Solving Systems of Difference Constraints Incrementally

G. Ramalingam, J. Song, L. Joskowicz, and R. E. Miller

Abstract. Difference constraints systems consisting of inequalities of the form $x_i - x_j \leq b_{i,j}$ occur in many applications, most notably those involving temporal reasoning. Often, it is necessary to maintain a solution to such a system as constraints are added, modified, and deleted. Existing algorithms handle modifications by solving the resulting system anew each time, which is inefficient. The best known algorithm to determine if a system of difference constraints is feasible (i.e., if it has a solution) and to compute a solution runs in $\Theta(mn)$ time, where $n$ is the number of variables and $m$ is the number of constraints.

This paper presents a new efficient incremental algorithm for maintaining a solution to a system of difference constraints. As constraints are added, modified, or deleted, the algorithm determines if the new system is feasible and updates its solution. When the system becomes infeasible, the algorithm continues to process changes until it becomes feasible again, at which point a feasible solution will be produced. The algorithm processes the addition of a constraint in time $O(m + n \log n)$ and the removal of a constraint in constant time when the original system is feasible. More precisely, additions are processed in time $O(\|\Delta\| + |\Delta| \log |\Delta|)$, where $|\Delta|$ is the number of variables whose values are changed to compute the new feasible solution, and $\|\Delta\|$ is the number of constraints involving the variables whose values are changed. When the original system is infeasible, the algorithm processes any change in $O(m + n \log n)$ amortized time. The new algorithm can also be used to check for the existence of negative cycles in dynamic graphs.

Key Words. Difference constraints, Incremental algorithm, Linear constraints, Shortest-path problem, Dynamic negative cycle.

1. Introduction. A system of difference constraints is a set of inequalities of the form $x_i - x_j \leq b_{i,j}$. Such a system is said to be feasible if there exists a solution to the system of inequalities. Systems of difference constraints occur in many applications involving temporal reasoning. In AI, Dechter et al. [5], [16] formulate a unifying temporal reasoning framework, called a temporal constraint network, based on difference constraints. Many real-time programming languages [12], [17], [15], [27] provide constructs that allow the specification of temporal relations as difference constraints. Multimedia applications also use difference constraints to specify temporal behavior [14], [3], [24]. For example, the play duration and relative ordering of multimedia objects such as audio or video segments are expressed as difference constraints relating the segment’s time duration and minimum and maximum bounds.

Often it is necessary to maintain a solution to a difference constraints system as constraints are added, modified, and deleted. For example, interactive multimedia systems let users create difference constraint systems by adding and deleting constraints. As the system evolves, it is necessary to check for the feasibility of the new system of
constraints and report inconsistencies, if any, and to provide a solution, if the system is feasible. Existing algorithms handle modifications by solving the resulting system anew each time, which is inefficient. The best known algorithm to determine if a system of difference constraints is feasible and to compute a solution runs in \( \Theta(mn) \) time, where \( n \) is the number of variables and \( m \) is the number of constraints.

This paper presents a new efficient incremental algorithm for testing the feasibility of a system of difference constraints and maintaining a solution to it. As constraints are added, modified, or deleted, the algorithm determines if the new system is feasible and updates its solution, providing the user with immediate feedback after each operation. If the system becomes infeasible, the algorithm will maintain the information and continue to process changes until it becomes feasible again, at which point a feasible solution will be produced. The algorithm processes the addition of a constraint in time \( O(m+n \log n) \) and the removal of a constraint in constant time when the original system is feasible. More precisely, additions are processed in time \( O(\|\Delta\| + |\Delta| \log |\Delta|) \), where \(|\Delta|\) is the number of variables whose values are changed to compute the new feasible solution, and \( \|\Delta\| \) is the number of constraints involving the variables whose values are changed. When the original system is infeasible, the algorithm processes any change in \( O(m+n \log n) \) amortized time.

The rest of the paper is organized as follows. In Section 2 we briefly discuss previous work in this area. In Section 3 we present background material concerning systems of difference constraints. In Section 4 we define the problem addressed in this paper. In Section 5 we present a simple algorithm for the problem, which will help motivate an improved algorithm presented in Section 6. We assume in Sections 5 and 6 that the original system is feasible. In Section 7 we show how to handle infeasible systems. In Section 8 we compare our work with related work. In Section 9 we discuss possible future work.

2. Previous Work. Various types of constraint systems have been widely studied. Pratt [19] showed that a system of difference constraints can be represented by a weighted directed graph such that a system is feasible iff there exists no negative weight cycle in the graph. He also gave an \( O(n^3) \) algorithm for solving systems of difference constraints, which uses the shortest-path algorithm. Shortest paths can also be computed in \( O(mn) \) time.

Shostak [23] generalized Pratt’s ideas to systems of two-variable linear constraints (constraints of the form \( ax + by \leq c \), where \( a, b, \) and \( c \) are real constants and \( x \) and \( y \) are variables). He showed how such systems could be represented by a constraint graph such that a system is feasible iff the graph contains no cycle of a special kind. His algorithms for testing for feasibility, however, have an exponential worst-case behavior. Aspvall and Shiloach [2] improved Shostak’s algorithm into a polynomial time algorithm. The most efficient algorithm currently known for this problem is an \( O(mn^2 \log m) \) algorithm due to Hochbaum and Naor [11]. Pape [18] shows how one can deal with difference constraints over totally ordered Abelian Groups. Jaffar et al. [13] considered the problem of two variable constraints of the form \( ax + by \leq c \), where \( a, b \in \{-1, 0, 1\} \). They present an algorithm for computing a feasible solution to a system of two variable constraints that processes the constraints one by one. The algorithm takes \( O(n^2) \) time per constraint,