Rucksackproblem und Verifizierbarkeit

Gegeben: n Gegenstände mit Gewichten $G=\{g_1,g_2,\ldots,g_n\}$ und Werten $W=\{w_1,w_2,\ldots,w_n\}$ sowie zulässiges Gesamtgewicht g.

Gesucht: Teilmenge $S\subseteq\{1,\ldots,n\}$ mit $\sum_{i\in S} w_i$ ist maximal unter der Bedingung $\sum_{i\in S} g_i \leq g$.

Rucksackproblem und Verifizierbarkeit

Gegeben: n Gegenstände mit Gewichten $G=\{g_1,g_2,\ldots,g_n\}$ und Werten $W=\{w_1,w_2,\ldots,w_n\}$ sowie zulässiges Gesamtgewicht g.

Gesucht: Teilmenge $S\subseteq\{1,\ldots,n\}$ mit $\sum_{i\in S} w_i$ ist maximal unter der Bedingung $\sum_{i\in S} g_i \leq g$.

$RS_{ent} := \{\langle G,W,g,w\rangle \mid$ es existiert ein $S\subseteq\{1,\ldots,n\}$ mit $\Sigma_{i\in S}\, g_i \leq g$ und $\Sigma_{i\in S}\, w_i \geq w\}$

Rucksackproblem und Verifizierbarkeit

- **Liegt RS_{ent} in P?**
- **Algorithmus mit Laufzeit $\Theta(n \cdot w)$ aus DuA ist kein polynomieller Algorithmus!**
- **Kennen weder Polynomialzeit DTM für RS_{ent} noch können wir beweisen, dass eine solche nicht existieren kann.**
- **Lösungen für RS_{ent} sind effizient überprüfbar.**
- **RS_{ent} teilt diese Eigenschaften mit vielen anderen Problemen.**

Verifizierer

Definition 3.7 Sei L eine Sprache. DTM V heißt **Verifizierer** für L falls

$$L = \{ w \mid \text{es gibt ein } c \text{, so dass } V \langle w,c \rangle \text{ akzeptiert} \}$$

c: **Zertifikat**, **Zeuge**

V heißt **polynomieller Verifizierer**, falls ein $k \in \mathbb{N}$ existiert mit

$$L = \{ w \mid \text{es gibt ein } c \text{ mit } |c| \leq |w|^k, \text{ so dass } V \langle w,c \rangle \text{ akzeptiert} \}$$

und die Laufzeit von V bei Eingabe $\langle w,c \rangle$ polynomiell in $|w|$ ist. L heißt dann **polynomiell verifizierbar**.

Verifizierer für RS_{ent}

V_1 bei Eingabe $\langle G,W,g,w,S \rangle$:

1. Teste, ob $\langle S \rangle$ die Kodierung einer Teilmenge von $\{1,\dots,n\}$ ist. Falls nicht, lehne ab.
2. Teste, ob $\sum_{i \in S} g_i \leq g$ und $\sum_{i \in S} w_i \geq w$. Falls ja, akzeptiere, sonst lehne ab.

Das Problem des Handlungsreisenden

TSP_{ent} := { $\langle \Delta, L \rangle$ | es gibt eine Rundreise durch alle n Städte der Länge $\leq L$ }

Verifizierer für TSP_{ent}

V_1 bei Eingabe $\langle \Delta, L, \pi \rangle$:

1. Teste, ob $\langle \pi \rangle$ die Kodierung einer Permutation der Zahlen von 1 bis n. Falls nicht, lehne ab.

2. Teste, ob $\sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)} \leq L$.

   Falls ja, akzeptiere, sonst lehne ab.

Element-Non-Distinctness

$L := \{ \#w_1\#\dots\ \#w_n \mid w_i \in \{0,1\}^*,\ w_i = w_j$ für ein $i \neq j \}$

Beispiele:

- $\#011\#11\#01\#10\#01 \in L$
- $\#001\#111\#10\#11 \notin L$

Klasse NP

Definition 3.8 NP ist die Klasse der Sprachen, die polynomiell verifizierbar sind.

RS_{ent}, $TSP_{ent} \in NP$.

Satz 3.9 $P \subseteq NP$.

Offenes Problem Ist $P = NP$ oder gilt $P \subset NP$?

Nichtdeterministische Turingmaschinen

- **Liefern alternative Beschreibung von NP.**
- **Erlaubt uns, die schwierigsten Probleme in NP zu identifizieren**
  $\rightarrow$ NP-Vollständigkeit.
- **Geben der Turingmaschine die Möglichkeit, einen von mehreren möglichen Rechenschritten auszuwählen**
  $\rightarrow$ Nichtdeterminismus.
- **Realisierung: $\delta(q, a)$ ist Menge von Tripeln der Form $(q´,b,D)$.**
- **NTMs nicht realistisch, aber sehr nützlich für Verständnis der Komplexität von Problemen.**

Nichtdeterministische Turingmaschinen

Definition 3.10 **Eine nichtdeterministische 1-Band Turing-maschine (NTM) ist ein 4-Tupel (Q,Σ,Γ,δ), wobei Q,Σ,Γ endliche Mengen sind. Weiter gilt**

1. **Q ist die Zustandsmenge mit $q_0,q_{accept},q_{reject} \in Q$, $q_{accept} \neq q_{reject}$.**

2. **Σ ist das Eingabealphabet, $\sqcup,\rhd \notin \Sigma$.**

3. **Γ ist das Bandalphabet, $\Sigma \subset \Gamma$, $\sqcup,\rhd \in \Gamma$.**

4. **$\delta: Q \setminus \{q_{accept},q_{reject}\} \times \Gamma \to \wp(Q \times \Gamma \times \{L,R\})$ ist die Übergangsfunktion.**

Nichtdeterministische Turingmaschinen

- $\wp(S)$:= Menge aller Teilmengen von S, Potenzmenge.
- Für alle $q \in Q$, $a \in \Gamma$, $a \neq \triangleright$: $(p,b,D) \in \delta(q, a) \Rightarrow b \neq \triangleright$.
- Für alle $q \in Q$: $(p,a,R) \in \delta(q,\triangleright) \Rightarrow a = \triangleright$.
- Rechenschritt = einmalige Anwendung der Übergangsfunktion.

Beispiel einer NTM N_1

- $Q = \{q_0,q_1,q_2,q_3\}$, $q_{accept} = q_2$, $q_{reject} = q_3$.
- $\Sigma = \{0,1\}$, $G=\{0,1,\sqcup,\triangleright\}$
- δ ist definiert durch die folgende Tabelle:

δ	0	1	$\triangleright$	$\sqcup$
q_0	$\{(q_0,0,R),(q_3,0,R)\}$	$\{(q_1,1,R)\}$	$\{(q_0,\triangleright,R)\}$	$\{(q_3,\sqcup,R)\}$
q_1	$\{(q_0,0,R),(q_3,0,R)\}$	$\{(q_2,1,L)\}$	$\{(q_3,\triangleright,R)\}$	$\{(q_3,\sqcup,R)\}$

NTM - Berechnung

NTM N = (Q,Σ,Γ,δ). Berechnung bei Eingabe $w \in \Sigma^*$:

- **Startet im Zustand q_0, mit Bandinhalt $\triangleright w$ und Lesekopf auf $\triangleright$.**
- **wendet in jedem Rechenschritt Übergangsfunktion δ an,**
- **bis Zustand q_{accept} oder q_{reject} erreicht wird,**
- **falls einer dieser Zustände je erreicht wird,**
- **sonst Endlosrechnung.**

Turingmaschine - Konfigurationen

NTM N = (Q,Σ,Γ,δ), $\alpha,\beta \in \Gamma^*$, $q \in Q$.

N ist in Konfiguration K = $\alpha q \beta$, wenn gilt:

- **auf dem Band der DTM N steht $\alpha\beta$, gefolgt von Blanks,**
- **N befindet sich im Zustand q,**
- **der Lesekopf von N steht auf dem ersten Symbol von β.**

NTM - Rechenschritt

- **NTM N in Konfiguration K = $\alpha q a \beta$.**
- **$\delta(q, a) = \{ (q_1,b_1,D_1),\ldots,(q_l,b_l,D_l) \}$.**
- **N kann jeden durch ein Triple $(q_i,b_i,D_i) \in \delta(q, a)$ beschriebenen Rechenschritt ausführen.**

Berechnungen und Konfigurationen

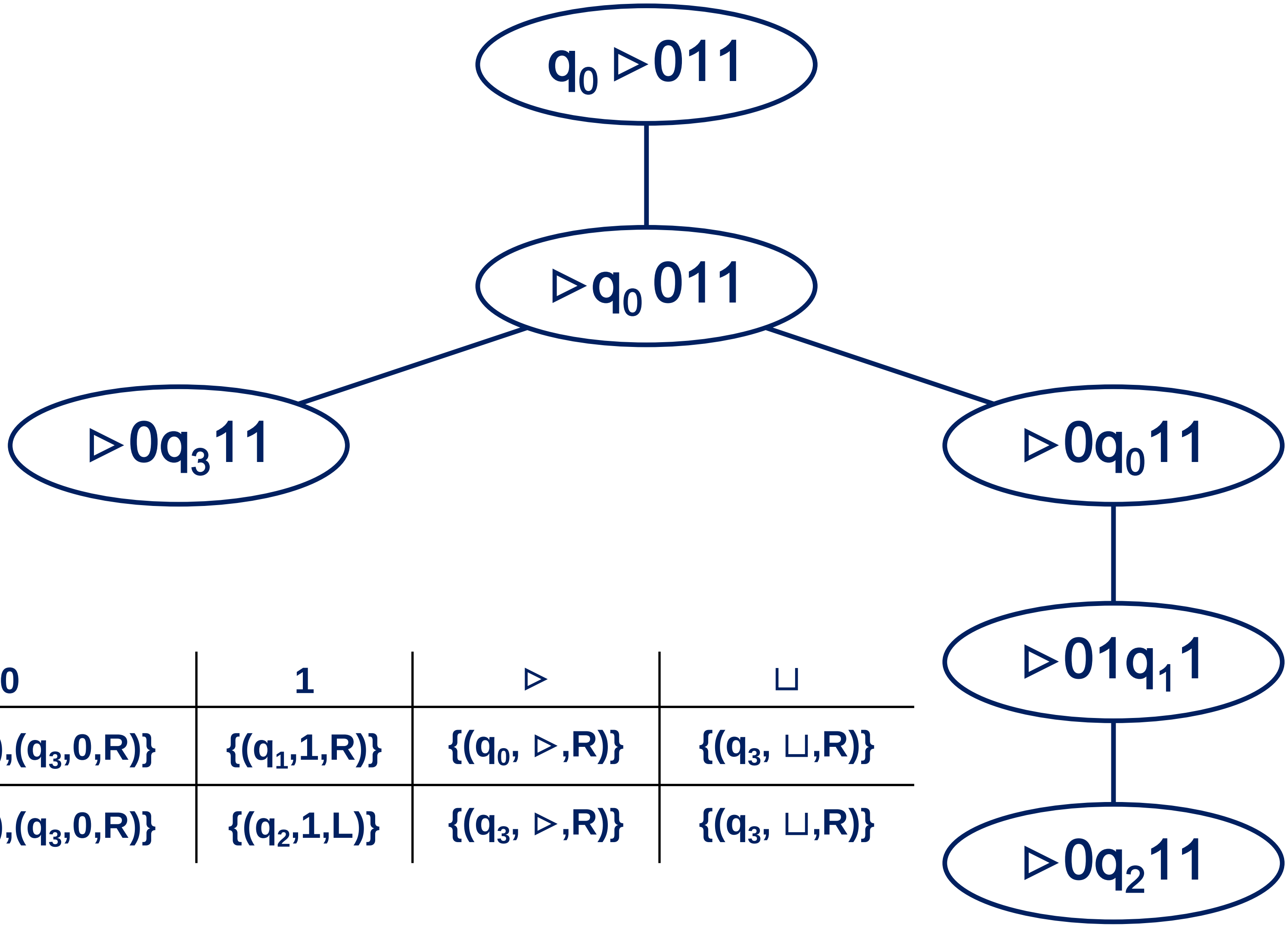
- **Eingabe für N ist w, dann heißt $s \triangleright w$ Startkonfiguration.**
- **$K=\alpha q \beta$ heißt akzeptierende Konfiguration, falls $q = q_{accept}$.**
- **$K=\alpha q \beta$ heißt ablehnende Konfiguration, falls $q = q_{reject}$.**
- **Berechnung von N bei Eingabe w führt zu Folge $K_1, K_2, \ldots$ von Konfigurationen.**
- **Es gibt mehrere Berechnungen von N bei Eingabe w, abhängig von den ausgewählten Rechenschritten.**
- **Darstellung möglicher Berechnungen im Berechnungsbaum.**

Beispiel einer NTM N_1

- $Q = \{q_0,q_1,q_2,q_3\}$, $q_{accept} = q_2$, $q_{reject} = q_3$.
- $\Sigma = \{0,1\}$, $\Gamma = \{0,1,\sqcup,\triangleright\}$
- δ ist definiert durch die folgende Tabelle:

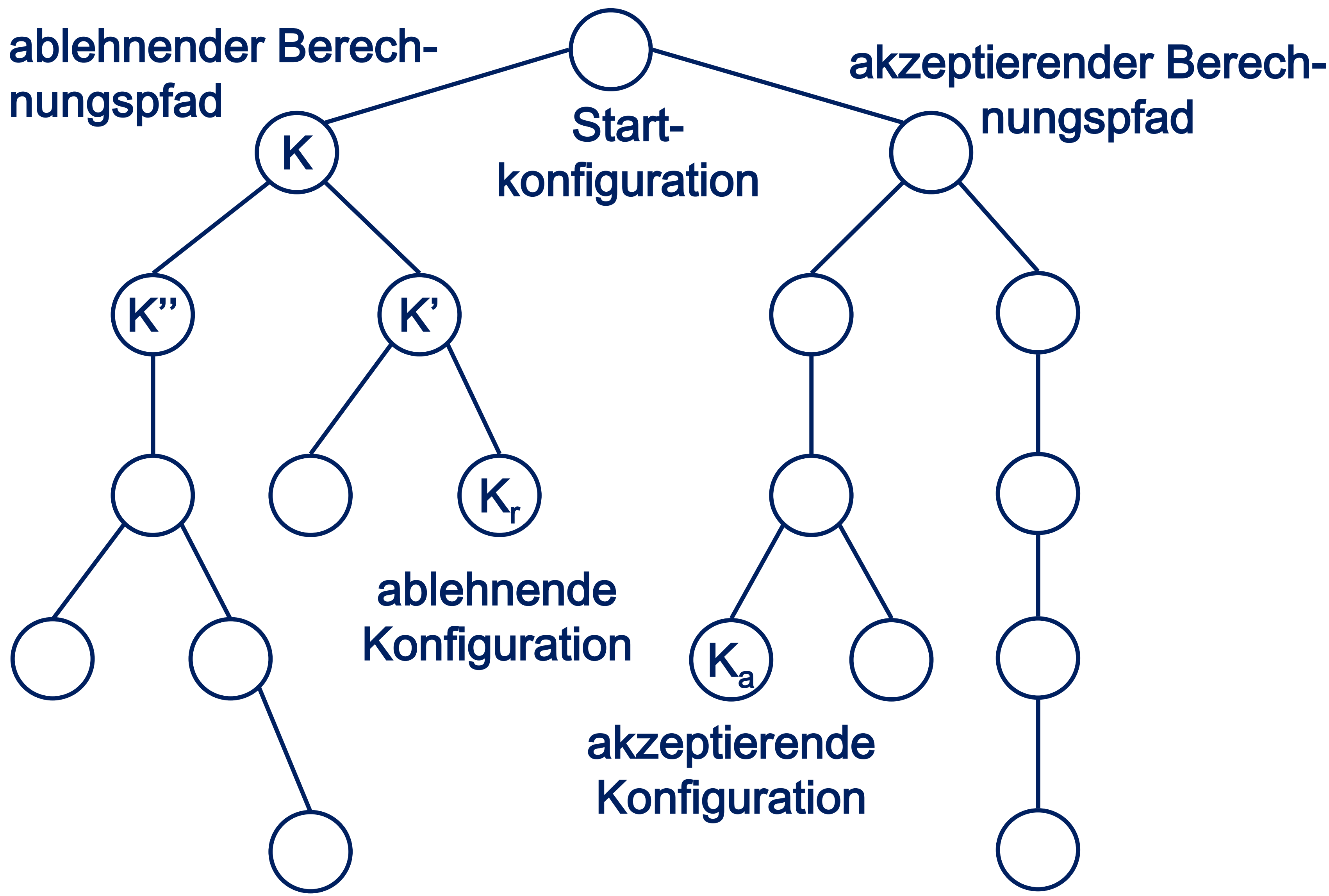
δ	0	1	$\triangleright$	$\sqcup$
q_0	$\{ (q_0,0,R),(q_3,0,R) \}$	$\{(q_1,1,R)\}$	$\{(q_0, \triangleright,R)\}$	$\{(q_3,\sqcup,R)\}$
q_1	$\{ (q_0,0,R),(q_3,0,R) \}$	$\{(q_2,1,L)\}$	$\{(q_3, \triangleright,R)\}$	$\{(q_3,\sqcup,R)\}$

Berechnungsbaum der NTM N_1 bei w = 011



δ	0	1	$\triangleright$	$\sqcup$
q_0	$\{(q_0,0,R),(q_3,0,R)\}$	$\{(q_1,1,R)\}$	$\{(q_0, \triangleright,R)\}$	$\{(q_3, \sqcup,R)\}$
q_1	$\{(q_0,0,R),(q_3,0,R)\}$	$\{(q_2,1,L)\}$	$\{(q_3, \triangleright,R)\}$	$\{(q_3, \sqcup,R)\}$

Berechnungsbaum einer NTM



Akzeptieren und Entscheiden

Definition 3.11 Sei $N = (Q,\Sigma,\Gamma,\delta)$ eine NTM. Die NTM **akzeptiert** $w \in \Sigma^*$, wenn es **mindestens eine** akzeptierende Berechnung von N bei Eingabe w gibt.

Akzeptieren und Entscheiden

NTM N **hält** bei Eingabe $w \in \Sigma^*$, wenn alle Berechnungspfade von N bei Eingabe w endlich sind.

Definition 3.12 Die von einer NTM $N = (Q,\Sigma,\Gamma,\delta)$ akzeptierte Sprache L(N) ist definiert als

$$L(N) := \{ w \in \Sigma^* \mid N \text{ akzeptiert } w \}.$$

- NTM N **akzeptiert** die Sprache L, falls L = L(N).
- N **entscheidet** die von ihr akzeptierte Sprache L(N), wenn N immer hält.

Beispiel einer NTM N_1

- $Q = \{q_0,q_1,q_2,q_3\}$, $q_{accept} = q_2$, $q_{reject} = q_3$
- $\Sigma = \{0,1\}$, $\Gamma = \{0,1,\sqcup,\triangleright\}$
- δ ist definiert durch die folgende Tabelle:

δ	0	1	$\triangleright$	$\sqcup$
q_0	$\{ (q_0,0,R),(q_3,0,R) \}$	$\{(q_1,1,R)\}$	$\{(q_0, \triangleright,R)\}$	$\{(q_3,\sqcup,R)\}$
q_1	$\{ (q_0,0,R),(q_3,0,R) \}$	$\{(q_2,1,L)\}$	$\{(q_3, \triangleright,R)\}$	$\{(q_3,\sqcup,R)\}$

$L(N_1) = \{ w \in \{0,1\}^* \mid w$ enthält die Teilfolge 11 $\}$

NTM für RS_{ent}

$RS_{ent} := \{\langle G,W,g,w\rangle \mid$ es existiert ein $S \subseteq \{1,\dots,n\}$ mit $\Sigma_{i\in S}\, g_i \leq g$ und $\Sigma_{i\in S}\, w_i \geq w\}$

N bei Eingabe $\langle G,W,g,w\rangle$:

1. Erzeuge nichtdeterministisch ein Wort $c \in \{0,1\}^n$.

   c kodiert eine Teilmenge $S \subseteq \{1,\dots,n\}$.

2. Teste, ob $\Sigma_{i\in S}\, g_i \leq g$ und $\Sigma_{i\in S}\, w_i \geq w$. Falls ja, akzeptiere, sonst lehne ab.

Laufzeit einer NTM

Definition 3.13 NTM N (Q,Σ,Γ,δ) halte immer.

- Für $w \in \Sigma^*$ ist $T_N(w)$ die maximale Anzahl von Rechenschritten in einer Berechnung von N bei Eingabe w.
- Für $n \in \mathbb{N}$ ist $T_N(n) := \max \{T_N(w) \mid w \in \Sigma^{\leq n}\}$.
- $T_N : \mathbb{N} \to \mathbb{N}$ heißt **Zeitkomplexität** oder **Laufzeit** der NTM N.
- N hat Laufzeit $O(f(n))$, wenn $T_N(n) = O(f(n))$.

Nichtdeterministische Zeitkomplexität

Definition 3.14 Sei $t: \mathbb{N} \to \mathbb{N}$ eine monoton wachsende Funktion. Die Klasse **NTIME(t(n))** ist dann definiert als

$$\mathrm{NTIME}(t(n)) := \left\{ \begin{array}{l} L \mid L \text{ ist eine Sprache, die von einer NTM} \\ \qquad \text{mit Laufzeit } O(t(n)) \text{ entschieden wird} \end{array} \right\}$$

Satz 3.15 **NP** ist die Klasse der Sprachen, die von einer nicht-deterministischen Turingmaschine mit polynomieller Laufzeit entschieden werden, d.h. $\mathrm{NP} = \bigcup_{k \in \mathbb{N}} \mathrm{NTIME}(n^k)$.

Vom Verifizierer zur NTM

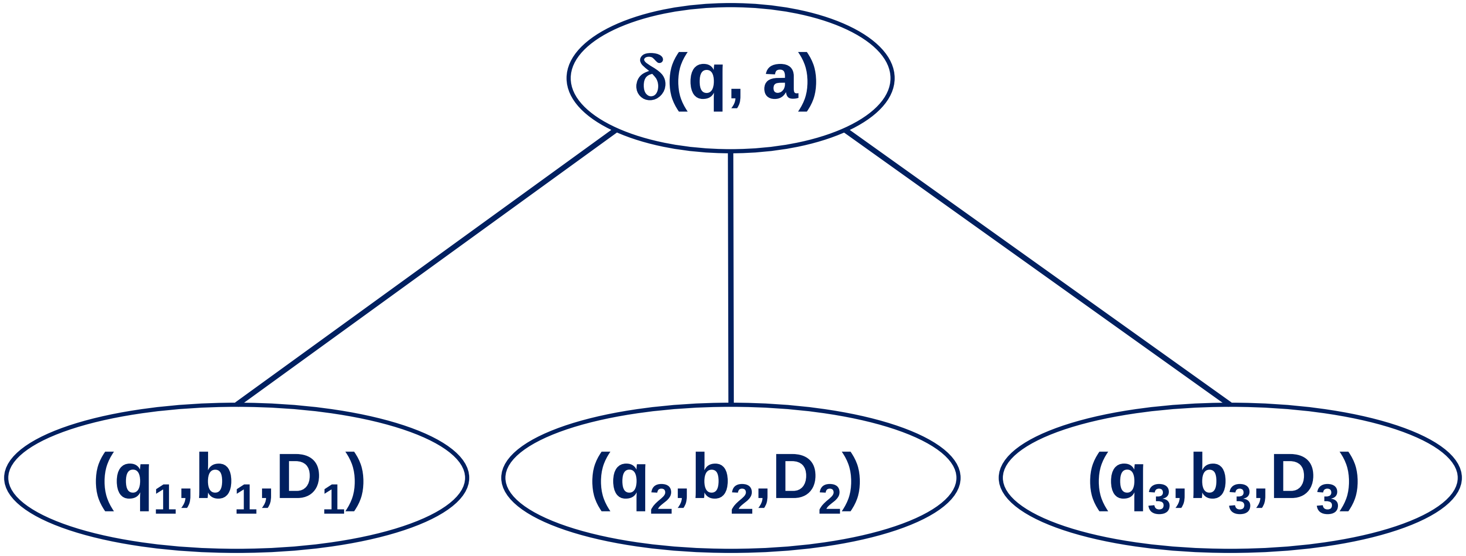
Sei $L \in NP$. Dann gibt es ein $k \in \mathbb{N}$ und einen polynomiellen Verifizierer V mit

$L = \{ w \mid \text{es gibt ein } c \text{ mit } |c| \leq |w|^k, \text{ so dass } V \langle w,c \rangle \text{ akz.} \}$

N für L bei Eingabe w der Länge n:

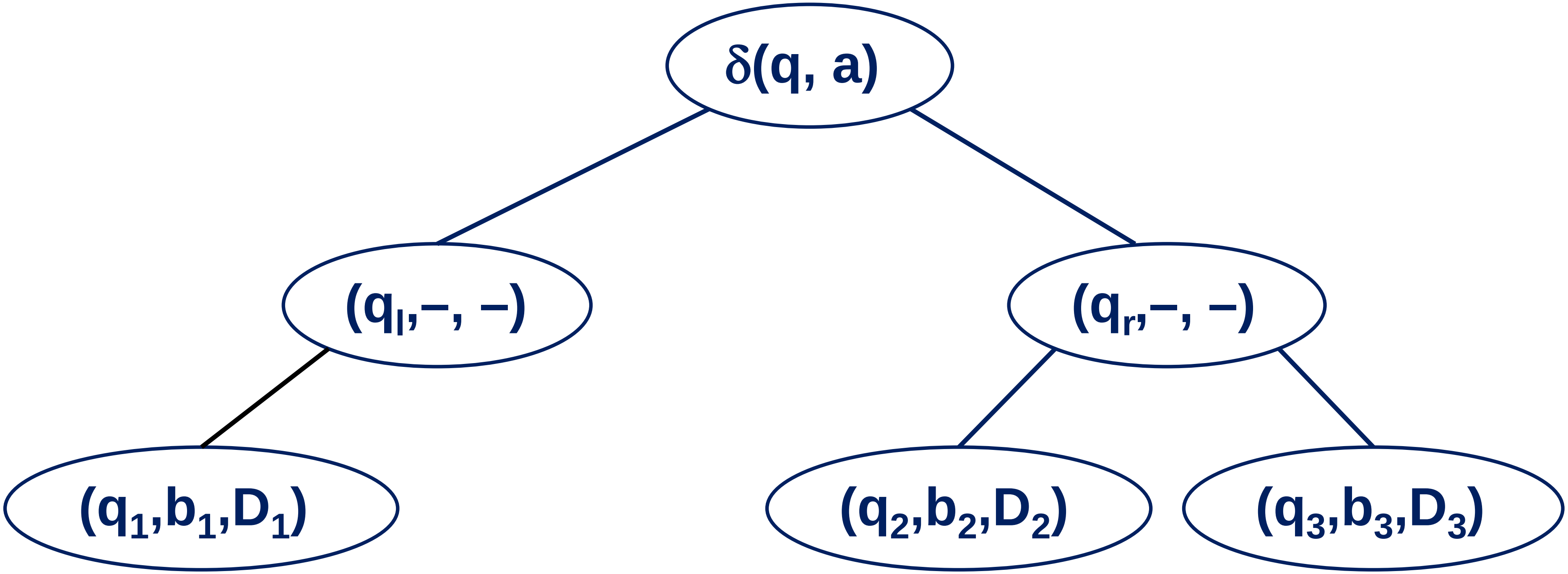
1. **Erzeuge nichtdeterministisch ein Wort c der Länge höchstens n^k (für das k oben).**
2. **Simuliere V mit Eingabe $\langle w,c \rangle$.**
3. **Wenn V die Eingabe $\langle w,c \rangle$ akzeptiert, akzeptiere. Sonst lehne ab.**

Einschränkung auf $|\delta(q, a)| \leq 2$



- Ersetzen einen Schritt durch $O(\log(|\delta(q, a)|))=O(1)$ viele Schritte.
- Asymptotische Laufzeit bleibt unverändert.

Einschränkung auf $|\delta(q, a)| \leq 2$



- Ersetzen einen Schritt durch $O(\log(|\delta(q, a)|))=O(1)$ viele Schritte.
- Asymptotische Laufzeit bleibt unverändert.

Von NTM zu Verifizierer

Sei L $\in \bigcup_{k\in\mathbb{N}}$ NTIME(n^k). Dann existiert NTM N, die L in poly Zeit entscheidet.

V bei Eingabe $\langle w,c \rangle$:

1. Interpretiere $c \in \{0,1\}^*$ als Kodierung eines möglichen Berechnungspfads von N bei Eingabe w.
2. Simuliere die Berechnung von N, die diesem Pfad entspricht.
3. Ist diese Berechnung von N akzeptierend, akzeptiere. Sonst lehne ab.

Boolesche Variablen, Operatoren, Formeln

- **Boolesche Variablen x können Werte wahr oder falsch annehmen.**
- **wahr $\triangleq$ 1, falsch $\triangleq$ 0.**
- **Boolesche Operatoren: und $\triangleq \wedge$; oder $\triangleq \vee$; nicht $\triangleq \neg$.**
- **Boolesche Formel: Ausdruck bestehend aus Booleschen Variablen und Operatoren, korrekt formatiert.**
- **Beispiel: $\phi = (\neg x \wedge y) \vee (x \wedge \neg z)$**

Belegungen und erfüllbare Formeln

- Boolesche Formel ϕ heißt **erfüllbar**, wenn es eine Belegung der Variablen in ϕ mit 1 und 0 gibt, so dass die Formel dann wahr (Wert = 1) ist.

- Beispiel: $\phi = (\neg x \wedge y) \vee (x \wedge \neg z)$ ist erfüllbar.
  Belegung $x = 0$, $y = 1$, $z = 0$.

- Beispiel: $\psi = x \wedge \neg x$ ist nicht erfüllbar.

Die Sprache SAT

SAT := { $\langle\phi\rangle$ | ϕ ist erfüllbare Boolesche Formel }

Beispiele:

- $\langle\phi\rangle \in$ **SAT,** $\phi = (\neg x \wedge y) \vee (x \wedge \neg z)$
- $\langle\psi\rangle \notin$ **SAT,** $\psi = x \wedge \neg x$

SAT liegt in NP

V_{SAT} bei Eingabe $\langle\phi,c\rangle$:

1. Teste, ob $\langle c\rangle$ die Kodierung einer Belegung der Variablen in ϕ ist. Falls nicht, lehne ab.
2. Falls $\langle c\rangle$ Belegung B kodiert und B die Formel ϕ erfüllt, akzeptiere. Sonst lehne ab.

SAT liegt in NP

N_{SAT} bei Eingabe $\langle\phi\rangle$:

1. Erzeuge nichtdeterministisch eine Belegung B der Variablen in ϕ.
2. Falls B die Formel ϕ erfüllt, akzeptiere. Sonst lehne ab.

Literale, Klauseln, KNF

- **Literale** sind Boolesche Variablen x oder Negationen Boolescher Variablen $\neg x$.
- **Klausel** ist Disjunktion von Literalen.
- Beispiel: $\neg x_1 \vee x_3 \vee \neg x_2$
- Formel ist in **konjunktiver Normalform** (KNF), wenn sie Konjunktion von Klauseln ist.
- $\phi = (\neg x_1 \vee x_3 \vee \neg x_2) \wedge (x_5 \vee \neg x_6) \wedge (x_7 \vee \neg x_1 \vee \neg x_3 \vee x_4)$
- In **3-KNF Formel** enthält jede Klausel 3 Literale.

Die Sprache 3SAT

3SAT := { $\langle\phi\rangle$ | ϕ ist erfüllbare 3-KNF Formel }

Beispiele:

- **$\langle\phi\rangle \in$ 3SAT,**
  $\phi = (\neg x_1 \vee x_3 \vee \neg x_2) \wedge (x_5 \vee \neg x_6 \vee x_4) \wedge (x_7 \vee \neg x_1 \vee \neg x_3)$

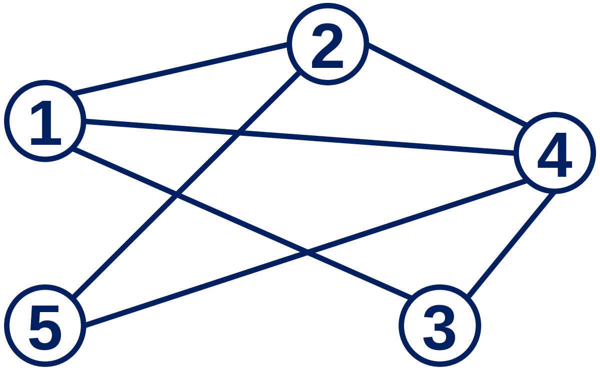
3SAT liegt in NP

N_{3SAT} bei Eingabe $\langle\phi\rangle$:

1. Falls ϕ keine 3KNF Formel ist, lehne ab.
2. Erzeuge nichtdeterministisch einen Bitvektor c.
3. Falls $\langle c\rangle$ Belegung B der Variablen in ϕ kodiert und B die Formel ϕ erfüllt, akzeptiere. Sonst lehne ab.

Graphen und Cliquen

- $G = (V, E)$ ungerichteter Graph.
- $C \subseteq V$ heißt **Clique**, wenn für alle $i,j \in C$ gilt $\{i,j\} \in E$.
- C heißt **k-Clique**, wenn C Clique und $|C| = k$.

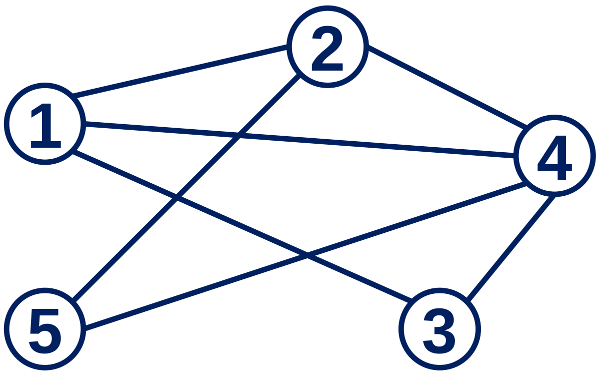


$C = \{1, 2, 4\}$ ist 3-Clique.

Die Sprache Clique

Clique := { $\langle G,k \rangle$ | G ist ein ungerichteter Graph mit einer k-Clique }

G_0 :=



$\langle G_0, 3 \rangle \in$ Clique

Clique liegt in NP

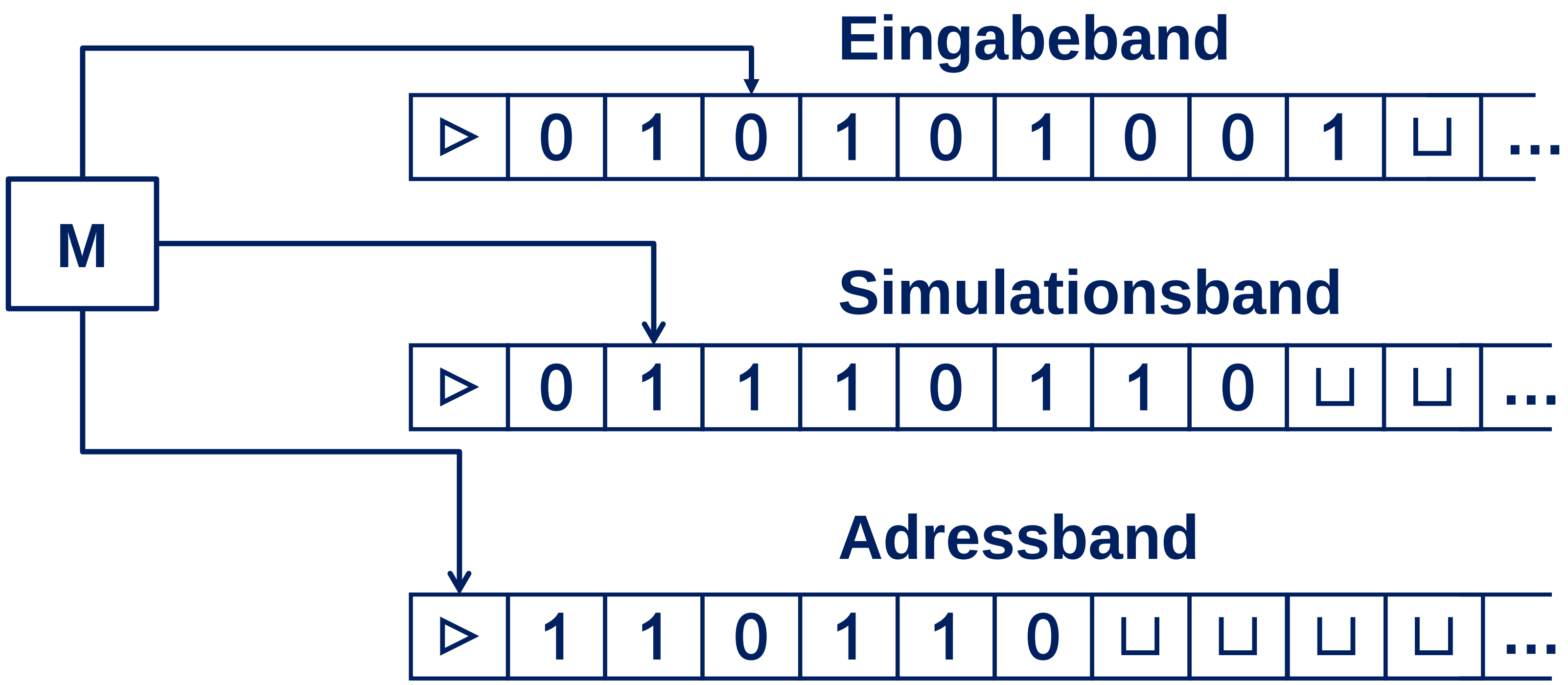
N_{Clique} bei Eingabe $\langle G,k \rangle$:

1. Erzeuge nichtdeterministisch eine Teilmenge C der Größe k der Knoten von G.
2. Bilden die Knoten in C eine Clique in G, akzeptiere. Sonst lehne ab.

Simulation einer NTM durch eine DTM

Satz 3.17 Sei $t : \mathbb{N} \rightarrow \mathbb{N}$ eine monoton wachsende Funktion mit $t(n) \geq n$ für alle $n \in \mathbb{N}$. Für jede NTM mit Laufzeit $t(n)$ gibt es eine DTM mit Laufzeit $2^{O(t(n))}$, die dieselbe Sprache entscheidet.

Simulation einer NTM durch eine DTM



Simulation einer NTM durch eine DTM

1. Schreibe $c = 0^{t(n)}$ auf das 3. Band, das Adressband.

2. Kopiere w vom 1. Band (Eingabeband) auf das 2. Band (Simulationsband).

3. Simuliere die durch c kodierte Berechnung von N bei Eingabe w auf dem 2. Band. Wird in der Simulation eine akzeptierende Konfiguration erreicht, akzeptiere. Sonst gehe zu 4.

4. Gibt es keine weiteren Worte in $\{0,1\}^{t(n)}$, lehne ab. Sonst ersetze c durch das lexikographisch nächste Wort in $\{0,1\}^{t(n)}$ und gehe zu 2.

Konsequenz für NP

$$\mathbf{NP} \subseteq \bigcup_{n \in \mathbb{N}} \mathbf{DTIME}(2^{n^k})$$

Offenes Problem **Ist P = NP oder gilt P $\subset$ NP?**