Der Gale-Shapley-Algorithmus
  - terminiert
  - liefert das Mann-Optimale stabile Matching
  - liefert das Frau-Optimale stabile Matching, wenn die Rollen von Männern und Frauen getauscht werden
- Das stabile Matching ist für das Problem eindeutig, wenn das Mann-Optimale Matching und Frau-Optimale Matching übereinstimmen

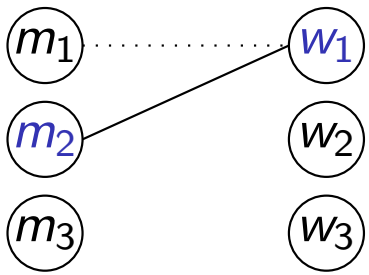
Beispiel

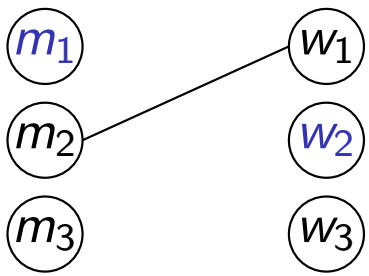
Seien $M = \{m_1, m_2, m_3\}$ und $W = \{w_1, w_2, w_3\}$

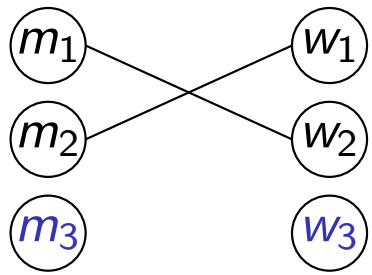
- $L_W(m_1) = (w_1, w_2, w_3)$
- $L_W(m_2) = (w_1, w_3, w_2)$
- $L_W(m_3) = (w_1, w_2, w_3)$
- $L_M(w_1) = (m_2, m_1, m_3)$
- $L_M(w_2) = (m_1, m_3, m_2)$
- $L_M(w_3) = (m_1, m_2, m_3)$

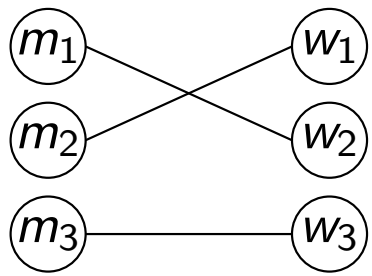
0.


1.


2.


3.


4.


5.


$$S = \{(m_1, w_2), (m_2, w_1), (m_3, w_3)\}$$

Abbildung: Fragment des Mechanismus von Antikythera (70-60 v. Chr.) - Erster (Analog-)Computer (Quelle: Marsyas, CC BY 2.5)

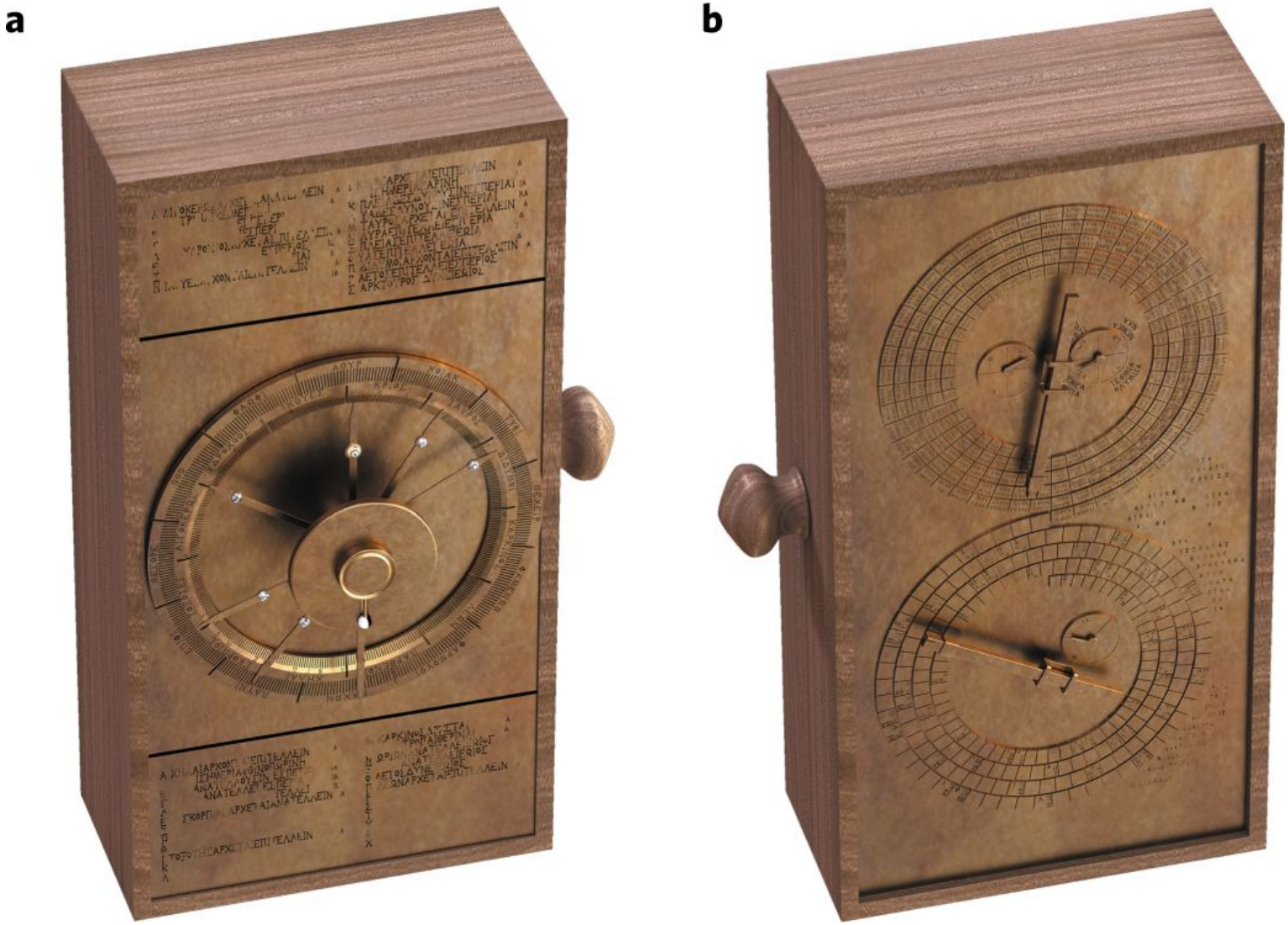


Abbildung: Rekonstruktion des Mechanismus von Antikythera - Astronomische Uhr (Quelle: Nature Astronomy, Volume 2, S.35–42 (2018))