Themen zum praktischen Block der Vorlesung

Advanced Algorithms for Bioinformatics (P4)

In den Klammern steht jeweils die maximale Gruppengröße

Thema 1: QUASAR (3)

- Implementieren Sie den Filteralgorithmus QUASAR.
- Entwickeln Sie ein Konzept zur Messung der Filterleistung.
- Wie ändert sich die Filterleistung von QUASAR in Abhängigkeit von den gewählten Parametern?
- Was bringt der Übergang zu gapped Q-Grams? Was verliert man dadurch, dass man dabei ja gleichzeitig von Edit- zur Hamming-Distanz übergeht?

Thema 2: Chaining (3)

- Entwickeln und Implementieren Sie einen Algorithmus zum Chaining von Alignment-Fragmenten zweier Sequenzen mit beliebigen Gap-Kosten ("Gap" = "Lücke zwischen den Fragmenten"). "Beliebig" heißt z.B., dass der Benutzer dem Chaining-Algorithmus eine Funktion seiner Wahl übergeben kann, welche - gegeben zwei miteinander vereinbare Fragmente - die Kosten für die direkte Verkettung dieser beiden Fragmente zurückgibt.
- Wie kann man den Algorithmus beschleunigen, wenn man Gaps generell kostenlos macht? Orientieren Sie sich z.B. an Gusfields Algorithmus (siehe Dan Gusfield, Algorithms on Strings, Trees, and Sequences).
- Vergleichen Sie die Laufzeiten beider Verfahren miteinander.

Thema 3: Suffix Array Aufbau (3)

- Untersuchen Sie, inwiefern sich Standardsortierverfahren wie z.B. Quicksort (bzw. dem `sort`-Algorithmus aus der STL) dazu eignen, Suffix-Arrays aufzubauen.
- Wie könnte man das Problem dieser Sortierverfahren mit Repeatregionen lösen? Entwickeln und Implementieren Sie Verfahren zur Lösung dieses Problems.
- Belegen Sie den Erfolg ihrer Bemühungen durch Laufzeitvergleiche, auch mit bereits bestehenden Suffix-Array-Aufbauverfahren (z.B. dem Skew 7-Algorithmus aus SeqAn).

Thema 4: Shift-Or mit Wildcards (2)

- Weiten Sie den Shift-Or-Algorithmus dahingehend aus, dass er Muster mit Wildcards (*, + und ?) sucht.
- Konstruieren sie ein worst-case-Szenario für die Suche.
- Messen Sie die Geschwindigkeit ihrer Implementierung an künstlichen und an realen Daten. Wenn man auf Wildcards verzichtet, wie schnell ist Ihr Algorithmus dann im Vergleich zu anderen Suchalgorithmen?

Thema 5: Der Wu-Manber-Trick (2)

- Implementieren Sie den Horspool-Algorithmus und erweitern ihn mit dem "Wu-Manber-Trick", d.h. vergrößern Sie effektiv das Alphabet, indem Sie jeweils q Zeichen zusammenfassen.
- Erstellen Sie eine Testumgebung, mit deren Hilfe Sie experimentell realistische (!) Aussagen darüber gewinnen können, wann welches q optimal ist. (Nicht "realistisch" wäre z.B., das Pattern einfach in einem random string zu suchen.) Untersuchen Sie auch ein worst-case-Szenario.
- Vergleichen Sie Ihre Ergebnisse zum einen mit den theoretischen Überlegungen aus dem Wu-Manber-Paper, und zum anderen mit konkurrierenden String Matching Algorithmen wie BOM oder BNDM (beide finden Sie u.A. in SeqAn).

Thema 6: Nussinov-SCFG (3)

- Entwickeln Sie eine Nussinov-SCFG zur RNA-Faltung.
- Verwenden Sie RFAM-Alignments als Trainingsdaten.
- Vergleichen Sie Ihre Faltungen mit denen von mfold, RNAfold und eventuellen Parallelpraktika.

Thema 7: Boyer-Moore vs. Horspool (2)

- Implementieren sie den Horspool und den Boyer-Moore Algorithmus
- Vergleichen sie die Laufzeit der beiden Algorithmen für verschiedene Eingabedaten.

Thema 8: SWIFT (4)

- Implementieren sie den Swift Algorithmus.
- Nutzen Sie dabei auch die Tricks zur Speicheroptimierung und Laufzeitoptimierung welche im Paper beschrieben sind.
- Vergleichen Sie die Filterleistung und Laufzeit mit der von Quasar (Gruppe 1)