Einführung wichtiger Komplexitätsklassen

Seminar

Ergebnisprüfung zur Laufzeit

Lehr- und Forschungsgebiet Theoretische Informatik

18.11.2003

Eric Bodden

Outline

- Kurze Geschichte der Komplexitätsforschung
- Wiederholung: P/NP
- Übertragung auf das NP Proof system
- Arthur/Merlin Games und Interactive Proofs (IP)
- Multi-Prover IPs
- Probabilistically Checkable Proofs (PCP)
- Fazit

P – Polynomial time

Klasse aller Sprachen, für die ein deterministischer Test auf Mitgliedschaft existiert, der in polynomieller Zeit ein Ergebnis (*true/false*) liefert.

$$\bigcup_{c \in N} TIME(n^c)$$

Ein *proof* ist ein String, der es einem *Verifier* ermöglicht, diese Mitgliedschaft zu testen.

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

NP – Nondeterministic, in polynomial time

- Die Klasse P kann angesehen werden als Klasse aller Sprachen für die in Polynomialzeit ein Beweis auf Mitgliedschaft *berechnet* werden kann.
- Für Sprachen, die in NP sind, steht der Nichtdeterminismus zur Verfügung. Obwohl also ein solcher Beweis evtl. nicht in Polynomialzeit *gefunden* werden kann, so kann er durch nichtdeterministisches Raten doch *überprüft* werden.
- Betrachte folgendes Beispiel...

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

NP – Nondeterministic, in polynomial time

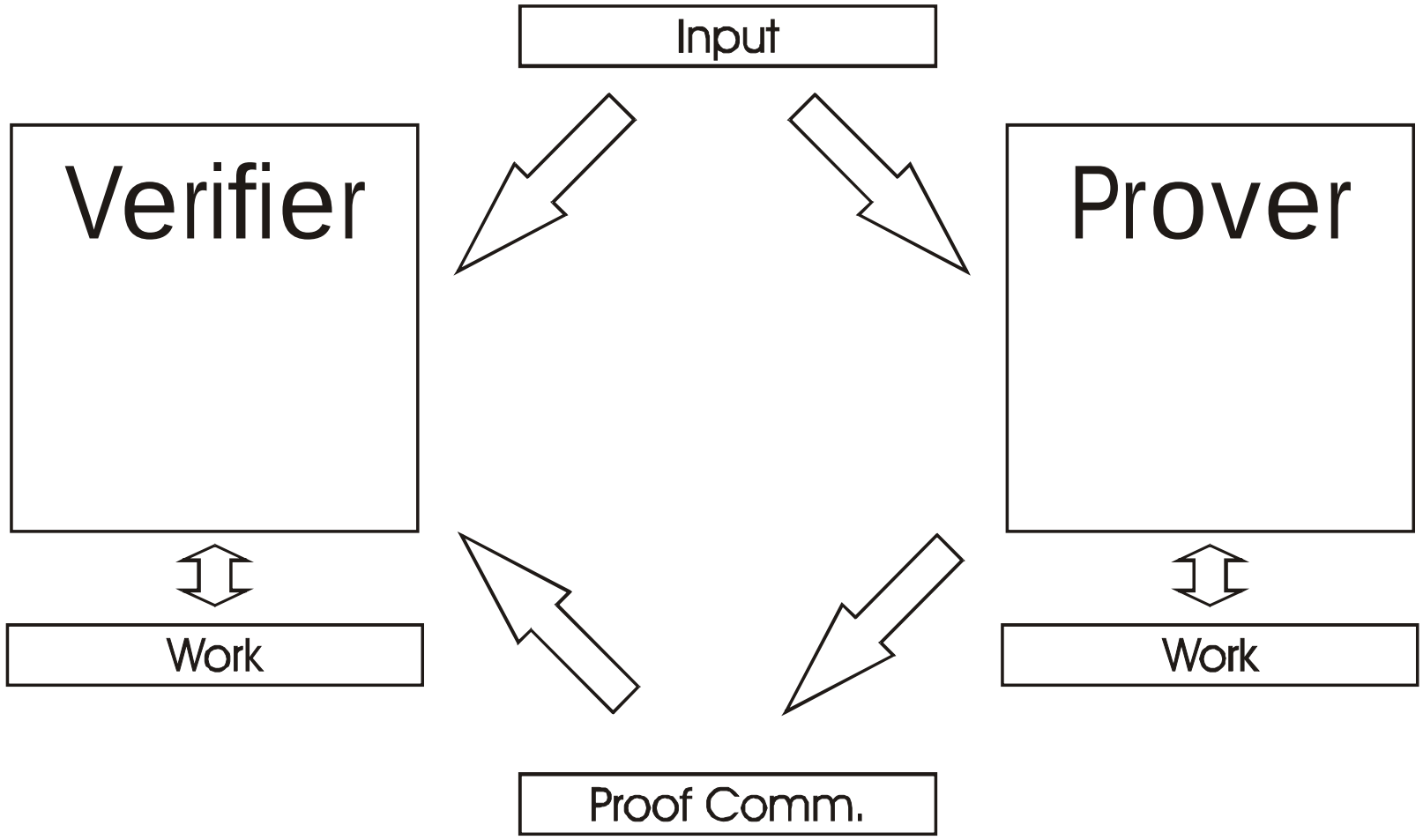
Klasse aller Sprachen, für die ein Test auf Mitgliedschaft in polynomieller Zeit *überprüft* werden kann.

Beispiel: *3SAT* (NP complete)

$$\bigwedge_{i=1}^{m} (w_{i_1} \vee w_{i_2} \vee w_{i_3})$$

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

NP – Proof system



Eigenschaften (NP – Proof system)

- Nur einseitige Kommunikation
  - § Prover sendet Beweis zum Verifier
  - § Verifier prüft in Polynomialzeit
- Zwei wichtige, natürliche Faktoren fehlen:
  1. Randomisierung ist nicht definiert, ist aber in der Realität anzutreffen
  2. Ebenso Interaktion: Nur Prover sendet zum Verifier
- Daher Definition von
  - § *Arthur-Merlin* Games (AM) [Babai, 1985]
  - § *Interactive Proof* systems (IP) [Goldwasser et al., 1985]
  - § Wie wir später sehen werden, gilt AM((t(n)) = IP(t(n)) für alle t(n).

Arthur/Merlin Spiele

Arthur/Merlin Spiele

In the left corner... Arthur:

- polynomiell beschränkt
- versucht, Merlin zu widerlegen

Waffen:

- Kommunikationskanal zu Merlin
- ein Random-Tape

Arthur/Merlin Spiele

In the right corner... Merlin:

- übermenschlich und somit nicht zeitbeschränkt
- versucht, Arthur von der Richtigkeit seines Proofs zu überzeugen

Waffen:

- Der magische Blick: Er sieht Arthurs zufällige Auswahlen
- Kommunikationskanal zu Arthur

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

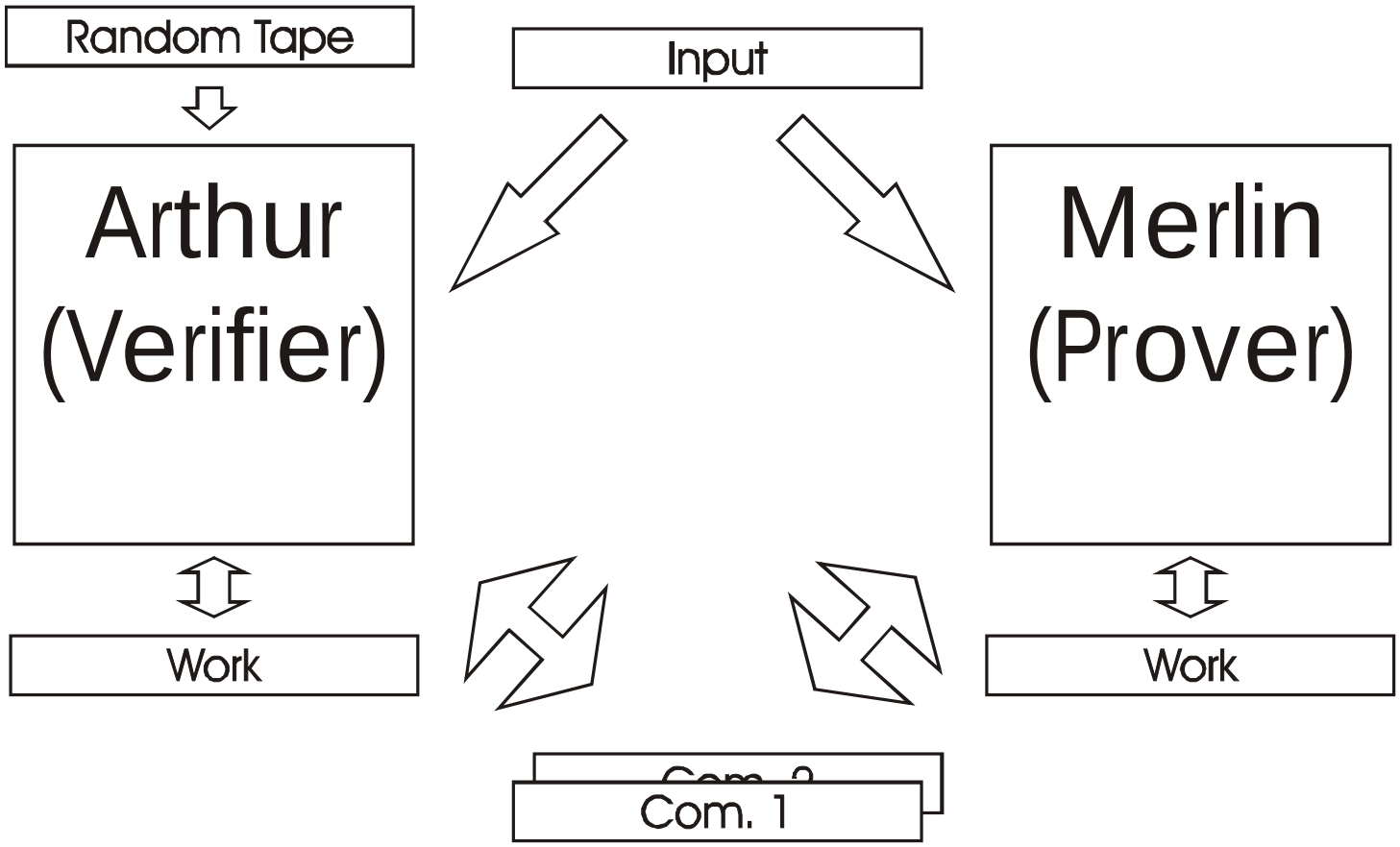
A/M Spiele

Grundsätzliches Protokoll:

- Eingabe: Sprache L, Element x; Frage: *x in L* ?
- Arthur berechnet basierend auf *x* und einer zufälligen Auswahl einen *Testfall* für Merlin, in der Absicht, Merlin zu widerlegen.
- Merlin liefert entweder true („*x* in *L*“) oder false.
- Arthur überprüft das Ergebnis.
  - § Stimmt es nicht, ist der Beweis falsch und Arthur gewinnt.
  - § Stimmt es, hat Merlin mit eventuell (gewisser W‘keit) betrogen.
    - – Iteration kann diese W‘keit nahe 0 bringen (Ziel: 1/3).

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

A/M Spiele



RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

Formal: Die Klasse AM(t(n))

Eine Sprache *L* ist in AM*(t(n))* gdw. ein Arthur/Merlin Proof System (A,M) existiert, so daß:

- Wenn *x in L*, dann akzeptiert A^M mit W‘keit > 2/3.
- Wenn *x nicht in L*, dann akzeptiert A^M mit W‘keit < 1/3.
- Für jeden Prover *M‘* wird die Berechnung $A^{M‘}$ nach höchstens *t(n)* Interaktionen fortgesetzt.
- Gebräuchlich auch die Notation AM(k) für k aus N.
- Beachte außerdem: Der Parameter steht für die Anzahl der Züge, nicht für die Anzahl der Runden!

Die Klasse AM, anschaulich

- In AM sind genau die Sprachen, für die ein Proof of Membership existiert, der durch ein A/M Spiel verifiziert werden kann.
- AM(t(n)) ist entsprechend die Klasse der Sprachen für die ein solches Spiel existiert, das höchstens t(n) Interaktionen pro Schritt erfordert.
- MA = MA(1): Merlin sendet, Arthur prüft, daher keine Interaktion
- AM = AM(2): Arthur wirft die Münze, sendet, Merlin sendet entsprechend

Die AM-Hierarchy kollabiert

Babei bewies 1985 die folgende Eigenschaft:

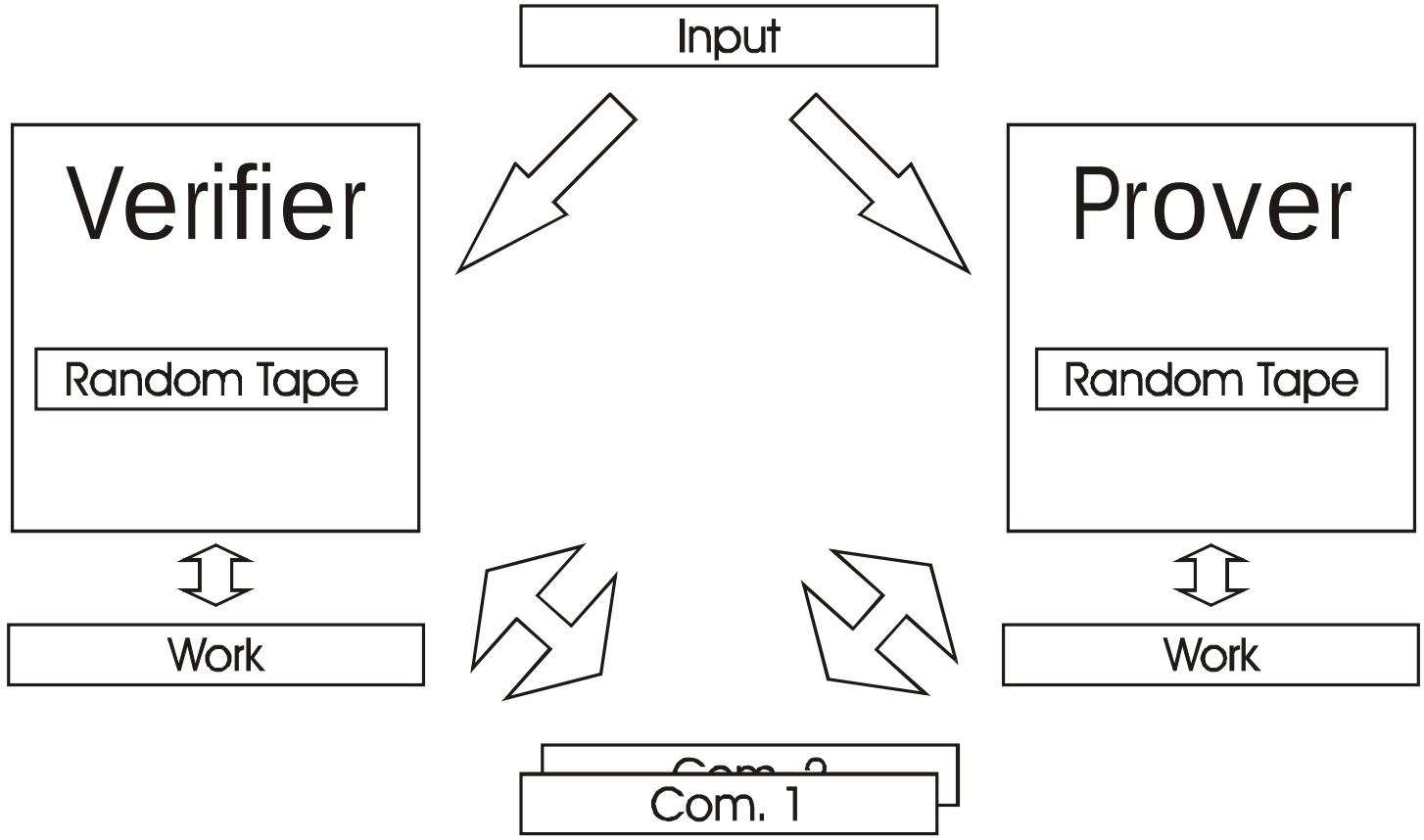
- Für alle k>2: AM(k) = AM(k+1)
- Daher die Bezeichnung: AM = AM(k) für alle k > 2.
- *Aber*: MA = MA(1) != AM
  - Anschaulich klar, denn MA beinhaltet eben *keine* Interaktion. Es gilt sogar *MA = NP*: Merlin sendet den Proof in einem Schritt zur polynomiellen Verifikation.
  - Weiterhin: *AM(1) = BPP*, (Bounded error probability in polynomial time) denn es wird keine Interaktion genutzt; *AM = BPNP* entsprechend

Interactive Proofs (IPs)

- Gleichzeitig, unabhängig entwickelt von Goldwasser, Micali und Rackoff.
- Einziger Unterschied:
  - Bei A/M Games sind die Randombits (Münzwürfe) öffentlich für den Prover, hier bei IPs jedoch *nicht*.
  - Bei IPs haben Prover und Verifier je ein random tape.
- Nach Goldwasser/Sipser (1986):
  - IPs und A/M Games sind äquivalent:
  - IP(t(n)) = AM(t(n)), speziell: IP(1) = MA, IP(2) = AM

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

Interactive Proof System



RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

Beispiel: Graph-Non-Isomorphie (NONISO), w'lich nicht in NP

- Geprüft werden soll, ob G_1,G_2 nicht isomorph sind.
- Verifier wählt n random bits c_i und n Permutationen H_i, je von G_1 gdw. c_i=0, und von G_2 sonst. *Die c_i sind geheim!*
- Verifier sendet die H_i zum Prover P.
- P prüft Isomorphie der H_i zu G_1 und G_2.
- Sind die G_j isomorph, so kann P für jedes H_i feststellen, ob H_i Permutation von G_1 oder G_2 ist.
- Anderenfalls muss P raten und macht mit W'keit 2^{-n} einen Fehler.

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

Formal: Die Klasse IP(t(n))

Eine Sprache *L* ist in *IP(t(n))* gdw. ein Interactive Proof System (V,P) existiert, so daß:

- Wenn *x in L*, dann akzeptiert V^P mit W‘keit > 2/3.
- Wenn *x nicht in L*, dann akzeptiert V^P mit W‘keit < 1/3.
- Für jeden Prover *P‘* wird die Berechnung $V^{P‘}$ nach höchstens *t(n)* Interaktionen fortgesetzt.
- Gebräuchlich auch die Notation IP(k) für k aus N..
- Beachte außerdem: Der Parameter steht für die Anzahl der Züge, nicht für die Anzahl der Runden!

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

Speziell: Die Klasse IP

Definition von IP:

$$IP = \bigcup_{k \geq 0} IP[n^k]$$

Liefert die folgenden Inklusionen:

$$NP \subseteq IP[2] \subseteq IP[t(n)] \subseteq IP[t(n)+1]$$

Private vs. public coin tosses

- Goldwasser et al. verglichen 1989 A/M Spiele mit IP Protokollen.
- Essentieller Unterschied: Bei A/M Spielen hat der Prover (Merlin) Einsicht in die Randombits des Verifiers (Arthur), beim IP-Protokoll sind diese bits *private* im Verifier
- Überraschend daher: IP(k) = BPNP = AM(k) für alle k > 1
- Aber für k = 1: AM(1) = BPP, IP(1) = MA(1) = NP

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

Anwendungen

In unserem Beispiel haben wir gezeigt:

$$NONISO \in IP(O(1))$$

Außerdem bekannt:

$$co - NP \notin IP(O(1))$$

Folgerungen:

- Graph-Non-Isomorphismus nicht *co*-NP hart
- Also auch: Graph-Isomorphismus nicht NP hart

Wichtiges Ergebnis 1989: Effizienter Test der Permanente (Lund, Fortnow, Karloff, Nisan protocol; LFKN)

Multi-Prover Interactive Proofs (MIPs)

- Wie IP, jedoch sind mehrere Prover (Merlins) zugelassen.
- Diese Prover kommunizieren nicht untereinander, jedoch hat der Verifier einen Kommunikationskanal zu allen.
- Ben-Or et al. zeigten 1988, daß MIP(k) = MIP(2) für alle k > 2, also sind nur genau zwei Prover nötig.
- Daher auch die Bezeichnung: MIP für MIP(k) für k>2
- Trotzdem: MIP(=NEXPTIME) ist echt mächtiger als IP.

1992, Arora/Safra

In den Folgejahren verschiedene Scale-Down Versuche:

- MIP = NEXPTIME => NP = LOGTIME ?
  - nicht wirklich, da V zumindest die Eingabe lesen muß
- Andere versuchen, die Eingabe zu verkleinern
- 1992 dann Arora/Safra:
  - Wie wirken sich Restriktionen auf die Memge der Random- und Query-Bits für Checker aus?
  - Lieferte neue Erkenntnisse für die Nicht-Approximierbarkeit des Clique-Problems

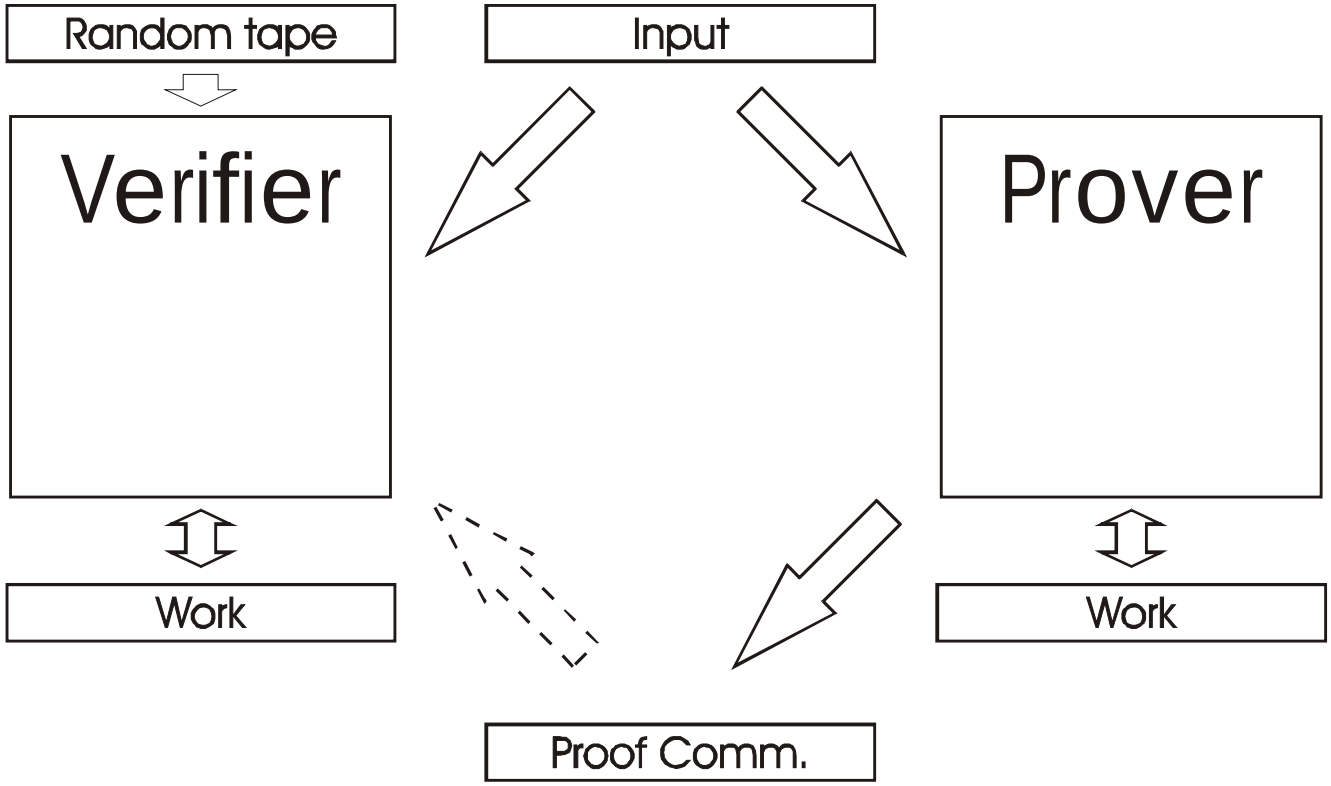
RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

Probabilistically Checkable Proofs (PCPs)

- Erlauben *keine* Interaktion aber *wohl* Randomisierung.
- System eher vergleichbar mit dem NP Prover/Verifier modell.
- Einzige Unterschiede zu diesem Modell.
  - Der Verifizierer ist randomisiert in *bestimmten* Maße.
  - Ein falscher Beweis wird (fälschlicherweise) akzeptiert mit gewisser W'keit P_- (standardmäßig ½).
  - Ferner: Ein korrekter Beweis wird mit P_+ (normalerweise 1) akzeptiert.

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE AACHEN
RWTH

Probabilistically Checkable Proofs (PCPs)



Verifier darf u.U. nur Teile des Proofs lesen.

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

Formal: Die Klasse PCP

Eine Sprache *L* ist in $PCP_{c,s}[r(n),q(m)]$, wenn ein randomisierter Verifier V^P existiert, der

- maximal O(*r(n)*) Random-Bits benutzt
- auf maximal O(*q(m)*) Bits des Beweises zugreift
- und die folgenden Eigenschaften aufweist:
  - Wenn x in L, akzeptiert V^P mit W‘keit >= *c,*
  - sonst akzeptiert er mit W‘keit <= *s*.
- Definitionen: $PCP(r(n),q(m)) = PCP_{1,1/2}(r(n),q(m))$, PCP(log(n),0) = NP, wie Herr Eckert zeigen wird

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
RWTH

Zusammenfassung, Ergebnisse

- Erweiterungen:
  A/M Games für mehr als einen Zug äquivalent zu Interactive Proof protocols und insbesondere auch alle reduzierbar auf Spiele mit zwei Zügen (einer Runde)
- Identitäten:
  AM(1) = BPP, IP(1) = NP
- Ohne Interaktion:
  $PCP_{1,1/2}(\log(n),0) = NP$
  „Nichtdeterminismus kann über W‘keitsverteilung deterministisch kodiert werden“

Fazit

- Die Klassen IP und MIP führen Interaktivität ein und schaffen somit neue, stabile, Einstufungen sowie neue Möglichkeiten der Reduktion für bekannt Probleme (z.B. Graphisomorphie).
- Mit PCP wird Randomisierung ohne Interaktivität berücksichtigt und die Query-/Randombits werden eingeschränkt.
- Insgesamt ergibt sich eine genaueres Klassenmodell mit feinerer Einteilung für alle Probleme „gerade über NP“.

Fragen ?