

Abstract Matrices and Constraints

Alan P. Sexton and Volker Sorge

School of Computer Science, University of Birmingham

A.P.Sexton|V.Sorge@cs.bham.ac.uk, www.cs.bham.ac.uk/~aps|~vxs

Abstract Matrices

In every day mathematical practice, it is routine to reason and compute with matrices that are of indefinite dimension and contain abbreviations and underspecified parts such as ellipses. While the ability to compute with abstract matrices is crucial when reasoning about propositions involving general matrices, there is very limited automated support for this. We introduce a new data type *Abstract Matrix*, implemented in the Computer Algebra System Maple [1], for which the dimensions of the matrices may be unknown and the matrix may include ellipses and fill terms. The algorithm enables the transformation of textbook-style matrices into lambda calculus expressions [3] as well as their use as templates for a class of concrete matrices that could be instantiated from the abstract matrix [2]. For example, the matrix A below can be transformed into the lambda term given in the middle and serves as a template for concrete instantiations of dimensions $1 \leq n \leq 3$:

$$A = \begin{pmatrix} a_1 & b & \cdots & b \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & b \\ 0 & \cdots & 0 & a_n \end{pmatrix} \quad \lambda i \lambda j . \begin{cases} a(i) & \text{if } i = j \\ b & \text{if } i < j \\ 0 & \text{else} \end{cases} \quad (a_1), \quad \begin{pmatrix} a_1 & b \\ 0 & a_2 \end{pmatrix}, \quad \begin{pmatrix} a_1 & b & b \\ 0 & a_2 & b \\ 0 & 0 & a_3 \end{pmatrix}$$

The major problems we are faced with when dealing with abstract matrices are not only to identify the occurring ellipses and their composition in terms of the elements they represent, but also to interpret them in relation to each other in order to find the different regions a matrix is composed of and to determine the relative size of the regions with respect to each other. Thus the relative size of a region is not only determined by the dimension of the matrix and the relative sizes of neighbouring regions, but also by the type and structure of the elements it is composed of. We do this by constructing a set of integer constraints when parsing an abstract matrix, which can then be used to construct a consistent set of instantiations for the matrix parameters.

Parsing Abstract Matrices

Parsing abstract matrices consists of three phases: (1) extracting ellipsis length constraints, (2) identifying regions in the matrix, and (3) extracting information on the content of each region.

Ellipsis Length Constraints: Ellipses in a matrix represent an expandable sequence of entries. However they cannot grow or shrink entirely independently of each other. We therefore represent lengths of ellipses as integer variables and capture their mutual relationships as a set of additive integer constraint equations, that we call *structural constraints*. To find these constraints we rely on some basic equivalences in ellipsis lengths: we construct a graph of vertical, horizontal, diagonal and anti-diagonal edges between matrix cell corners based on ellipsis occurrences and individual terms. The edges in this graph are labelled with weights constructed from the ellipsis length variables for an edge corresponding to an ellipsis, and the value 1, corresponding to a simple term. Edges have a separate horizontal and vertical weight. Then we can construct weights, as integer expressions, for any path in the graph by summing the weights of all edges along the path. Now we can construct the ellipsis length constraints by identifying the weight expressions for every pair of sub-paths between the same vertices in the graph. The set of constraints found can be simplified using standard simultaneous equation reduction techniques, but can only be fully solved if the matrix is fully concrete.

Identifying Regions: A region is essentially a minimal ellipsis cycle in the input matrix. The edges of a region may include concrete terms as well as ellipses. We define a region as a closed polyline of *generalised positions*, where a generalised position is simply a pair of expressions over integers, ellipsis length and dimension variables that defines the row and column position index of the cell. Generalised positions can be computed by analysing the graph produced as part of the ellipsis length analysis. Once we have obtained the required generalised positions, we use a path-following algorithm for boundary detection on the input matrix to find the minimal ellipsis cycles.

Extracting Meaning from Terms: In order to determine the correct content of a region we extract information from the concrete elements on its boundary. We construct a *generalised term*, that can be unified with each concrete term on the region boundary employing a general unification algorithm for first order logic. But, contrary to regular unification, our algorithm produces a minimal disagreement set for the terms forming the boundary of region.

Each region must have associated with it a single generalised term. Furthermore, it must be possible to instantiate, or *concretise*, a region of variable position and extent in an abstract matrix to a region of fully determined position and extent for any consistent instantiations of the ellipsis lengths. This means that, for all positions within a region, we must be able to interpolate values for the unification variables of the generalised term based on the instantiation values of the boundary terms and the positions of those terms. We use a plane fitting algorithm to interpolate the values independently for each unification variable of the generalised term. More precisely, we treat each point of the region shape as a point in 3-D space, where the row and column expressions correspond to the x and y dimensions and the difference sub-term corresponds to the z dimension. Given the interpolation function we can already construct a lambda expression that fully determines the given abstract matrix.

When concretising an abstract matrix, all three coordinates will be integer constants. Solving this problem adds further linear constraints — so-called *sub-term constraints* since they mix ellipsis length variables and sub-terms of the boundary terms — to the system of linear ellipsis length constraints.

Computations on Abstract Matrices

The only operation currently implemented on abstract matrices is their concretisation to regular fully specified matrices. Concretisation is achieved through a stepwise refinement of the constraint set of the abstract matrix by successively fixing the undefined values. Once this results in a consistent and fully solved constraint store, the resulting concrete matrix can be used in further computations.

The next step in our work will involve extending the data type of abstract matrices by arithmetic methods, in particular addition and multiplication. These operations will be implemented as a suitable combination of the respective constraint stores of two matrices, thus making the combination of constraints conditional with respect to the length of ellipses involved. For example, when adding the two row vectors $A = (a_1 \dots a_l, b_1 \dots b_{n-l})$ and $C = (c_1 \dots c_k, d_1 \dots d_{n-k})$ the resulting ellipsis constraints depend on the relative value of k and l .

Adding abstract matrices leads to a resultant abstract matrix whose regions are partitions of the regions of the original matrices. In some cases, the optimisation of merging neighbouring regions into a single region may be possible. When multiplying abstract matrices, the partitioning of regions is more severe and the complex constraints that result leave open problems in the area of identifying and merging compatible neighbouring regions.

References

- [1] A. Heck. *Maple Manuals*. Springer, 3rd edition, 2003.
- [2] A. Sexton and V.Sorge. Semantic Analysis of Matrix Structures. In Proc. of International Conference in Document Analysis and Recognition (ICDAR). IEEE Computer Society Press, 2005.
- [3] A. Sexton and V.Sorge. Processing Textbook-style Matrices. Submitted to International Conference on Mathematical Knowledge Management 2005.