libfbi: a C++ implementation for fast box intersection and application to sparse mass spectrometry data

Marc Kirchner1,2,3,t,‡, Bo Sun1,2,†, Hanno Steen1,2,3 and Judith A. J. Steen1,4

1Proteomics Center, Children’s Hospital Boston, 2Department of Pathology, Children’s Hospital Boston, 3Department of Pathology, Harvard Medical School and 4Department of Neurobiology, Harvard Medical School and T. M. Kirby Neurobiology Center, Children’s Hospital, Boston, MA 02115, USA

ABSTRACT

Motivation: Algorithms for sparse data require fast search and subset selection capabilities for the determination of point neighborhoods. A natural data representation for such cases are space partitioning data structures. However, the associated range queries assume noise-free observations and cannot take into account observation-specific uncertainty estimates that are present in e.g. modern mass spectrometry data. In order to accommodate the inhomogeneous noise characteristics of sparse real-world datasets, point queries need to be reformulated in terms of box intersection queries, where box sizes correspond to uncertainty regions for each observation.

Results: This contribution introduces libfbi, a standard C++, header-only template implementation for fast box intersection in an arbitrary number of dimensions, with arbitrary data types in each dimension. The implementation is applied to a data aggregation task on state-of-the-art liquid chromatography/mass spectrometry data, where it shows excellent run time properties.

Availability: The library is available under an MIT license and can be downloaded from http://software.steenlab.org/libfbi.

Contact: marc.kirchner@childrens.harvard.edu

Supplementary information: Supplementary data are available at Bioinformatics online.

Received on October 29, 2010; revised on February 7, 2011; accepted on February 10, 2011

1 INTRODUCTION

Modern high-resolution liquid chromatography/mass spectrometry (LC/MS) datasets are a prominent example of sparse data stored in a standard preprocessing steps (Cox and Mann, 2008; Khan et al., 2009). Each of these steps requires fast neighborhood evaluation for hundreds of thousands of single measurements. A straightforward approach to this problem is the use of space partitioning data structures such as BSP trees, Octrees, R-trees or k-d trees and to evaluate neighborhood relations on the fly using data-dependent range queries (Khan et al., 2009). However, real-world measurements are often subject to varying magnitudes of noise. Consequently, aggregation methods will deliver varying uncertainty estimates for e.g. calculated centroids and/or XICs. This context gives rise to a major conceptual reservation against simple range query approaches: although a range query is a natural representation for the detection of measurements that fall into the uncertainty bound of the point from which the query is issued, it cannot take into account the uncertainty of the points that should be returned by the query. Consequently, the range query assumes that the queried observations are noise free, and is bound to miss observations where points fall outside the query range but query and target uncertainty ranges overlap (Fig. 1).

The key to overcoming this limitation is to reformulate the classical range query in terms of a (potentially multi-dimensional) box intersection problem with point-specific box sizes. Observations and range queries both define a set of axis-parallel boxes, and the goal is to determine all box intersections between the sets.

This contribution introduces an implementation of a fast box intersection procedure termed libfbi and illustrates its application to state-of-the-art MS data.

2 METHOD AND IMPLEMENTATION

A box $X$ is defined as the Cartesian product of half-open intervals $[L_1, U_1) \times [L_2, U_2) \times \cdots \times [L_D, U_D)$, where the $L_i$ and $U_i$, $d \in \{1, \ldots, D\}$ are the lower and upper interval bounds in dimension $d$. The box intersection problem takes two sets of boxes $A = \{ A_i \}, i \in \{1, \ldots, M\}$ and $B = \{ B_j \}, j \in \{1, \ldots, N\}$ and determines an adjacency list holding the set of index pairs $(i, j), k \in \{1, \ldots, K\}$ of intersecting boxes. libfbi provides an implementation of the fast box intersection algorithm proposed in Zomorodian and Edelsbrunner (2000). The approach follows a divide and conquer scheme, iteratively separating the sets of boxes in every dimension based on implicitly constructed segment trees. Reaching the last dimension or a threshold $\theta$ in the number of boxes, the algorithm switches to a brute-force scanning procedure to avoid the comparatively large hidden constants of the segment tree. Such a hybrid approach yields $O(M + N) + N \log \theta$ space complexity, $O(M + N) + \log \theta$ space complexity for the partitioning and $O((N + M) \log(\theta + \delta))$ for the scan, with $\delta = \{1, \ldots, \theta\}$.
libfbi is a library for the computation of box intersections in an arbitrary number of dimensions. Application scenarios include LC/MS feature extraction, feature correspondence estimation, bounding volume determination and collision detection in geometric and image processing problems and more. libfbi is available from http://software.steenlab.org/libfbi.

Acknowledgements

The authors thank Dominic Winter for data acquisition.

Funding: M.K. gratefully acknowledges financial support by the Alexander von Humboldt-Foundation.

Conflict of Interest: none declared.

References

