RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
RWTH

Fingerprinting Techniques

Christoph Bürger

Gliederung

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
RWTH

1. Einführung
2. Matrix-Multiplikation
3. Identität von Polynomen
4. Perfekte Matchings in Graphen
5. Gleichheit von Strings
6. Pattern Matching
7. Zusammenfassung und Ausblick

Einführung

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
RWTH

- komplexe Probleme z.T. durch gute, aber komplizierte Algorithmen gelöst
- Korrektheit dieser Algorithmen ist nicht einfach zu verifizieren
- Programmverifikation, etc. sehr schwierig
- Alternative: Überprüfung des Ergebnisses eines Algorithmus auf Korrektheit.

Einführung

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
RWTH

- Ergebnis eines Programms aus einem sehr großen Universum U
- Problem besteht im Vergleich zweier Elemente x und y aus U
- Gleichheit ist in log(|U|) überprüfbar
- Verbesserung: Mapping vom Universum U in deutlich kleineres Universum V
  und Vergleich der Bilder von x und y
- in Zeit log(|V|) möglich
- hohe Wahrscheinlichkeit dafür, dass $x = y \Leftrightarrow Bild(x) = Bild(y)$
- die Bilder von x und y heißen ***Fingerprints*** von x bzw. y

Matrix-Multiplikation

Gegeben: $n \times n$-Matrizen A, B, C

Behauptung: AB=C

naiver Algorithmus läuft in Zeit $O(n^3)$

bester (bekannter) Algorithmus läuft in Zeit $O(n^{2,376})$

einfachste Testmöglichkeit: mit naivem Algorithmus nachrechnen

Problem: Verbesserung durch schnelleren Algorithmus würde verloren gehen

Vorstellung eines randomisierten $O(n^2)$-Zeit Algorithmus mit begrenzter Fehlerwahrscheinlichkeit

Matrix-Multiplikation (Freivalds Technik)

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

Verfahren:

Geg.: $n \times n$ *-Matrizen A, B, C*

1. Wähle Zufallsvektor $r \in \{0,1\}^n$
2. berechne $x = Br$
3. berechne $y = Ax = ABr$
4. berechne $z = Cr$
5. wenn y=z, dann Ausgabe „korrekt“, sonst Ausgabe „falsch“

Matrix-Multiplikation

Klar: Wenn AB=C, dann auch y=z

Theorem: Seien A, B, C $n \times n$ -Matrizen, mit $AB \neq C$. Dann gilt für ein gleichverteilt zufällig gewähltes $r \in \{0,1\}^n$,

$$\Pr(ABr = Cr) \leq \frac{1}{2}$$

k-fache Iteration des Tests verringert Fehlerwahrscheinlichkeit auf Wert

$$\leq \frac{1}{2^k}$$

Produkte von Polynomen

- Geg.: $P_1(x), P_2(x), P_3(x) \in F[x]$
- Beh.: $P_1(x) \times P_2(x) = P_3(x)$

- $\max(Grad(P_1(x)), Grad(P_2(x))) \leq n \Rightarrow Grad(P_3(x)) \leq 2n$

- Multiplikation ist in Zeit $O(n \log n)$ möglich

- Auswertung eines Polynoms in einem Punkt n in Zeit $O(n)$ möglich
- Verfahren:
  - wähle $S \subseteq F, |S| \geq 2n+1$
  - wähle zufällig $r \in S$
  - falls $P_1(r)P_2(r) = P_3(r)$: Ausgabe „korrekt“, ansonsten Ausgabe „falsch“

Produkte von Polynomen

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

- setze $Q(x) = P_1(x)P_2(x) - P_3(x)$
- $Grad(Q(x)) \leq 2n$
- wenn $Q(x) \neq 0$, dann ist mit hoher Wahrscheinlichkeit $Q(r) \neq 0$
- Q hat höchstens 2n Wurzeln
- 2n verschiedene Wahlen von $r \in S$ können $Q(r) = 0$ verursachen

$$\Rightarrow \Pr(Fehler) = \frac{2n}{|S|}$$

- passende Wahl von S bzw. Iterationen des Algorithmus führen zu kleinen Fehlerwahrscheinlichkeiten
- det. Variante möglich, indem alle $r \in S$einmal ausprobiert werden
- aber benötigt 2n+1 Berechnungen $\Rightarrow$ Gesamtlaufzeit $\Theta(n \log^2 n)$
- Multiplikation der symbolischen Polynome wäre einfacher

Identität von Polynomen

- vorgestelltes Verfahren auch geeignet, um Polynome zu vergleichen
- Vergleich explizit gegebener Polynome in O(n) möglich
- $Q(x) = P_1(x) - P_2(x) = 0$?
- Verfahren vorteilhaft, wenn Polynome nur implizit gegeben

Bsp.: Determinante einer symbolischen Matrix als Identität multivariater Polynome lösbar

Vandermonde-Matrix M

$$\det(M) = \prod_{j<i} (x_i - x_j)$$

$$Q(x_1, \ldots, x_n) = \det(M) - \prod_{j<i} (x_i - x_j)$$

Technik: wähle zufällige Werte für jedes x_i und teste Q=0

Schwartz-Zippel Theorem

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

• Formalisierung als Erweiterung von Freivalds Technik auf multivariate Polynome

• $Q(x_1,\ldots,x_n)$ der Grad eines Terms eines multivar. Polynoms ist die Summe der Exponenten der Variablen des Terms

• Grad des Polynoms: Maximum der Grade der Terme

Theorem (Schwartz-Zippel):

Sei $Q(x_1,\ldots,x_n) \in F[x_1,\ldots,x_n]$ ein multivariates Polynom mit absolutem Grad d. Sei $S \subseteq F$ und $r_1,\ldots,r_n$ unabhängig, zufällig aus S gewählt.

Dann gilt:

$$\Pr[Q(r_1,\ldots,r_n) = 0 \mid Q(x_1,\ldots,x_n) \neq 0] \leq \frac{d}{|S|}$$

• Potentielles Problem: in der Berechnung sehr große Werte als Ergebnisse der Polynome

• Lösung: Berechnungen modulo einer „kleinen“ Primzahl p

Perfekte Matchings in Graphen

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

- Geg.: bipartiter Graph G(U,V,E) mit unabhängigen Knotenmengen $U = \{u_1, \ldots, u_n\}$ und $V = \{v_1, \ldots, v_n\}$
- Ein Matching ist eine Sammlung von Kanten $M \subseteq E$ so dass jeder Knoten höchstens einmal in M vorkommt
- Ein perfektes Matching ist ein Matching der Größe n
- Jedes perfekte Matching kann als Permutation von U in V gesehen werden

- Die perfekten Matchings in G können in eine 1-zu-1 Korrespondenz mit den Permuationen in S_n gesetzt werden, wobei das Matching, das zu einer Permutation $\pi \in S_n$ korrespondiert durch die Paare $(u_i, v_{\pi(i)})$ gegeben ist

Perfekte Matchings in Graphen (Edmonds Theorem)

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

Theorem (Edmonds Theorem):

Sei A die wie folgt aus G entstehende $n \times n$ -Matrix:

$$A_{ij} = \begin{cases} x_{ij} & (u_i, v_j) \in E \\ 0 & (u_i, v_j) \notin E \end{cases}$$

Sei $Q(x_{11}, x_{12}, \ldots, x_{nn}) = \det(A)$

Dann hat G ein perfektes Matchings gdw. $Q \neq 0$

Entsprechend Abschnitt 3 lässt sich nun ein einfaches Verfahren zum Test der Existenz eines perfekten Matchings konstruieren

- Laufzeit des Algorithmus von der Berechnung der Determinante dominiert
- Da gute Algorithmen zu Matching-Berechnung existieren bringt Verfahren kaum Vorteile

Gleichheit von Strings

- Alice & Bob besitzen Kopien derselben Daten(bank)
- periodischer Abgleich notwendig, aber Kommunikation sehr teuer

- Alices Daten: Bitsequenz $(a_1,\ldots,a_n)$
- Bobs Daten: Bitsequenz $(b_1,\ldots,b_n)$
- deterministischer Konsistenz-Check mit weniger als n Bits wird scheitern

- Fingerprint-Verfahren:

interpretiere Daten als n-Bit Integer a und b

$$a = \sum_{i=1}^{n} a_i 2^{i-1} \qquad b = \sum_{i=1}^{n} b_i 2^{i-1}$$

Fingerprint-Funktion: $F_p(x) = x \bmod p$ für Primzahl p

Annahme: $a \neq b \Rightarrow F_p(a) \neq F_p(b)$

Anzahl der zu übertragenden Bits: $O(\log p)$

Gleichheit von Strings

- Strategie ist für festes p leicht zu überlisten
- Lösung: wähle p zufällig

- für jedes k sei $\pi(k) \approx \frac{k}{\ln k}$ die Anzahl der Primzahlen kleiner k

- $c = |a - b|$
- Verfahren versagt nur, wenn $c \neq 0$ und „p teilt c“
- $N = 2^n \qquad c \leq N$

- **Lemma:** Die Anzahl der Primteiler einer beliebigen Zahl $< 2^n$ ist höchstens n

- Wähle Schwelle $\tau > n(= \log N)$

Gleichheit von Strings

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

- höchstens n Primzahlen kleiner τ können Teiler von c sein
- wähle $p < \tau$ um Fingerprint-Funktion zu definieren
- Anzahl der Kommunikationsbits: $O(\log \tau)$
- wähle $\tau = tn \log tn$ für großes t
- **Theorem:** $\Pr[F_p(a) = F_p(b) \mid a \neq b] \leq \frac{n}{\pi(\tau)} = O\left(\frac{1}{t}\right)$
- Fehlerwahrscheinlichkeit $O\left(\frac{1}{t}\right)$
- zu übertragende Bits: $O(\log t + \log n)$

Pattern Matching

text := $X = x_1 \dots x_n$ über festem Alphabet Σ, $m \leq n$

pattern := $Y = y_1 \dots y_m$

o.B.d.A. $\Sigma = \{0,1\}$

pattern kommt in text von, wenn ein j ex. mit $j \in \{1,2,\dots,n-m+1\}$ so dass für $1 \leq i \leq m$, $x_{j+i-1} = y_i$

triviale Lösung in $O(nm)$

guter deterministischer Algorithmus in $O(n+m)$

Definiere sub-string $X(j) = x_j x_{j+1} \dots x_{j+m-1}$

Match liegt vor, wenn j ex. mit $Y = X(j)$

kleinstes j, das Bedingung erfüllt, führt zu Eindeutigkeit der Lösung

Pattern Matching

- Brute-Force Verfahren: vergleiche Y mit jedem X(j)
- Randomisiertes Verfahren: wähle Fingerprint F(Y) und vergleiche ihn mit jedem F(X(j))

Fehler: F(Y)=F(X(j)) und $Y \neq X(j)$

wähle F so, dass Fehlerwahrscheinlichkeit klein ist

Wahl von F: für jeden String $Z \in \{0,1\}^m$, interpretiere Z als m-bit Integer

$$\Rightarrow F_p(Z) = Z \bmod p$$

interpretiere Y und X(j) als Integer und vergleiche $F_p(Y)$ und $F_p(X(j))$

$$\Pr[F_p(Y) = F_p(X(j)) \mid Y \neq X(j)] \leq \frac{m}{\pi(\tau)} = O\left(\frac{m \log \tau}{\tau}\right)$$

Fehlerwahrscheinlichkeit für eine der max. n Werte von j ist $O\left(\frac{nm \log \tau}{\tau}\right)$

Pattern Matching

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

- wähle $\tau = n^2 m \log n^2 m \Rightarrow \Pr[\text{false match}] = O\left(\frac{1}{n}\right)$

- Monte Carlo Algorithmus:

  vergleiche Fingerprints aller X(j) mit dem von Y und gebe erstes j aus, für das Match eintritt

- **Theorem:** Der Monte Carlo Algorithmus hat eine Laufzeit von $O(n+m)$ und eine Fehlerwahrscheinlichkeit von $O\left(\frac{1}{n}\right)$

- Las Vegas Algorithmus:

  wenn Match zwischen den Fingerprints von Y und X(j) vorliegt, vergleiche Y und X(j) in Zeit O(m). Wenn dabei false-match auftritt, Algorithmus abbrechen und brute-force Methode anwenden

- Algorithmus ist fehlerfrei mit erwarteter Laufzeit $O(n+m)$

Fingerprinting Techniken

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE RWTH

- Im wesentlichen 2 verschiedene Techniken
- Geg.: Strings bzw. Vektoren $a = (a_1, \ldots, a_n)$ und $b = (b_1, \ldots, b_n)$

$$a_i, b_i \in \Sigma, \Sigma \text{ endlich}$$

- Alphabet kodierbar mit Zahlenmenge $\Gamma = \{0,1,\ldots,k-1\}$ mit $k \in |\Sigma|$
- Polynome: $A(z) = \sum_{i=0}^{n-1} a_i z^i$, $B(z) = \sum_{i=0}^{n-1} b_i z^i$ mit Integer - Koeffizienten und Grad $\leq$ n
- Ziel: $a = b \Leftrightarrow A(z) = B(z)$

Fingerprinting Techniken I

- Wähle feste Primzahl $p > 2n$ und $p > k$
- $A(z), B(z) \in Z_p[z], \Gamma \subseteq Z_p$
- Fingerprint: wähle zufälliges $r \in Z_p$ und berechne $A(r), B(r)$
- $a = b \Rightarrow A(z) = B(z) \Rightarrow A(r) = B(r)$
- $\Pr[A(r) = B(r) \mid a \neq b] \leq \frac{n}{p} \leq \frac{1}{2}$ für gewähltes p

Feld fixieren und zufälligen Berechnungspunkt wählen

Fingerprinting Techniken II

- Wähle festes z=2
- Wähle zufällige Primzahl q genügend klein
- Fingerprints: berechne A(2) und B(2) über Z_q

Evaluierungspunkt fixieren und zufällig Feld wählen

Fingerprinting Techniken III

- Annahme: k=2
- Fixiere Primzahl $p > 2^n$
- wähle zufälliges Polynom P(z) über Z_p
- Fingerprints: bestimme P(a), P(b) über Z_p
- dann Reduktion modulo einer Zahl nahe $\log n$ (universal hash functions)

Zusammenfassung

RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
RWTH

- Vorstellung von Fingerprinting-Techniken für
  - Matrix-Multiplikation
  - Identitäten von Polynomen
  - Perfekte Matchings in Graphen
  - Gleichheit von Strings
  - Pattern Matching