Musterlösung
Übungszettel 3
Algorithmen & Datenstrukturen für Bioinformatiker

Adrian Haß

10. Dezember 2002

1 NFA's

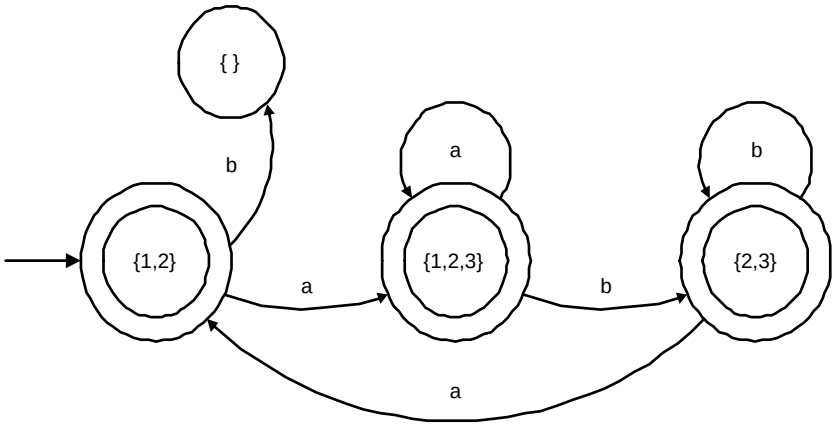


Abbildung 1: dfa durch Potenzmengen-Konstruktion vom nfa

Hier bei bedeutet der mit bezeichnete Zustand einen Fehlerzustand. Man hätte Ihn auch weglassen können, denn es ist klar, dass ein dfa nicht akzeptiert, wenn von einem Knoten keine Kante ausgeht, die mit dem aktuell gelesenen Buchstaben bezeichnet ist.

2 Match-Count

Hier ist die sinnvollste Vorgehensweise eine Vorverarbeitung der beiden Strings α und β in $O(n^2)$, die es uns erlaubt beliebige Match–Count–Anfragen danach in konstanter Zeit zu beantworten.

Dafür überlegt man sich, dass ich doch einen Teilstring $S[i..j]$ eines Wortes S beschreiben kann als Präfix $S[1..j]$ ohne den Präfix $S[1..i]$ oder analog für Suffix–Paare. Dies lässt sich leicht auf die Match–Counts (MC) übertragen: wenn ich den MC für die Präfixe kenne, dann ist der MC des Teilworts genau MC(Präfix lang)-MC(Präfix kurz).

Dies erreichen wir mit folgender Vorverarbeitung:

- Bestimme Matches von $A[n]$ an allen Stellen $B[k]$ und speichere sie in Matrix M an $M(1,k)$.
- Bestimme nun Matches von $A[m-1..m]$ (2-er Suffix), indem wir $A[m-1]$ explizit mit $B[k]$ vergleichen und für $A[m]$ $B[k+1]$ aus bereits berechneter 1.Zeile um Eins nach links verschobene Werte addieren: (Vergleich $A[m]$ mit $B[k+1]$ haben wir ja bereits gemacht.)
  Man kann sich das ganze einfach so vorstellen, als schieben wir $A[m-1..m]$ an B vorbei.
- Dies für jeden Suffix bis wir in der letzten Zeile ganz A an B "vorbeigeschoben" haben.
- Für Anfragen für Matches die nicht Suffices sind, berechnen wir die Matches aus der Matrix in konstater Zeit mit

$$\mathrm{mc}(A[r,r+k],B[s,s+k]) = M(n-r+1,s) - M(n-(r+k),s+k+1)$$

Zur Veranschaulichung ein Beispiel mit $A =$ `ggCCTTA` und $B =$ `ACCgTTA`, dann ergibt sich:

mc	A	C	C	g	T	T	A
`A`	1	0	0	0	0	0	1
`TA`	0	0	0	0	1	2	0
`TTA`	0	0	0	1	3	1	0
`CTTA`	0	1	2	3	1	0	0
`CCTTA`	1	3	4	1	0	0	0
`gCCTTA`	3	4	1	1	0	0	0
`ggCCTTA`	4	1	1	1	0	0	0

Suchen wir nun ein Match–Count zwischen $A[4..6] =$ `CCT` und $B[3..5] =$ `CgT`, dann rechnen wir:

(`CTTA` an Stelle 3 minus `A` an Stelle 6)

$$\mathrm{mc}(A[4..6],B[3..5]) = M(7-4+1,3) - M(7-6,5+1) = M(4,4) - M(1,6) = 2-0$$

und erhalten, dass beim Hinsehen als richtig erkannte Ergebnis.

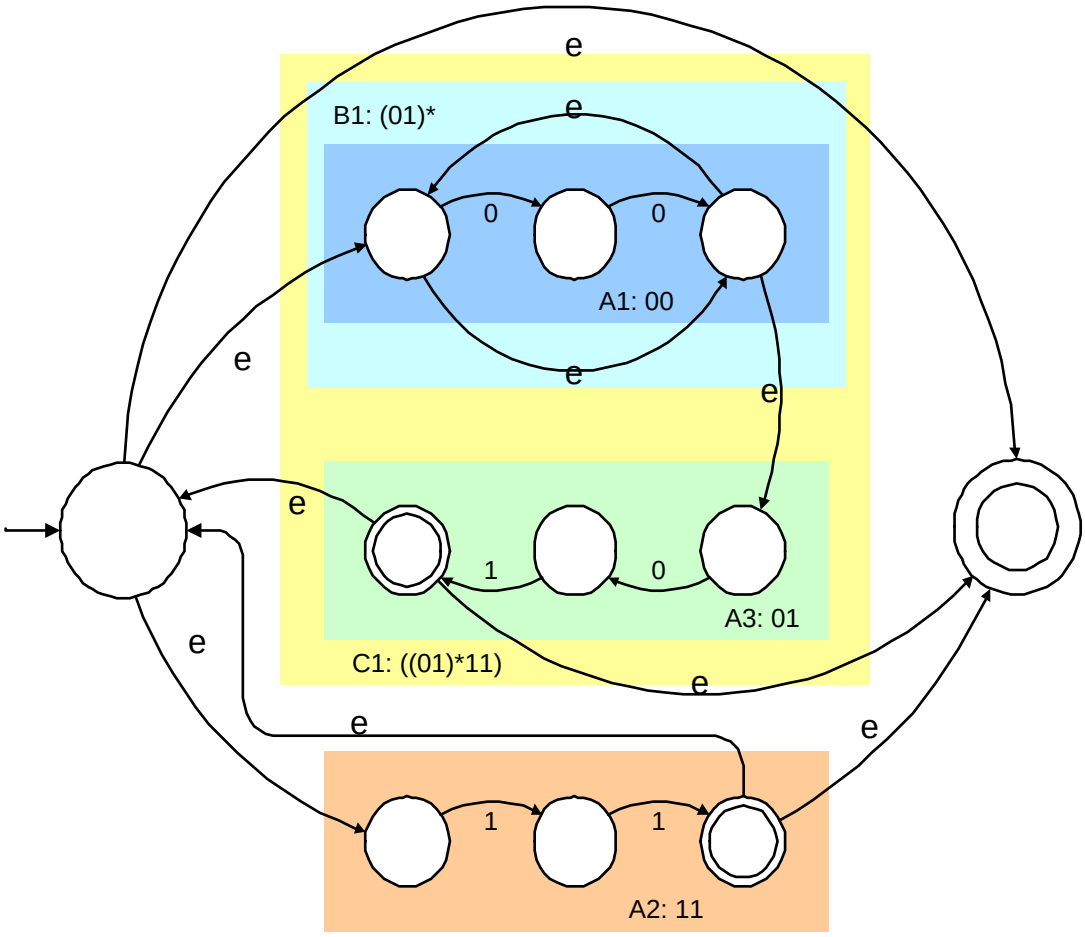
3 Gemeinsame Medianbestimmung in $O(\log n)$

Zu Beginn noch einmal die Definition des Median einer endlichen Menge von Werten:

> Der Median einer geordneten, endlichen Folge von Größen ist das Element, der diese in zwei gleichmächtige Teile spaltet. I.A. wählt man bei grader Anzahl an Elementen, das arithmetische Mittel der beiden der Mitte nächsten Elemente, im Prinzip würde es jedoch jede Zahl dazwischen tun.

Man verfährt nun wie folgt:

- Man vergleicht die Mediane $a = m(A[1..n])$ und $b = m(B[1..n])$ und folgert aus $a < b$, dass der gemeinsame Median von A und B nicht mehr rechts von a in A liegen kann und nicht mehr links von b in B. Analog beim anderen Fall. Wichtig hierbei ist jedoch: *habe ich den Mittelwert einer Liste durch ein arithm. Mittel gewonnen, dann Suche ich im Bereich, der beide Werte mit einschliesst*!
- Somit reicht im zweiten Schritt ein Median–Vergleich zwischen $A[m..n]$ und $B[1..m]$, wobei m die Position des Medians im letzten Vergleich war, korrigiert um die Bedingung beim arithm. Mittel.
- So lange obige Schritte ausführen, bis beide Listen nur noch 2 Werte enthalten, danach wird sortiert. Der Median der 4-elementigen Liste ist dann auch der gesuchte gemeinsame Median beider ganzer Listen.



R1: (((00)*11) | 01)*

R2: (0|1)*000(0|1)*

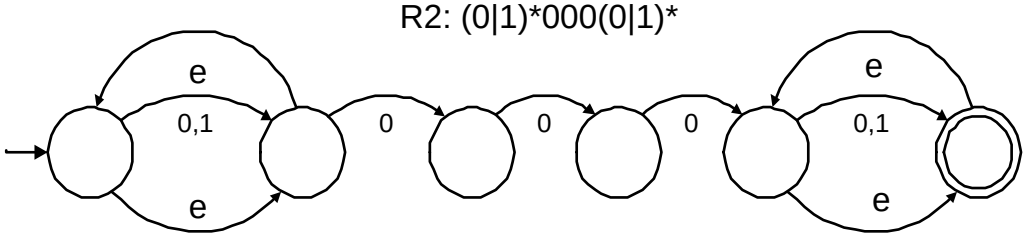


Abbildung 2: dfa's aus Regulären Sprachen