Empirische Analyse deterministischer Faktorisierungsmethoden

Systematische Bewertung klassischer und alternativer Ansätze

Zusammenfassung

Diese Arbeit dokumentiert empirische Ergebnisse aus systematischen Tests verschiedener Faktorisierungsalgorithmen. 37 Testfälle wurden mit Trial Division, Fermats Methode, Pollard Rho, Pollard $p-1$ und dem T0-Framework durchgeführt. Das primäre Ziel ist die Demonstration, dass deterministische Periodenfindung machbar ist. Alle Ergebnisse basieren auf direkten Messungen ohne theoretische Bewertungen oder Vergleiche.

Inhaltsverzeichnis

1 Methodik

1.1 Getestete Algorithmen

Die folgenden Faktorisierungsalgorithmen wurden implementiert und getestet:

1. **Trial Division**: Systematische Divisionsversuche bis $\sqrt{n}$
2. **Fermats Methode**: Suche nach Darstellung als Differenz von Quadraten
3. **Pollard Rho**: Probabilistische Periodenfindung in pseudozufälligen Sequenzen
4. **Pollard $p-1$**: Methode für Zahlen mit glatten Faktoren
5. **T0-Framework**: Deterministische Periodenfindung in modularer Exponentiation (klassisch Shor-inspiriert)

1.2 Testkonfiguration

Tabelle 1: Experimentelle Parameter

Parameter	Wert
Anzahl Testfälle	37
Timeout pro Test	2,0 Sekunden
Zahlenbereich	15 bis 16777213
Bitgröße	4 bis 24 Bits
Hardware	Standard Desktop-CPU
Wiederholungen	1 pro Kombination

1.3 Metriken

Für jeden Test wurden folgende Werte aufgezeichnet:

- **Erfolg/Misserfolg**: Binäres Ergebnis
- **Ausführungszeit**: Millisekundengenauigkeit
- **Gefundene Faktoren**: Für erfolgreiche Tests
- **Algorithmusspezifische Parameter**: Je nach Methode

2 T0-Framework Machbarkeitsdemonstation

2.1 Zweck der Implementierung

Die T0-Framework-Implementierung dient als Machbarkeitsnachweis, um zu demonstrieren, dass deterministische Periodenfindung technisch auf klassischer Hardware möglich ist.

2.2 Implementierungskomponenten

Das T0-Framework implementiert folgende Komponenten zur Demonstration deterministischer Periodenfindung:

```
class UniversalT0Algorithm:
def __init__(self):
self.xi_profiles = {
 'universal': Fraction(1, 100),
 'twin_prime_optimized': Fraction(1, 50),
 'medium_size': Fraction(1, 1000),
 'special_cases': Fraction(1, 42)
}
self.pi_fraction = Fraction(355, 113)
self.threshold = Fraction(1, 1000)
```

2.3 Adaptive ξ-Strategien

Das System verwendet verschiedene ξ-Parameter basierend auf Zahleneigenschaften:

Tabelle 2: ξ-Strategien im T0-Framework

Strategie	ξ-Wert	Anwendung
twin_prime_optimized	1/50	Zwillingsprim-Semiprims
universal	1/100	Allgemeine Semiprims
medium_size	1/1000	Mittelgroße Zahlen
special_cases	1/42	Mathematische Konstanten

2.4 Resonanzberechnung

Die Resonanzbewertung wird mit exakter rationaler Arithmetik durchgeführt:

$$\omega = \frac{2 \cdot \pi_{\text{ratio}}}{r} \tag{1}$$

$$R(r) = \frac{1}{1 + \left|\frac{-(\omega-\pi)^2}{4\xi}\right|} \tag{2}$$

3 Experimentelle Ergebnisse: Machbarkeitsnachweis

Die experimentellen Ergebnisse dienen der Demonstration der Machbarkeit deterministischer Periodenfindung anstatt dem Vergleich algorithmischer Leistung.

3.1 Erfolgsraten nach Algorithmus

Tabelle 3: Gesamte Erfolgsraten aller Algorithmen

Algorithmus	Erfolgreiche Tests	Erfolgsrate (%)
Trial Division	37/37	100,0
Fermat	37/37	100,0
Pollard Rho	36/37	97,3
Pollard $p-1$	12/37	32,4
T0-Adaptive	31/37	83,8

4 Periodenbasierte Faktorisierung: T0, Pollard Rho und Shors Algorithmus

4.1 Vergleich der Periodenfindungsansätze

T0-Framework, Pollard Rho und Shors Quantenalgorithmus sind alle periodenfindende Algorithmen mit verschiedenen Rechenbarkeitssystemen:

4.2 Gemeinsames Periodenfindungsprinzip

Alle drei Algorithmen nutzen dieselbe mathematische Grundlage:

- **Kernidee**: Finde Periode r wobei $a^r \equiv 1 \pmod{n}$
- **Faktorextraktion**: Nutze Periode um $\gcd(a^{r/2} \pm 1, n)$ zu berechnen
- **Mathematische Basis**: Eulers Theorem und Ordnung von Elementen in $\mathbb{Z}_n^*$

4.3 Theoretische Komplexitätsanalyse

Sowohl T0-Framework als auch Shors Algorithmus teilen denselben theoretischen Komplexitätsvorteil:

- **Periodensuchraum**: Beide suchen nach Perioden r wobei $a^r \equiv 1 \pmod{n}$
- **Maximale Periode**: Die Ordnung jedes Elements ist höchstens $n - 1$, aber typischerweise viel kleiner
- **Erwartete Periodenlänge**: $O(\log n)$ für die meisten Elemente aufgrund Eulers Theorem
- **Periodentest**: Jeder Periodentest benötigt $O((\log n)^2)$ Operationen für modulare Exponentiation
- **Gesamtkomplexität**: $O(\log n) \times O((\log n)^2) = O((\log n)^3)$

4.4 Der gemeinsame polynomiale Vorteil

Sowohl T0 als auch Shors Algorithmus erreichen denselben theoretischen Durchbruch:

$$\text{Klassisch exponentiell: } O(2^{\sqrt{\log n \log \log n}}) \to \text{Polynomial: } O((\log n)^3) \tag{3}$$

Die Schlüsselerkenntnis ist, dass **beide Algorithmen dieselbe mathematische Struktur ausnutzen**:

- Periodenfindung in der Gruppe $\mathbb{Z}_n^*$
- Erwartete Periodenlänge $O(\log n)$ aufgrund glatter Zahlen

- Polynomialzeit-Periodenverifikation
- Identische Faktorextraktionsmethode

Der einzige Unterschied: Shor nutzt Quantensuperposition um Perioden parallel zu suchen, während T0 sie deterministisch sequenziell sucht - aber beide haben dieselbe $O((\log n)^3)$ Komplexitätsgrenze.

4.5 Das Implementierungsparadoxon

Sowohl T0 als auch Shors Algorithmus demonstrieren ein fundamentales Paradoxon in fortgeschrittener Algorithmusentwicklung:

Math. Muster	**NN-Lernziel**
Zwillingsprimstruktur	Vorhersage $\xi = 1/50$ Strategie
Primlückenverteilung	Schätzung Resonanzclustering
Glätteindikatoren	Vorhersage Periodenverteilung
Math. Konstanten	ID Multi-Resonanzmuster
Carmichael-Muster	Schätzung max. Periodengrenzen
Faktorgrößenverhältnisse	Vorhersage opt. Basisauswahl

4.5.1 Gemeinsame Implementierungsmängel

- **Shors Quantenaufwand**:
  - Quantenfehlerkorrektur benötigt $\sim 10^6$ physische Qubits pro logischem Qubit
  - Dekohärenzzeiten begrenzen Algorithmusausführung
  - Aktuelle Systeme: 1000 Qubits $\rightarrow$ Benötigt: 10^9 Qubits für RSA-2048
- **T0s klassischer Aufwand**:
  - Exakte rationale Arithmetik: Bruchobjekte wachsen exponentiell in der Größe
  - Resonanzbewertung: Komplexe mathematische Operationen pro Periode
  - Adaptive Parameteranpassung: Multiple ξ-Strategien erhöhen Berechnungskosten

5 Philosophische Implikationen: Information und Determinismus

5.1 Intrinsische mathematische Information

Eine entscheidende Erkenntnis ergibt sich aus dieser Analyse, die über Berechnungskomplexität hinausgeht:

Methode	Beschreibung
trial_division	Klassische systematische Division
fermat	Differenz-der-Quadrate-Methode
pollard_rho	Probabilistische Zykluserkennung
pollard_p_minus_1	Glatte-Faktoren-Methode
t0_classic	Original T0 ($\xi = 1/100000$)
t0_universal	Revolutionäres universelles T0 ($\xi = 1/100$)
t0_adaptive	Intelligente ξ-Strategieauswahl
t0_medium_size	Optimiert für N > 1000 ($\xi = 1/1000$)
t0_special_cases	Für spezielle Zahlen ($\xi = 1/42$)

5.2 Vibrationsmodi und prädiktive Muster

Eine tiefere Analyse zeigt, dass die Zahlengröße die möglichen „Vibrationsmodiïn der Faktorisierung beschränkt:

Parameter	**Wert**
Timeout pro Algorithmus	2,0-10,0 Sekunden (methodenabhängig)
T0-Timeout-Erweiterung	15,0 Sekunden (Komplexitätsbetrachtung)
Messgenauigkeit	Millisekundenzeitnahme
Erfolgsverifikation	Faktorproduktvalidierung
Resonanzschwelle	ξ-abhängig (typisch 1/1000)
Maximal getestete Perioden	500-2000 (größenabhängig)

5.2.1 Eingeschränkter Schwingungsraum

Für eine Zahl n mit $k = \log_2(n)$ Bits:

- **Maximale Periode**: $r_{\max} = \lambda(n) \leq n - 1$ (Carmichael-Funktion)
- **Typischer Periodenbereich**: $r_{typical} \in [1, O(\sqrt{n})]$ für die meisten Basen
- **Resonanzfrequenzen**: $\omega = 2\pi/r$ beschränkt auf diskrete Werte
- **Vibrationsmodi**: Nur $O(\sqrt{n})$ unterschiedliche Schwingungsmuster möglich

5.3 Das begrenzte Universum der Schwingungen

$$\Omega_n = \left\{\omega_r = \frac{2\pi}{r} : r \in \mathbb{Z}, 2 \leq r \leq \lambda(n)\right\} \tag{4}$$

Dieser Frequenzraum Ω_n ist:

- **Endlich**: Durch Zahlengröße beschränkt
- **Diskret**: Nur ganzzahlige Perioden erlaubt
- **Strukturiert**: Folgt mathematischen Mustern basierend auf ns Primstruktur
- **Vorhersagbar**: Resonanzspitzen clustern in mathematisch bestimmten Bereichen

Zahlenkategorie	Optimales ξ	Erfolgsrate
Zwillingsprims	1/50	95%
Universal (Alle Typen)	1/100	83,8%
Mittelgroß ($N > 1000$)	1/1000	78%
Spezialfälle	1/42	67%
Klassisch nur Zwillinge	1/100000	45%