Assignment 07

1) (I changed the numbering to alphabetical)
   a. Normalizing constants are needed to normalize the probabilities to 1. So,

$$\sum_{i=1}^{1000} \frac{i}{Z} = 1 \Leftrightarrow \frac{1}{Z} * \sum i = 1 \Leftrightarrow Z = 500500$$

   So if we take a look at our summing weights, we see that W1= 1/500500, W2= 3/500500, and W1000= 500500/500500 = 1.

   b. Hier suchen wir nach dem Erwartungswert der Vergleiche. Mit anderen Worten, wir suchen nach dem „average case“ der Linearen Suche. Das heißt, in welchem Bereich $]W_{i-1}, W_i]$ liegt unser gezogenes U. Wenn wir jeden dieser Bereiche betrachten können wir erkennen, dass sie mit steigendem i immer größer werden. Anschaulicher:
   Bereich W_1, W_2 bedeckt 1/Z bis 3/Z also insgesamt 2/Z, wohingegen der Bereich W_{10} W_{11} 55/Z bis 66/Z also insgesamt 11/Z abdeckt.
   Unser Erwartungswert berechnet sich wie folgt:

$$\sum_{i=1}^{1000} \frac{i}{Z} * i = 667$$

   So we need to make 667 comparisons.

   c. In the case of backward linear search we have to hit the same „point“ in our sum, but we reach it after 1000-667 iterations, so 333 iterations.

2) (I will explain in german otherwise this might get confusing)
   a. Wir nehmen an, dass ¼ unser Threshold T ist. Alle Balken, die über T liegen gelten als groß, alle die darunter liegen als klein. Diejenigen die genau darauf enden gelten als fertig.

   Wir wollen zeigen, dass, wenn es einen großen Balken Bg gibt, es auch einen kleinen Balken Bk geben muss. Des Weiteren sei die Gesamtheit der Balken n und die Summe der Balken ergibt 1. Hier im Beispiel der Vorlesung ist n = 4, T = ¼ , also 1 = ¼ + ¼ + ¼ + ¼.

   Wenn wir annehmen, dass Bg einen Wert > 1/n hat, geht unsere Gleichung von 1 = n*1/n offensichtlich nicht mehr auf. Wenn wir von n<m ausgehen, bekommen wir die Gleichung

$$1 = (n-1) * \frac{1}{n} + \frac{1}{m}$$

   Also im Fall von n = 4 zum Beispiel:

$$1 = (4-1) * \frac{1}{4} + \frac{1}{2} \Leftrightarrow 1 = \frac{3}{4} + \frac{1}{2}$$

   Das heißt, je größer unser m wird, desto weiter Entfernen wir uns von der 1, also Differenz D von unserem Wert zu 1 wird immer größer. Um unsere Formel korrekt zu halten muss diese Differenz von einer der anderen Balken abgezogen werden. Also muss die Summe der Differenzen aller Balken zu 1/n immer 0 ergeben.

b. Wenn die Aussage “single object“ dahin referenziert, dass z.B. Spalte C vor Ausführung des Algorithmus nur C, also blau enthalten sein darf, erklärt sich das wie folgt. Ich nehme eine Fallunterscheidung vor.

Fall 1: Die Spalte ist fertig (balanciert). Hierher können wir gekommen sein, indem wir eine große Spalte verkleinert haben und die Lücke zwischen der Spalte und T erneut mit dem Überlauf einer anderen Spalte geschlossen haben (> 1 Objekt). Oder, indem eine zu große Spalte auf T reduziert wurde (= 1 Objekt). Balancierte Spalten werden beim „umfüllen“ übergangen und die nächste Spalte, in der weniger als 1/n ist wird befüllt.

Fall 2: Die Spalte ist zu groß. Dieser Fall kann nur eintreten, wenn die Spalte von vorneherein zu groß war (= 1 Objekt)

Fall 3: die Spalte ist zu klein. Dieser Fall tritt ein, wenn die Spalte von vorneherein zu klein war oder „zu viel“ der Spalte entfernt wurde um die folgende Spalte zu füllen. Dieses Verhalten resultiert darin, dass eine Spalte die zu leicht ist, sofort bis T aufgefüllt wird, auch wenn dadurch die vorher zu große Spalte A nun zu klein wird. Spalte A wird später durch den Rest des Überlaufs von T wieder balanciert. Da die Summe immer 1 ergeben muss ergeben sich hieraus keine Probleme.

c. Wie in 2) gezeigt, wird eine leichte Spalte die auf eine schwere Spalte folgt (womit die leichte Spalte per Definition Platz für einen Teil des Inhalts der schweren Spalte hat) immer vollständig mit Inhalt aus der schweren Spalte aufgefüllt. Wird dies wie in unserem Beispiel weitergeführt, ist C balanciert, also muss nichts in G übertragen werden. T wiederrum ist allerdings zu schwer und darum übertragen wir 1/n abzüglich dem was schon in A steht von T nach A, womit A wieder balanciert ist und im Folgenden übergangen wird. T benötigt aber noch mehr Platz. Da Spalte C voll ist, wird der Rest in Spalte G übertragen.

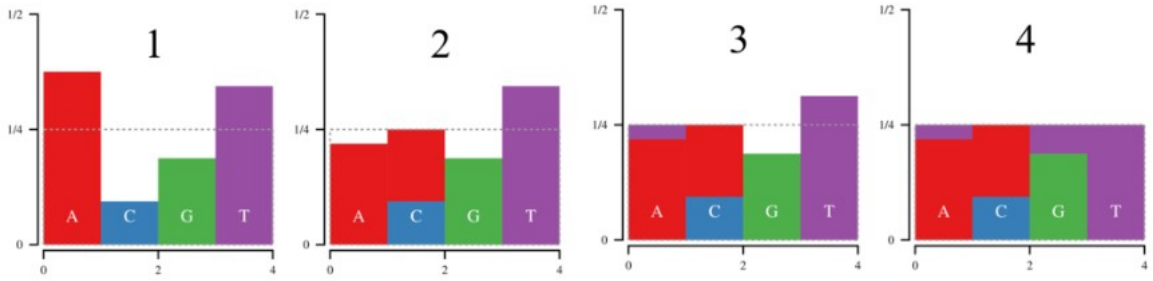

Image: Reed A. Cartwright’s blog “Panda’s Thumb” from Aug 03, 2012

d. Diese Aussage erschließt sich direkt aus 1). Würden wir während des Algorithmus die Gewichte der Objekte verändern, wäre 1 = n * 1/n nur noch möglich, wenn z.B. -x Gewicht verliert und A +x Gewicht gewinnt. Da wir hier aber von Beobachtungen von Nukleinbasen sprechen, würde es jeder Vernunft entbehren diese Beobachtungsgewichte zu verändern, nur weil wir eine Transformation der Daten vornehmen, um sie leichter verwertbar zu machen. Immerhin würde es bedeuten, dass wir von z.B. eine Beobachtung von G wegnehmen und zu C dazutun.