Interaktive Beweise

NP ist die Menge aller Sprachen, für die es polynomiell lange Beweise gibt.

Alternative Sicht: eine Prover TM kommuniziert mit einer polynomial zeitbeschränkten Verifier TM über einen gemeinsamen Kommunikations$. Beide haben Zugang zu dem Input$ und haben „private" Arbeits$s.

Die Berechnung erfolgt in höchstens polynomiell vielen Runden. Nur eine der beiden TMen ist jeweils in einer Runde aktiv. Eine Runde beginnt mit dem Lesen der Informationen auf dem Kommunikations$ (oder Input$) und endet nach eventuellen privaten Berechnungen auf den Arbeits$s mit dem Schreiben einer neuen (höchstens polynomiell langen) Information auf dem Kommunikations$.

Definition: Die Menge der durch solch ein interaktives Beweissystem darstellbaren Sprachen A ist wie folgt definiert: Es muss einen Verifier V geben, so dass $\forall x \in A$ es eine Prover Strategie gibt, so dass der Verifier bei Eingabe x und Kommunikation mit diesem Prover schließlich akzeptiert. Im Falle von $x \notin A$ verlangen wir, dass der Verifier bei Eingabe x und egal bei welcher Prover Strategie verwirft.

Bemerkung: Dieses anscheinend allgemeinere interaktive Kommunikationsmodell kann nach wie vor nichts anderes, als die Sprachen in **NP** berechnen.

Bemerkung: Die Sachlage ändert sich (möglicherweise) drastisch, wenn sich die TMen probabilistisch verhalten können.

Definition: Eine Sprache A liegt in der Klasse **IP** (Inderactive Proofs), falls es eine probabilistische, polynomial zeitbeschränkte TM V (den Verifier) gibt, so dass für alle x gilt:

$$\begin{aligned} x \in A \quad &\Rightarrow \quad \exists \text{Prover } P : W\big[(P,V)(x) = \text{JA}\big] > 2/3\,, \\ x \notin A \quad &\Rightarrow \quad \forall \text{Prover } P : W\big[(P,V)(x) = \text{JA}\big] < 1/3\,. \end{aligned}$$

Bemerkung: **NP** $\subseteq$ **IP** .

Bemerkung: **IP** $\subseteq$ **PSPACE** .

Wie gross ist die Klasse **IP** nun wirklich? Um wieviel wird die Klasse **NP** erweitert? Ein bekanntes Beispiel für eine Klasse in **IP**, für die nicht bekannt ist, ob sie in **NP** liegt, ist das Komplement des Graphenisomorphieproblems $\overline{GI}$: Gegeben zwei Graphen G_1 und G_2, beweise (interaktiv), dass diese nicht isomorph sind.

Prover	Komm$	Verifier
		Rät zufällig $i \in \{1,2\}$ und eine Permutation π auf $\{1,\ldots,n\}$ (n ist die Anzahl der Knoten in G_1 bzw G_2) und berechnet den Graphen $H = \pi(G_i)$;
	$\leftarrow H \leftarrow$	
Bestimmt $j \in \{1,2\}$, so dass G_j und H isomorph sind;		
	$\rightarrow j \rightarrow$	
		Akzeptiert, falls $i = j$.

Bemerkung: Wenn G_1 und G_2 nicht isomorph sind, dann kann ein geeigneter Prover immer mit dem richtigen j antworten, so dass der Verifier akzeptiert:

$$(G_1, G_2) \in \overline{GI} \Rightarrow \exists P : W[(P,V)(G_1, G_2) = \text{JA}] = 1 > 2/3$$

Im Falle von zwei isomorphen Graphen kann der Prover dagegen nur mit Wahrscheinlichkeit $1/2$ raten:

$$(G_1, G_2) \notin \overline{GI} \Rightarrow \forall P : W[(P,V)(G_1, G_2) = \text{JA}] \leq 1/2$$

Die **IP**-Definition ist noch nicht ganz erfüllt, da die zweite Wahrscheinlichkeit nicht kleiner als $1/3$ ist. Durch k faches Wiederholen (oder Ausrechnen von k „Zufallsgraphen" $H_1, \ldots, H_k$ in der ersten Runde durch den Verifier) kann man die zweite Wahrscheinlichkeit aber auf 2^{-k} verkleinern. Es gilt: $\overline{GI} \in$ **IP**(2) (2= Anzahl der Runden).

Zero Knowledge

Beispiel: Graphenisomorphieproblem GI: Es wäre das einfachste und würde nur polynomiell viele Bits kosten, einen Isomorphismus zwischen den beiden Graphen vom Prover zum Verifier zu übertragen. Damit wäre aber das ganze Geheimnis sozusagen verraten: der Verifier könnte dieses Geheimnis einem Dritten weitererzählen usw. In bestimmten Kontexten (zB Authentifizierung) könnte es unerwünscht sein, den Beweis zu verraten. Trotzdem möchte der Prover den Verifier davon überzeugen, dass er den Beweis kennt (zB den Isomorphismus zwischen zwei gegebenen isomorphen Graphen). Dieses anscheinend paradoxe Anliegen vermag ein Zero Knowledge Beweis tatsächlich zu erfüllen.

Prover	Komm$	Verifier
Rät zufällig $i \in \{1,2\}$ und eine Permutation π auf $\{1,\ldots,n\}$ (n ist die Anzahl der Knoten in G_1 bzw G_2) und berechnet den Graphen $H = \pi(G_i)$;		
	$\rightarrow H \rightarrow$	
		Wählt zufällig $j \in \{1,2\}$;
	$\leftarrow j \leftarrow$	
Bestimmt σ, so dass $\sigma(G_j) = H$;		
	$\rightarrow \sigma \rightarrow$	
		Akzeptiert, falls $\sigma(G_j) = H$.

Bemerkung: Dieses Protokoll ist (wenn man den üblichen Wiederholungs-Trick zur Reduktion der Fehlerwahrscheinlichkeit verwendet) ein **IP**-Protokoll, wobei der Verifier nichts über den Isomorphismus zwischen G_1 und G_2 erfährt: Es erfüllt nämlich die Zero Knowledge Definition: Die Information, die über den Kommunikation$ fließt, beinhaltet statistisch nichts Neues für den Verifier. Er wäre selbst in der Lage, mit den Berechnungsressourcen, die ihm zustehen, mit derselben Wahrscheinlichkeitsverteilung solche Tripel (H, j, σ) zu erzeugen, wie sie in einer typischen Realisierung des Protokolls auftreten würden. Daher kann er durch Beobachten der entstandenen Kommunikation nichts, aber auch gar nichts Neues erfahren.

Bemerkung: Es genügt, wenn der Prover selbst auch nur eine probabilistische, polynomiell zeitbeschränkte TM ist, falls ihm der Isomorphismus ϕ von G_1 nach G_2 bekannt ist ($\rightarrow$ praktische Relevanz).

Beweis: Man kann dem Protokoll Wort für Wort folgen, nur an der Stelle, wo es für den Prover heisst „Bestimme σ, so dass $\sigma(G_1) = H$" benutzt der Prover nun seine geheime Zusatzinformation ϕ mit $\phi(G_1) = G_2$ und berechnet σ in Polynomialzeit wie folgt:

$$\sigma = \begin{cases} \pi & \text{falls } i = 1, j = 1 \\ \pi\phi^{-1} & \text{falls } i = 1, j = 2 \\ \pi\phi & \text{falls } i = 2, j = 1 \\ \pi & \text{falls } i = 2, j = 2 \end{cases} \qquad \surd$$