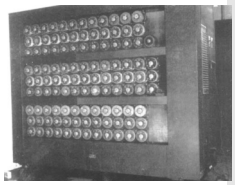
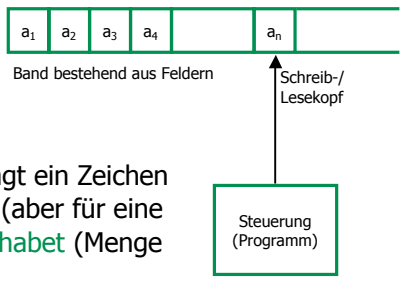
Abstrakte Maschinenmodelle: Turingmaschine (TM)

- 1936 von Alan Turing zum theoretischen Studium der Berechenbarkeit eingeführt
- Besteht aus
  - einem festen Teil ("Hardware")
  - einem variablen Teil ("Software")
- TM meint nicht eine Maschine, die genau eine Sache tut, sondern ein allgemeines Konzept, welches eine ganze Menge von verschiedenen Maschinen definiert
- Alle TM sind aber nach einem Schema aufgebaut und funktionieren nach den selben Regeln



"Hardware"

- Einseitiges, unendlich großes Band aus einzelnen Feldern, Schreib-/Lesekopf und Steuerung.



- Jedes Feld des Bandes trägt ein Zeichen aus einem frei wählbaren (aber für eine Maschine festen) Bandalphabet (Menge von Zeichen).
- Der Schreib-/Lesekopf ist auf ein Feld positioniert, welches dann gelesen oder geschrieben werden kann.

"Software"

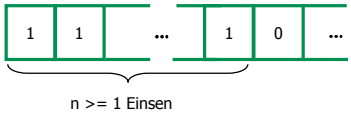
- folgende Operationen auf der Hardware möglich:
  - Überschreibe Feld unter dem Schreib-/Lesekopf mit einem Zeichen und gehe ein Feld nach rechts.
  - Überschreibe Feld unter dem Schreib-/Lesekopf mit einem Zeichen und gehe ein Feld nach links.
- besteht aus endlich vielen Zuständen und einer Tabelle, die beschreibt wie man von einem Zustand in einen anderen gelangen kann.
- Tabelle ist variabler Teil der Maschine, die "Software"

Beispiel

Zustand	Eingabe	Operation	Folgezustand
1	0	0 links	2
2	1	1 rechts	1

- Aktueller Zustand q, Maschine liest x.
- Paar (q,x) bestimmt Zeile in der Tabelle, in der man Operation b und Folgezustand q' findet.
- Operation b wird ausgeführt und die Maschine in den Zustand q' versetzt.
- Die Tabelle heißt auch Programm

- Start/Stop
  - Ein ausgezeichneter Endzustand
  - Maschine beginnt im Anfangszustand (oBdA Nr. 1)
  - Landet die Maschine im Endzustand, so wird gestoppt.



Zustand	Eingabe	Operation	Folgezustand	Bemerkung
1	1 0	0, rechts 0, rechts	1 2	Anfangszustand
2				Endzustand

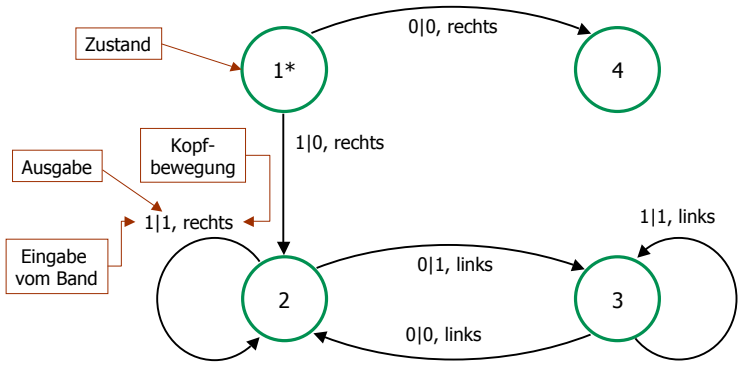
- Löschen einer Einserkette; Bandalphabet {0,1}

Beispiel 2: Programm mit vier Zuständen

Zustand	Eingabe	Operation	Folgezustand	Bemerkung
1	1 0	0, rechts 0, rechts	2 4	Start
2	1 0	1, rechts 1, links .	2 3	
3	1 0	1, links . 0, rechts	3 2	
4				Ende

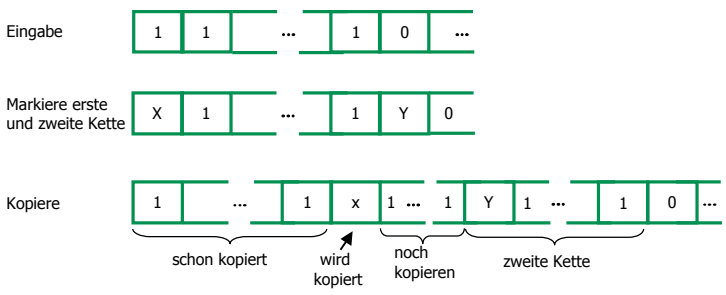
- Was macht das Programm? (Anwenden auf 1 0 0 ...)

- Übergangsgraph zu Beispiel 2

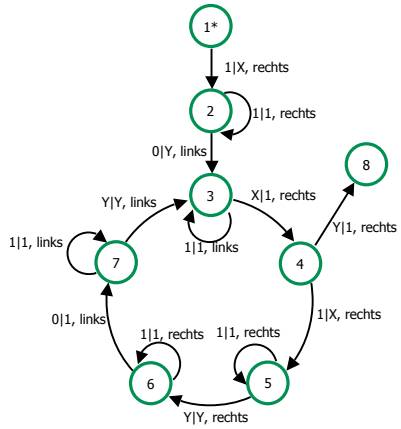


Bsp. 3: Verdoppeln einer Einserkette

- Eingabe: n Einsen, danach Nullen
- Ausgabe: 2n Einsen, danach Nullen
- Idee:

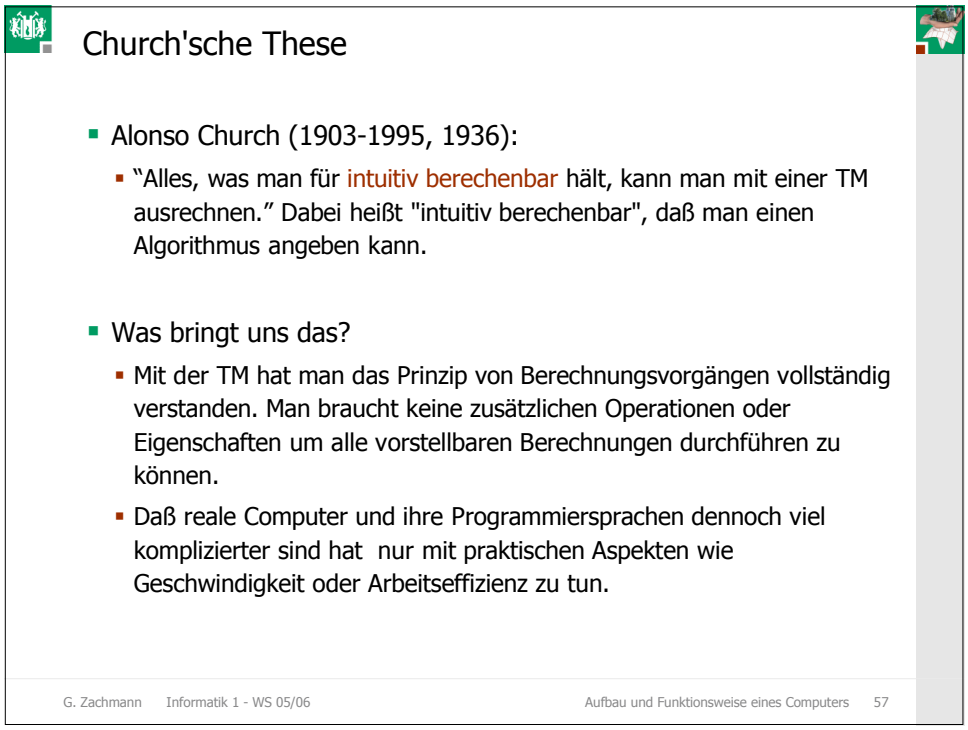


- Übergangsgraph



Mächtgkeit von abstrakten Maschinenmodellen

- Was kann man mit einer Turingmaschine alles berechnen?
- Auf einem PC (mit unendlich viel Speicher) kann man eine Turingmaschine simulieren
  - Alles was man mit einer Turingmaschine berechnen kann, kann man auch mit einem PC berechnen
- Zu einem PC mit gegebenem Programm kann man eine TM angeben, die die Berechnung des PCs nachvollzieht!
- PC und TM können dieselbe Klasse von Problemen lösen!
  - Beweis in Informatik III, Theorie der Berechenbarkeit
    - Was ist eine berechenbare Funktion bzw. Ein algorithmisch lösbares Problem?
    - Gibt es nicht-berechenbare Funktionen?
    - Gibt es Problemstellungen, die wohldefiniert, aber nicht algorithmisch lösbar sind?


Church'sche These
Alonso Church (1903-1995, 1936):
"Alles, was man für intuitiv berechenbar hält, kann man mit einer TM ausrechnen." Dabei heißt "intuitiv berechenbar", daß man einen Algorithmus angeben kann.
Was bringt uns das?
Mit der TM hat man das Prinzip von Berechnungsvorgängen vollständig verstanden. Man braucht keine zusätzlichen Operationen oder Eigenschaften um alle vorstellbaren Berechnungen durchführen zu können.
Daß reale Computer und ihre Programmiersprachen dennoch viel komplizierter sind hat nur mit praktischen Aspekten wie Geschwindigkeit oder Arbeitseffizienz zu tun.
G. Zachmann Informatik 1 - WS 05/06 Aufbau und Funktionsweise eines Computers 57