Lehrstuhl für Informatik I
Algorithmen und Komplexität
Universität Würzburg

Wüzburg, den 7. Juli 2021

Dr. Thomas van Dijk
Felix Klesen M.Sc.

Vorlesung Algorithmen für Geographische Informationssysteme SS 2021

Odds & Ends

Abgabe: in 2er Gruppen bis Mittwoch, den 14.7.2021, 10:15 Uhr, in analoger Form oder über die E-Learning-Seite der Lehrveranstaltung. Bitte geben Sie auf Ihrer Ausarbeitung die Namen beider Gruppenteilnehmer an.

1 Fréchet Map Matching

Consider the polyline-and-graph parameter space from the video "Odds & Ends: Fréchet distance map matching." Given a particular maximum Fréchet distance ε, consider the question: does there exists a path in G that has Fréchet distance at most ε to the polyline P? How can you tell from the free space diagram whether this is the case – what kind of path, from where to where, are we looking for? You may ignore the question of how to compute if such a path exists. *(3 Punkte)*

2 Run-length Encoding

The video "Odds & Ends: GloBiMaps" briefly mentions run-length encoding for the black and white pixels along one row of a bitmap image. Note that a such a row is basically a string over the binary alphabet $\{0, 1\}$. In this exercise we will represent the black pixels by 1 (and therefore in the dataset from the video, the string is mostly 0).

One way to represent a string $A = (a_1, a_2, \ldots, a_n) \in \{0, 1\}^n$ is to just write the literal values using n bits; the length of the string can be observed from the number of bits. Run-length encoding is a general idea for compressing strings if we expect long runs of repeated symbols; let a "run" be a maximal interval in the string that contains only a single symbol. For example, a run-length encoding represents `1111100000011` as "$5 \times$ `1`, $6 \times$ `0`, $2 \times$ `1`"; in general, this is a list of tuples (ℓ_i, v_i) representing a run of ℓ_i repeats of the symbol v_i. But this idea is not yet a full specification: how do we represent this as a binary string?

(a) Given that $a_i \in \{0, 1\}$, just writing v_i takes only a single bit. Describe how can we get away with not even storing all v_i individually. *(1 Punkt)*

(a) Describe a way to represent any number $k \in \mathbb{N}$ in such a way that, given a bit string that starts with your representation, you can uniquely determine how many bits to read and which integer is represented. Use at most $2\lceil \log_2 k \rceil$ bits for $k \geq 3$ and at most 4 bits for 1 and 2. *(3 Punkte)*

(a) Using the concepts above, give a run-length encoding scheme to represent arbitrary strings $(a_1, a_2, \ldots, a_n) \in \{0, 1\}^n$ as a hopefully shorter binary string. *(1 Punkt)*

(a) Consider random bit strings of length n with $\Pr[a_i = 1] = p$ for each i independently. How many bits does your representation use in expectation as n goes to infinity? Give either a probabilistic analysis or answer this question experimentally. *(6 Bonuspunkte)*