Dimension-Independent Simplification and Multi-Resolution Representation of Morse Complexes

Lidija Čomić†
University of Novi Sad

Leila De Floriani‡
University of Genova

Federico Iuricich‡
University of Genova

Abstract—We have defined and implemented atomic and dimension-independent simplification operators on a graph-based representation of Morse complexes, and we have defined and implemented a multi-resolution model for Morse complexes built through such simplification operators.

1 INTRODUCTION

Morse theory [9] offers a natural and intuitive way of analyzing the structure of a scalar field \( f \) and of compactly representing it through decompositions of the domain \( M \) of \( f \) into meaningful regions associated with its critical points, called Morse and Morse-Smale complexes. A descending (ascending, resp.) Morse complex \( \Gamma_d \) (\( \Gamma_a \)) partitions manifold \( M \) into cells defined by integral lines converging to (originating at, resp.) critical points of \( M \). Morse theory [9] offers a natural and intuitive way of analyzing the structure of a scalar field \( f \) and of compactly representing it through decompositions of the domain \( M \) of \( f \) into meaningful regions associated with its critical points, called Morse and Morse-Smale complexes. A descending (ascending, resp.) Morse complex \( \Gamma_d \) (\( \Gamma_a \)) partitions manifold \( M \) into cells defined by integral lines converging to (originating at, resp.) critical points of \( M \). For a survey of algorithms for computing Morse and Morse-Smale complexes, see [1, 4]. In our work, we use the algorithm in [10] which is based on discrete Morse theory. The algorithm in [10] is the basis for the algorithms in discrete Morse theory.

In [2], we have defined atomic and dimension-independent simplification and refinement operators on Morse complexes, which form a minimally complete set of operators for creating and updating Morse and Morse-Smale complexes. The \( \text{remove} \) simplification operator collapses two saddles of consecutive index that are connected through an integral line, and such that one of them is connected to at most one saddle of the same index as the other one. Let \( \text{remove} \) apply when \( f \) has two different saddles \( i \) and \( i+1 \) of index \( i \) and \( i+1 \), respectively. The effect of \( \text{remove} \) is illustrated in Figure 1: \( i+1 \)-cell \( q \) is deleted, and in \( i+1 \)-cell \( p \) is deleted. All arcs incident in \( q \) and \( p \) are deleted, and \( i+1 \)-node becomes \( i \)-node.

2 SIMPLIFICATION OPERATORS ON MORSE COMPLEXES

In [2], we have defined atomic and dimension-independent simplification and refinement operators on Morse complexes, which form a minimally complete set of operators for creating and updating Morse and Morse-Smale complexes. The \( \text{remove} \) simplification operator collapses two saddles of consecutive index that are connected through a unique integral line, and such that one of them is connected to at most one saddle of the same index as the other one. Let \( \text{remove} \) apply when \( f \) has two different saddles \( i \) and \( i+1 \) of index \( i \) and \( i+1 \), respectively. The effect of \( \text{remove} \) is illustrated in Figure 1: \( i+1 \)-cell \( q \) is deleted, and in \( i+1 \)-cell \( p \) is deleted. All arcs incident in \( q \) and \( p \) are deleted, and \( i+1 \)-node becomes \( i \)-node.

We encode the topology of the descending and ascending Morse complexes \( \Gamma_d \) and \( \Gamma_a \), respectively, through the Morse Incidence Graph (MIG) [3] (see Figure 1 for a 2D example). An MIG is a dimension-independent multigraph \( G = (N, A) \) such that (i) there is a one-to-one correspondence between the nodes in \( N \) and the \( i \)-cells of \( \Gamma_d \) (and thus the \((n-i)\)-cells of \( \Gamma_a \) (called i-nodes)), and (ii) there are \( k \) arcs joining an \( i \)-node \( p \) with an \((i+1)\)-node \( q \) if and only if \( i \)-cell \( p \) appears \( k \) times on the boundary of \((i+1)\)-cell \( q \) in \( \Gamma_d \).

A \( \text{remove} \) operator on an MIG \( G = (N, A) \) eliminates the two nodes corresponding to the two critical points \( p \) and \( q \). A \( \text{remove} \) is feasible on \( G \) if there is a unique arc \((p, q)\) in \( A \), and there is a unique \((i+1)\)-node \( p' \) different from \((i+1)\)-node \( p \) and connected to \( i \)-node \( q \). The effect of the \( \text{remove} \) operator on the MIG is completely local, since it deletes the arcs incident in \( p \) and the \( (i+2) \)-nodes in the MIG, and it modifies the arcs incident in \( p \) and in \( i \)-nodes in the MIG which become incident in \( p' \).

The cancellation \( \text{cancellation} \) eliminates an \( i \)-saddle \( q \) and an \((i+1)\)-saddle \( p \) that are connected through a unique integral line. There is no limit on the number of saddles connected to \( q \) and of the same index as the other saddle (i.e., \( p' \) is not unique). Let \( T \) be the set of \((i+1)\)-nodes connected to \( q \). The effect of the \( \text{cancellation} \) of \( p \) and \( q \) on the MIG consists of deleting nodes \( p \) and \( q \) and all arcs incident in either of them, and creating arcs connecting each \( i \)-node previously connected by an arc to \((i+1)\)-node \( p \) with each node in \( T \).

Thus, a \( \text{cancellation} \) deletes two nodes from the MIG, but, unlike \( \text{remove} \), it may increase the number of arcs. Some strategies have been proposed to bound the number of arcs introduced by a \( \text{cancellation} \) [6].

\( \text{Figure 1.} \) The descending 2D Morse complex \( \Gamma_d \) for function \( f(x, y) = \cos(x) \cos(y) \) (a) and the corresponding MIG (b). An example of \( \text{remove} \) on \( \Gamma_d \) (c), and on the MIG (d). After \( \text{remove} \) on \( \Gamma_d \), \( \Gamma_d \) consists of deleting two nodes from \( \Gamma_d \) (called i-nodes), and \( \Gamma_a \) remains. The MIG is illustrated in Figure 1: 1-cell \( q \) is deleted, and 2-cell \( p \) is merged into 2-cell \( p' \). After \( \text{remove} \) on \( \Gamma_d \), \( \Gamma_d \) becomes completely dual.

\( \text{Figure 2.} \) The field behavior for the xMaxTree data set (a). The MIG at full resolution (b), the MIG after 10K (c) and after 16K simplifications (d).
3 Experimental Results

We have developed a simplification algorithm for Morse complexes including operators on the dimension-independent MIG based on the remove operators. We associate a value $P(p,q)$ to any feasible remove operator, deleting two critical points $p$ and $q$, as the absolute difference in function values at $p$ and $q$, and we perform simplifications in order of increasing $P(p,q)$ values. We have performed experiments on the simplification of Morse complexes by using six data sets describing 2D scalar fields, and eight data sets describing 3D scalar fields on a 3.2GHz processor with 2 GBytes of memory. We have used different threshold values on $P(p,q)$: 1% of the maximum $P(p,q)$ value for light noise removal, 10% for stronger noise removal, and 20% or greater for consistently reducing the complexity of the MIG. The storage cost of the simplified MIG using these three different thresholds is equal to 95%, 65% and 35% of the cost of the MIG at full resolution.

In Table 1, we show the results obtained by comparing the remove operator with the cancellation operator. For each data set, we show in the first row the number of nodes and arcs in the full resolution MIG. In the second and third rows, we show the statistics related to cancellation and remove operators, respectively: the number of simplifications applied, the number of nodes and arcs in the simplified MIG, the cost of the data structure encoding the MIG (in Mb), and the time (in sec) needed to perform the simplifications. The number of nodes and arcs in the graph simplified with cancellation always exceeds the number of arcs in the graph simplified with the same number of remove. Such behavior influences the efficiency of the whole algorithm, doubling the time needed to manage and enqueue a larger number of arcs (and thus, a greater number of possible simplifications) for large data sets. The cost of the MIG is reduced by 10% ~ 20% by using remove instead of cancellation.

We have applied our simplification algorithm on a 3D xMasTree data set (see Figure 2 (a)), which represents a tomography of a Christian tree. The MIGs at full-resolution, and after 10K and 16K simplifications are shown in Figures 2 (b), (c) and (d), respectively.

4 The Multi-Resolution Morse Incidence Graph

We have defined the inverse insert refinement operator which undoes the effect of the remove simplification operator, and we have designed and implemented a multi-resolution model for the MIG representation of the Morse complexes, that we call the Multi-Resolution Morse Incidence Graph (MMIG) [3]. An MMIG is built from the full-resolution MIG by applying a sequence of remove operators. It consists of the base MIG $G_B$, representing the Morse complexes at the coarsest resolution, the set of insert refinement modifications, inverse to the remove applied in the simplification phase, and a dependency relation between refinement modifications, which has been shown to be a partial order relation. From an MMIG, it is possible to extract representations of the Morse complexes at any intermediate (uniform or variable) resolution by considering a subset of refinements that is closed with respect to the partial order.

We have implemented two data structures for storing the MMIG, that we call the implicit and the explicit data structures, respectively. In the implicit data structure, the direct dependencies of a modification $\mu$ are encoded in two arrays, ancestors and descendants, and a vector $pq$-ancestors of two bit elements of the same size as ancestors. $pq$-ancestors[$j$] encodes whether a modification $\mu$ depends on node $p$, on node $q$ or on both in ancestors[$j$]. Thus, the sets of nodes that will be connected to $q$ or $p$ are implicitly encoded. In the explicit data structure, three lists are created for each modification $\mu$, storing the actual pointers to the above mentioned sets of nodes. The implicit data structure occupies 3 to 10 times less space than the explicit one. The extraction algorithm is from 33% to 50% faster when using the explicit data structure.

Acknowledgments

This work has been partially supported by the Italian Ministry of Education and Research under the PRIN 2009 program, and by the National Science Foundation under grant number IIS-1116747.

References


