Walking the Dog Fast in Practice, Algorithm Engineering of the Fréchet Distance

Bram Elderhorst

- What is freceht distance
- Applications
- Running time and overveiw
- Free-Sapce diagrams
- Pruning rule (main novelty)
- Benchmarks
- Conclusion

What is it

It is a number that sais how similar two curves are. You can imagine a dog running on the first curve and a person on the second and the fréchet distance is the minimum length of the leash.

A more formal definition is to have to curves and have a set of traversing functions $\mathcal{T}$. For every possible combination of traversal functions we calculate the maximum distance and then we take the infimum of all the maximums

Applications

- Signature authenticity
- Map-matching

Running time

- Decision problem: Given two curves is the distance smaller than δ
- 1995: $o(mn)$ for the decision problem
- Optimization problem in $o(mn\log(mn))$
- Later it was proven that it is not possible to have an algorithm better than $o(mn(\log mn))$

Free-space diagram

$$F := \{(p,q) \in [1,n] \times [1,m] \mid \|\pi_p - \sigma_q\| \leq \delta\}$$

Reachable free-space diagram

$$R := \{(p,q) \in F \mid \text{There exist a monotone path fom (1,1) to (p,q) in }] \text{ F}\}$$

- A cell is one rectangle based on a segment of π and of σ
- Calculate the free space of a cell in constant time
- The free space in a cell is convex, so the intersection with a cell border is at most one line segment
- Propagate this to the opposite cell border
- Dynamic programming: $O(nm)$

New algorithm

A different algorithm uses a box B consisting of cells between two point of π and σ, boundaries B_i, B_r, B_b, B_t, inputs $B_i^R = B_i \wedge R, B_b^R = B_b \wedge R$, outputs $B_r^R = B_r \wedge R, B_l^R = B_f \wedge R$.
The first pruning rule is very simple. If $B_l^R = B_b^R = \emptyset$ and all inputs are empty, so propagation is impossible then $B_r^R = B_l^R = \emptyset$.

For the next rule we look at free-space boundaries and try to determine if they are reachable. Simple boundaries have at most one intersection with the free-space. f $B_l \wedge F = \emptyset$ the output is empty. If the boundary starts with a free-space part and the left corner is reachable, then the free-space part is reachable. If the free-space part start in the middle, we try if we can propagate from a reachable point at the bottom.

Experiment setup

- Three data sets with more than thousand curves
- Select a random curve π
- Sort the other curves by increasing d_F to π to get $\sigma_i, \ldots, \sigma_n$
- For all $k \in \{1, \ldots, |\log n|\}$: Pick a random curve $\sigma \in \{\sigma_{2^k}, \ldots, \sigma_{2^{k+1}}\}$
- Filters can quickly decide for very small and large δ
- No-instances can terminate earlier instead of traversing the entire free-space diagram
- For similar curves, the diagram is more dificult to compute

Speed-up in running times

- Comparison with the previous best from two years earlier
- Speed ups mostly around 10x
- Improved filters for yes-instances

Effect of individual pruning rules

- Omit each rule one by one to see its effect
- All rules are important, because that can remove a significant part of the free-space

- The last rule can only be used for the edges of a diagram

Conclusion

- New recursive algorithm using techniques from the first published algorithm

Future work

- Tighter running time bound or a certain class of curves
- Incorporate the algorithm in software libraries
- Use this approach for solving other Fréchet distance problems