Aufgaben zum Tut am 24.07.2006

Thomas Pajor

24. Juli 2006

Aufgabe 1.

Aus der Info IV Probeklausur 2005 von Carmen Stüber

Eve hört eine RSA–verschlüsselte Nachricht ab, die sie gerne entschlüsseln möchte. Dazu muss sie den RSA–Modul N (Teil des öffentlichen Schlüssels) faktorisieren. Sie weiß, dass N das Produkt zweier Primzahlen p und q im Bereich von 2 bis 2^{1024} ist. Eve entwirft nun einen Algorithmus, der eine reelle Zahl N zwischen 2 und 2^{2048} als Eingabe hat, und den kleinsten Primteiler von $\lfloor N \rfloor$ ausgibt.

Bestimmen Sie eine möglichst genaue untere Schranke für einen solchen Algorithmus im worst–case. Beweisen Sie diese mit dem aus der Vorlesung bekannten Lower–Bound–Theorem.

Hinweis: Für eine positive Zahl n gilt: Es gibt $\Theta(\frac{n}{\ln n})$ Primzahlen, die kleiner als n sind. Desweiteren ist 1 keine Primzahl.

Lösung.

Die Funktion, die von Eve's Algorithmus berechnet wird ist

$$f(N) := \text{„kleinster Primteiler von } \lfloor N \rfloor\text{“}$$

Sei nun p_i die i–te Primzahl, also $p_1 = 2$, $p_2 = 3$, $p_3 = 5$ und so weiter. Weiterhin sei q die Anzahl Primzahlen $< 2^{2048}$. Da $f(N)$ immer eine Primzahl zwischen 2 und 2^{2048} liefert, können wir uns nun q rationale Funktionen definieren:

$$Q_i(N) := p_i$$

Weiterhin definieren wir uns q Stützstellen

$$X_i := p_i + \frac{1}{2}$$

Wir haben damit q (also ungefähr $\frac{2^{2048}}{\ln 2^{2048}}$) verschiedene Stützstellen bzw. und rationale Funktionen konstruiert. Es gilt nun

$$\forall X \in U(X_i, \varepsilon) : f(X) = Q_i(X) \qquad i = 1, \ldots, q$$

zum Beispiel wenn wir $\varepsilon := \frac{1}{3}$ wählen. Damit gilt für jede der Stützstellen, dass $X_i + \frac{1}{3}$ nicht „zu groß“ wird, so dass $f(X)$ nicht schon 2 als kleinsten Primteiler liefern würde[1]. Des Weiteren ist $X_i - \frac{1}{3} \geq p_i$, und damit das Ergebnis von $f(X)$ ebenfalls noch p_i.

Somit folgt mit Satz A, dass die Anzahl Vergleiche R, die mindestens nötig sind um $f(X)$ im RAM–Modell zu berechnen

$$\log_2 q$$

ist. Mit dem Hinweis sind also

$$\Omega\big(\log_2 \frac{2^{2048}}{\ln 2^{2048}}\big) = \Omega\big(\log_2(e) 2^{2048}\big)$$

vergleiche nötig.

Aufgabe 2.

Aus der Info IV Probeklausur 2005 von Carmen Stüber

Gegeben sei ein Zellraster mit n Zellen sowie natürliche Zahlen k, n mit $n > 3$ und $k < n$.

(a) Wie viele Möglichkeiten gibt es für k Schlüssel, durch Hashing in die n Zellen einsortiert zu werden?

(b) Wie viele Möglichkeiten gibt es noch, wenn man nur kollisionsfreie Hashfunktionen betrachtet?

(c) Wie hoch ist die Wahrscheinlichkeit, dass beim gleichverteilten Hashing von $k = n$ Schlüsseln in die n Zellen die dritte Zelle leer bleibt?

Lösung.

(a) Da wir hier eine allgemeine Hashfunktion betrachten, gibt es für jeden Schlüssel n Möglichkeiten in eine Hashzelle zu geraten. Schließlich haben wir ja n Hashzellen, und jede könnte ausgewählt werden. Da dies für jeden der k Schlüssel gilt, gibt es für k Schlüssel gerade n^k Möglichkeiten in die n Zellen einsortiert zu werden.

[1] Wenn ε zu groß ist, zum Beispiel $\frac{2}{3}$, dann ist $\lfloor X - \varepsilon \rfloor = p_i + 1$, und da p_i ungerade ist, wäre $p_i + 1$ gerade, und damit der kleinste Primteiler 2.

(b) Wenn wir keine Kollisionen erlauben, so darf in jeder Hashzelle höchstens ein Schlüssel enthalten sein. Das heißt die k Schlüssel belegen insgesamt genau k der n Hashzellen. Die Anzahl Möglichkeiten eine k Elementige Teilmenge aus einer n–Menge auszuwählen ist gerade $\binom{n}{k}$. Es gibt somit $\binom{n}{k}$ Möglichkeiten die k Schlüssel kollisionsfrei auf n Hashzellen zu verteilen.

(c) Es gilt nun $n = k$. Die Wahrscheinlichkeit, dass die dritte Hashzelle leer bleibt können wir mit dem Ansatz „Anzahl günstige Fälle / Anzahl mögliche Fälle“ berechnen. Die Anzahl möglicher Fälle die n Schlüssel auf n Hashzellen zu verteilen ist, nach Aufgabe a, gerade n^n. Da wir wollen, dass die dritte Hashzelle leer bleiben soll, berechnen wir als Anzahl günstiger Fälle die Anzahl Möglichkeiten n Schlüssel auf $n-1$ Hashzellen zu verteilen[2]. Wir erhalten somit nach Aufgabe a $(n-1)^n$. Es gilt somit insgesamt

$$P(\text{3. Zelle bleibt leer}) = \frac{\#\text{ günstige Fälle}}{\#\text{ mögliche Fälle}} = \frac{(n-1)^n}{n^n} = \left(\frac{n-1}{n}\right)^n$$

[2] Also alle Hashzellen bis auf die dritte