TECHNISCHE UNIVERSITÄT DARMSTADT

Mathematik

Complexity Theory

Martin Ziegler

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik

Seminar: SAT Solving in Kryptoanalyse

Veranstaltungsform:	S2
Hochschullehrer:	Dr. Stanislav Bulygin / Prof. Johannes Buchmann
Ort / Zeit:	Do: 14:15-16:00 in S2/02-A313
Voraussetzungen:	Einführung in die Kryptographie
Anmeldung/Registration:	über TUCaN (link)

Lehrinhalte

In the seminar we will address the topic of SAT-solving applied to algebraic cryptanalysis. We will start with the background on the satisfiability problem from the propositional logic. Then we will consider main approaches used to solve this problem, including the overview of the most known algorithms such as DPLL. Conjunctive Normal Form (CNF), a de fact standard for input to SAT solvers, will be studied together with some examples how different combinatorial problem may be modeled as an instance of the satisfiability problem in CNF. Then we switch to algebraic cryptanalysis of symmetric primitives: block and stream ciphers. After the general overview of the area and methods used there, we will focus on how the problem of breaking a cipher may be modeled as a satisfiability problem. In particular, how one may translate an algebraic representation of a cipher to a problem in CNF. Approaches, heuristics, and tricks of the area will be addressed. As specific examples we will consider stream ciphers Grain and Trivium, block ciphers KeeLoq, KTAN-Family, and PRINTCipher.

Reminder: Asymptotics

- Landau: For $f,g : \mathbb{N} \to \mathbb{R}$ write
  - $f=O(g) \quad \Leftrightarrow \quad \exists M\ \forall n \geq M: f(n) \leq M \cdot g(n)$
  - $f=\Omega(g) \quad \Leftrightarrow \quad \exists M\ \forall n \geq M: f(n) \geq g(n)/M$
  - $f=\Theta(g) \quad \Leftrightarrow \quad f=O(g) \wedge f=\Omega(g)$
- These notions neglect lower order terms and allow to simplify many expressions
- e.g. $5 \cdot n^3 - 27 \cdot n^2 + 933 \cdot n + 2197 = \Theta(n^3)$
- further examples in the exercises
- f is <u>polynomialy bounded</u> $\Leftrightarrow \exists k: f=O(n^k)$

Asymptotic Running Times

n	$\log_2 n \cdot 10$s	$n \cdot \log n$ sec	n^2 msec	n^3 µsec	2^n nsec
10	33sec	33sec	0.1sec	1msec	1msec
100	≈1min	11min	10sec	1sec	40 Mrd. Y
1000	≈1.5min	≈3h	17min	17min	
10 000	≈2min	1.5 days	≈1 day	11 days	
100 000	≈2.5min	19 days	4 months	32 years	

- Running times of some sorting algorithms
  - BubbleSort: $O(n^2)$ comparisons and copy instr.s
  - QuickSort: typically $O(n \cdot \log n)$ steps
    but $O(n^2)$ in the *worst-case*
  - HeapSort: always at most $O(n \cdot \log n)$ operations
- Here: always worst-case considerations!
  - w.r.t. input size (e.g. bit length) $=: n \to \infty$

Example Matrix Addition Multiplication

TECHNISCHE UNIVERSITÄT DARMSTADT

Complexity Theory

Fix a ring $(R,+,-,0,\times,1)$

- Input: entries of matrices $A,B \in R^{n\times n}$
- Wanted: entries of $n\times n$-matrix $C := A+B$
- school: n^2 inner products á $O(n)$: $O(n^2)$, optimal

Z

Z_3: $\ldots_{2,2})$

Multiplication of $n\times n$-matrices using

$$\begin{array}{|c|c|}\hline C_{1,1} & C_{1,2} \\ \hline C_{2,1} & C_{2,2} \\ \hline \end{array} = \begin{array}{|c|c|}\hline A_{1,1} & A_{1,2} \\ \hline A_{2,1} & A_{2,2} \\ \hline \end{array} \cdot \begin{array}{|c|c|}\hline B_{1,1} & B_{1,2} \\ \hline B_{2,1} & B_{2,2} \\ \hline \end{array}$$

7 multiplications +18 additions of $(n/2)\times(n/2)$-matrizes

$$L(n) = 7\cdot L(\lceil n/2 \rceil)$$

asymptotics dominated by #multiplications

$$L(n) = O(n^{\log_2 7}), \quad \log_2 7 \approx 2{,}81$$

World record: $O(n^{2.38})$ [Coppersmith&Winograd'90]

More on Jan.11, 2012 in our colloqium…

Models of Computation

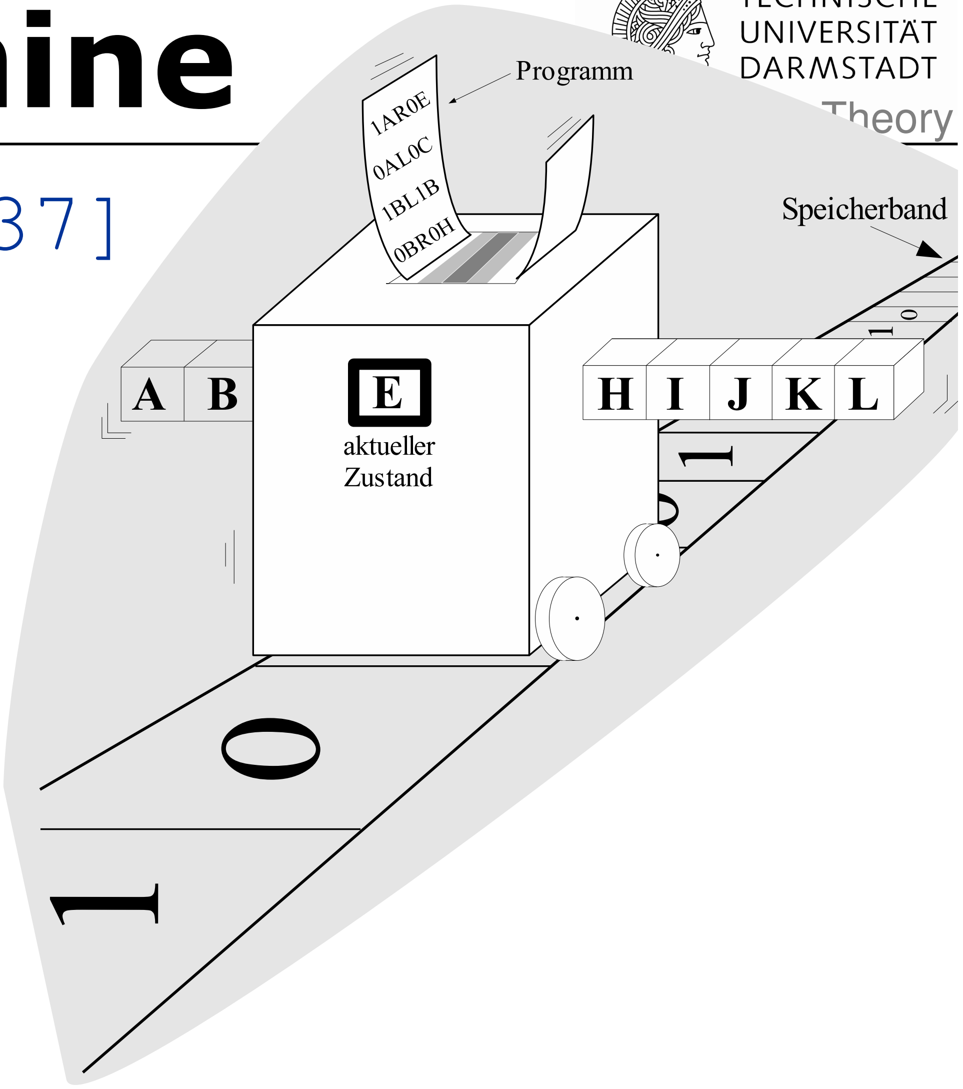
- Matrix Multipl. count arithmetic operations
  - $2n^2$ inputs, n^2 outputs: $\Omega(n^2)$.
- HeapSort: $O(n \cdot \log n)$ operations
  - Can improve (asymptotically) ?
  - Yes, 1 operation suffices: **`sort`**$(x_1,\ldots,x_n)$
- Complexity subj.to *model of computation*:
- mathem. formalization (&idealization)
  - which operationen are available
  - and at what 'cost' in terms of resources
- resources such as (run) time, memory, #prozessors (parallel computing)

Here mostly *Turing Maschines*

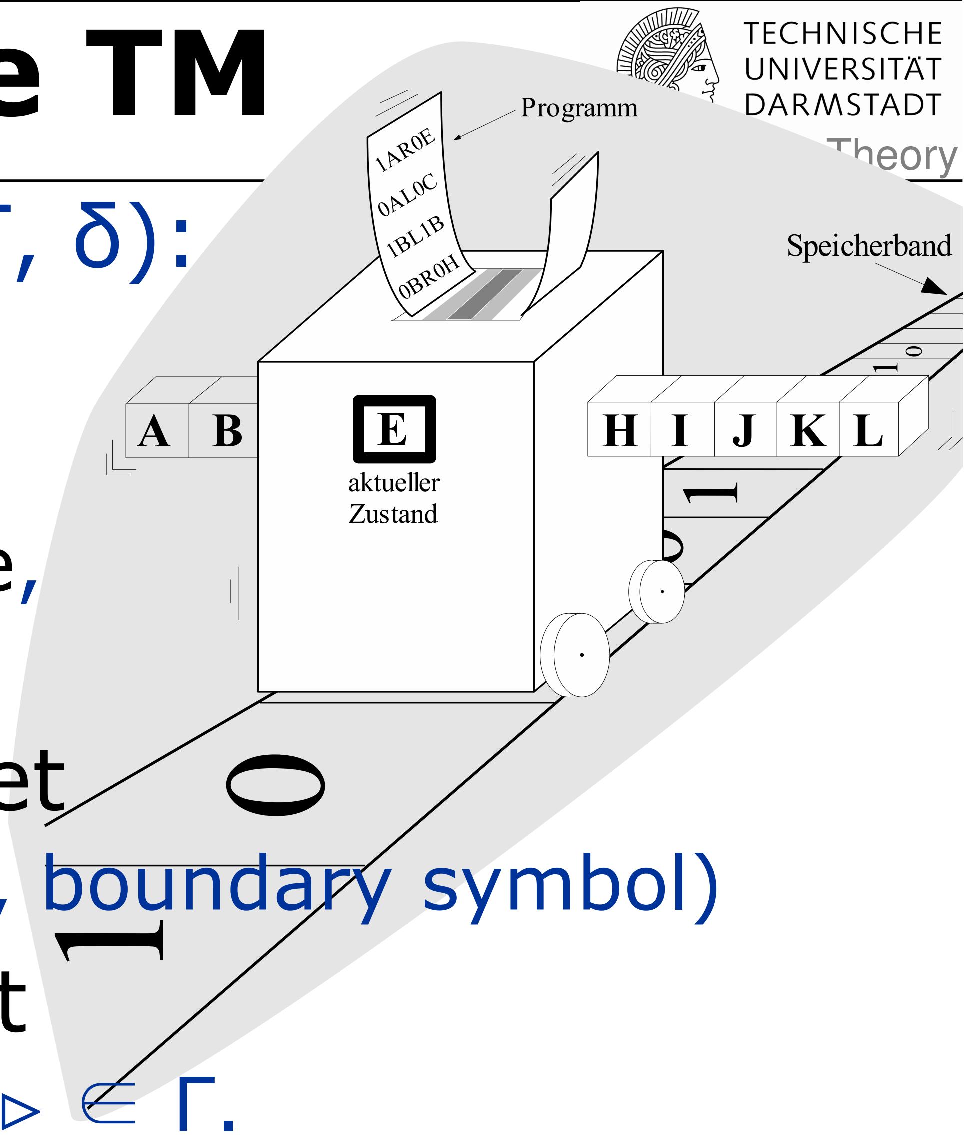
Turing Maschine

Alan M. Turing [1937]

- mathematical idealization and abstraction of his assistents (so called *„computers“*)
- nowadays generally accepted als model for digital computing machines (PCs)



determ. 1-tape TM



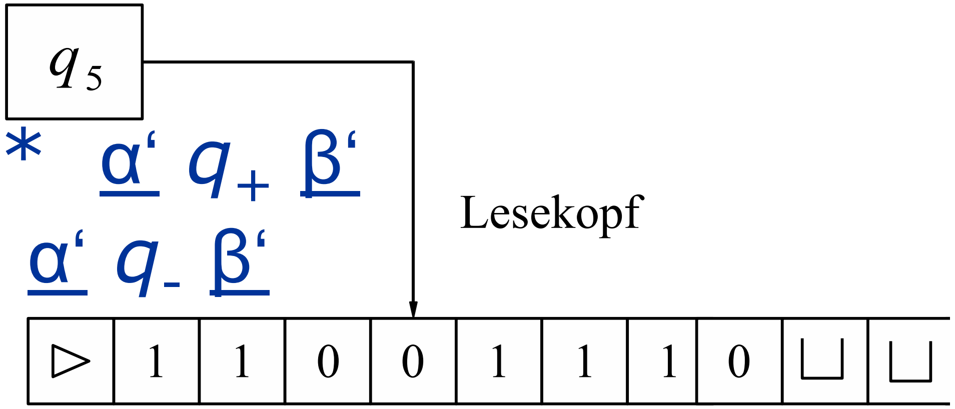
- 4-tupel $\mathcal{M} = (Q, \Sigma, \Gamma, \delta)$:
- Q finite set of states
  - with s=initial state
  - q_+=accept. final state,
  - q_-=reject. final state
- Σ finite input alphabet
  - with , $\rhd \notin \Sigma$ (blank, boundary symbol)
- Γ finite tape alphabet
  - where $\Sigma \subset \Gamma$ and , $\rhd \in \Gamma$.
- $\delta : Q \backslash \{q_+ , q_-\} \times \Gamma \rightarrow Q \times \Gamma \times \{\mathtt{R}, \mathtt{L}, \mathtt{N}\}$ denotes the transition function

Configuration, Successor, Computation

- $\mathcal{M} = (Q,\Sigma,\Gamma,\delta)$; $\Gamma^* = \{$ finite sequen.s over Γ $\}$
- Configuration: $\underline{\alpha}\ q\ \underline{\beta}$, here „110 q_5 01110“
  - (beyond $\underline{\beta}$ only   s; $\underline{\beta}$ does not end on   )
- one step according to δ: direct

  successor configuration $\underline{\alpha}\ q\ \underline{\beta}\ \vdash\ \underline{\alpha'}\ p\ \underline{\beta'}$
- *n*-th successor configuration $K \vdash^n K''$
- (indirect) successor configuration $K \vdash^* K''$
- $\mathcal{M}$ accepts $\underline{w}$ if there

  are $\underline{\alpha'},\underline{\beta'}\in\Gamma^*$ with $s\ \underline{w}\ \vdash^*\ \underline{\alpha'}\ q_+\ \underline{\beta'}$
- $\mathcal{M}$ rejects $\underline{w}$ if $s\ \underline{w}\ \vdash^*\ \underline{\alpha'}\ q_-\ \underline{\beta'}$



Acceptance und Decision

- The language accepted by $\mathcal{M}$ is $L(\mathcal{M}) := \{\ \underline{w} : M \text{ accepts } \underline{w}\ \}$
  - For $\underline{w} \notin L(M)$, $\mathcal{M}$ may enter a closed loop!
- $\mathcal{M}$ decides $L(\mathcal{M})$ if it in addition rejects every $\underline{w} \in \Sigma^* \backslash L$.
- $L \subseteq \Sigma^*$ called semi-decidable if accepted by some TM;
- L called decidable if decided by some TM.

M accepts $\underline{w}$ if there exists $\underline{\alpha'},\underline{\beta'} \in \Gamma^*$ with $s\,\underline{w} \vdash^* \underline{\alpha'}\, q{+}\, \underline{\beta'}$

M rejects $\underline{w}$ if there exists $\underline{\alpha'},\underline{\beta'} \in \Gamma^*$ with $s\,\underline{w} \vdash^* \underline{\alpha'}\, q{-}\, \underline{\beta'}$

Some logical considerations

TM $\mathcal{M}=(Q,\Sigma,\Gamma,\delta)$ with Q,Σ,Γ finite sets
and $\delta : Q\setminus\{q_+,q_-\} \times \Gamma \to Q \times \Gamma \times \{\mathtt{R},\mathtt{L},\mathtt{N}\}$

- Renaming states does not really affect the TM
- There are only countably many TMs
- but continuously many $L\subseteq\Sigma^*$ (Cantor)
- Each TM semi-decides precisely one $L\subseteq\Sigma^*$.

$\Rightarrow$ *Almost every* $L\subseteq\Sigma^*$ is not semi-decidable!

- Gödel: truth of arithmetic sentences not (semi-) decidable; Davis, Robinson, Matiyasevich: Unsolvability of diophantic equations not semi-decidable

$L \subseteq \Sigma^*$ called semi-decidable if there exists a TM accepting precisely those $\underline{w}$ in L.

Resource: Time

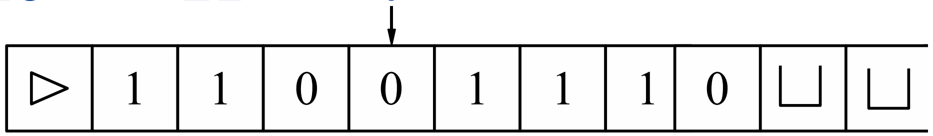
Let $\mathcal{M}=(Q,\Sigma,\Gamma,\delta)$ denote a DTM.

- One step is a direct transition between configurations $\underline{\alpha}\ q\ \underline{\beta} \vdash \underline{\alpha'}\ p\ \underline{\beta'}$
- For $\underline{w}\in\Sigma^*$ let $T_{\mathcal{M}}(\underline{w})$ denote the number of steps $\mathcal{M}$ executes on input $\underline{w}$ before terminating; $T_{\mathcal{M}}(\underline{w}):=\infty$ if $\mathcal{M}$ does not terminate on $\underline{w}$.
- For $n\in\mathbb{N}$ let $T_{\mathcal{M}}(n):=\max\{T_{\mathcal{M}}(\underline{w}) \mid \underline{w}\in\Sigma^{\leq n}\}$ denote the (worst-case) running time of $\mathcal{M}$ on inputs of length $\leq n$; $T_{\mathcal{M}}:\mathbb{N}\to\mathbb{N}$ is the running time function of $\mathcal{M}$
- $T_{\mathcal{M}}(n)\leq O(t(n))$: $\mathcal{M}$ called $O(t(n))$-time bounded
- $\mathrm{DTIME}(t(n)) := \{\ L(\mathcal{M})$: DTM $\mathcal{M}$ is $O(t(n))$—time bounded$\}$

Example TM for Palindromes

$$\mathrm{PALIN} := \{\ \underline{w} \in \{0,1\}^*\!:\ \underline{w} = \underline{w}^{®}\ \}$$

- $Q := \{\ s\ ,\ q_{r0}\ ,\ q_{r1}\ ,\ q_{z0}\ ,\ q_{z1}\ ,\ q_{\ell}\ ,\ q_{+}\ ,\ q_{-}\ \}$
- δ informally:



  - s: first symbol is ? Then q_+
  - Otherwise 'remember' first symbol i in state q_{ri} overwrite with , and skip one cell to the right
  - q_{ri}: scan tape to the right for then skip back by one cell and enter state q_{zi}
  - q_{zi}: present symbol different from i? $\rightarrow$state q_- (over)write , state q_ℓ and one cell to the left
  - q_ℓ: scan left for then right, restart with state s