Appears in the IEEE Transactions on Parallel and Distributed Systems, June 95

Integer Programming for Array Subscript Analysis

Jaspal Subhlok

School of Computer Science, Carnegie Mellon University, Pittsburgh PA 15213

Ken Kennedy

Department of Computer Science, Rice University, Houston, TX 77251

Abstract: We present a new method to determine whether a convex region contains any integer points. The method is designed for array subscript analysis in parallel programs. The general problem is whether a system of linear equalities and inequalities has an integer solution. A set of known techniques is used to transform the problem to that of finding whether a convex region contains any integer points. The main result of the paper is a set of new search procedures that identify an integer solution in a convex region, or prove that no integer solutions exist. They are based on the geometrical properties of convex regions that are not empty, but also do not contain any integer points. The results contribute to exact and efficient dependence and synchronization analysis of parallel programs.

Index terms – Subscript analysis, dependence testing, integer programming, parallelizing compilers, parallel program analysis, synchronization analysis

This research was sponsored in part by the Advanced Research Projects Agency/CSTO monitored by SPAWAR under contract N00039-93-C-0152, and in part by the National Science Foundation under Grant No. CCR-912008. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the U.S. government.

1 Introduction

Several mathematical problems that arise in the development of parallelizing compilers can be transformed to integer programming problems. Determining whether a data dependence exists between two array references can be transformed to the problem of determining whether an integer solution to a set of equalities and inequalities exists [1, 13]. Similarly, in an event variable synchronization model, determining whether the synchronization present in a program is sufficient to protect a dependence can be transformed to the problem of determining whether a system of linear equations has a non-negative integer solution [3]. Both these problems are instances of integer programming, which is known to be NP-complete.

Solving Diophantine equations and Fourier-Motzkin elimination are well known techniques that have been used by researchers to solve integer programming problems in parallelizing compilers. But they are not sufficient to solve the problem exactly; in the worst case it is necessary to search a convex region for integer points, which is the subject of this paper. We also illustrate how the various methods are combined to form an efficient solution procedure. For details on the context in which the results of this paper are applicable, the reader is referred to [9, 10].

Our approach is oriented towards integer programming problems that are solved in a parallelizing compiler, where the number of equations and variables is tied to the loop nesting depth and the number of subscripts in array references, both of them typically small integers. The small size of problem instances makes it possible to develop an efficient solution procedure even though the general problem is NP-complete. The main results in the paper are for problems in two dimensions. Since a two dimensional problem is obtained from the simplest problem that is not directly solved by other methods, it is likely to occur most often, and we believe that an efficient method to solve it is an important contribution. Generalization of the results is discussed briefly.

2 Background

This paper presents a method to determine if a given convex region bounded by a set of inequalities contains any integer points. The complete problem is to determine if a system of equalities and inequalities has an integer solution. We outline the overall solution procedure.

First solve the equation system for all integer solutions. This is often referred to as solving linear Diophantine equations. If such a solution is unique or does not exist, the problem is immediately solved. If there are multiple solutions, we obtain parametric equations that describe the solution space, and add the inequality constraints to obtain a convex region. We check if the convex region is empty, in which case no integer or real solutions exist. If it is not empty, the methods in this paper are used to determine if it contains any integer points.

2.1 Solving linear Diophantine equations

The theory of solving linear Diophantine equations has been discussed in many texts addressing number theory and integer programming [7, 8]. The problem can be stated as follows:

Find all integer solutions of

$$[A]_{m\times n}.[x]_{n\times 1} = [b]_{m\times 1}$$

The Generalized GCD Test [2] and conversion to the Smith normal form [11, 12] are some practical approaches to solving such an equality system. The result is, a unique solution, or infinite solutions expressed as parametric equations, or proof that no solutions exist. If parametric equations are obtained, they are combined with other inequality constraints to yield a convex region, and the overall system has an integer solution if and only if this convex region has at least one integer point. For a system of m independent equations in n variables, if $m \geq n$, there is a unique integer solution or no solutions exist. If $m < n$, there are infinite solutions parameterized by $n - r$ new variables, and the convex region describing the solution space is in $n - r$ dimensions.

2.2 Fourier-Motzkin elimination

The number of variables in a system of inequalities can be successively reduced by using the Fourier-Motzkin elimination method. A variable is eliminated by creating a new inequality for each pair of inequalities in which that variable has a different sign. We illustrate the method briefly with an example. A complete treatment can be found in [4]. Consider the following set of inequalities:

$$\begin{aligned} 0x - y + 6 &\geq 0 && (1)\\ x + 2y - 6 &\geq 0 && (2)\\ -2x - 3y + 7 &\geq 0 && (3)\\ -4x - 5y + 40 &\geq 0 && (4) \end{aligned}$$

We show how x can be eliminated from this system of inequalities[1]. x has a positive sign in (2) and a negative sign in (3) and (4). By combining inequality pairs (2)-(3) and (2)-(4) to eliminate x, we obtain the following system:

$$\begin{aligned} -y + 6 &\geq 0 && && (5)\\ y - 5 &\geq 0 && \text{from } (2) \textit{ and } (3) && (6)\\ 3y + 16 &\geq 0 && \text{from } (2) \textit{ and } (4) && (7) \end{aligned}$$

This simplifies to:

$$5 \leq y \leq 6$$

Thus a solution set to the inequalities does exist, and bounds on y are obtained.

Solving a system of inequalities by the Fourier-Motzkin elimination techniques can potentially increase the number of inequalities from n to $(n/2)^2$ in every stage, so it is an exponential method. However, there is evidence that it is a practical and useful method, particularly for small problems [4].

3 Finding bounding rectangles for convex regions

Every finite convex region has a unique bounding n-rectangle. An *n-rectangle* is a closed convex region defined by a set of hyperplanes parallel to the axis hyperplanes (that is, defined by $x = constant$ for some coordinate axis x). An n-rectangle R bounds a convex region C, if it is the smallest n-rectangle that contains all points in C. Finding the bounding rectangle of a convex region is an important step in our method, and we discuss how it can be done efficiently.

The bounding rectangle of a convex region is defined by the range of the convex region in each dimension, which can be determined by the Fourier-Motzkin elimination. By repeating the process for each dimension, bounds in all dimensions can be obtained. For a convex region in n dimensions (i.e. n free variables), this would require n^2 steps, where a step is defined as the elimination of one variable from a system of inequalities. This is an expensive process and we present two techniques to reduce the cost.

Reusing partially solved systems:

It is clear that the equation system obtained by eliminating one variable, say x, can be used as a common step for finding the range for all other variables. This observation can be used to build a schedule where $O(n * log(n))$ steps are required for finding the bounds for all n variables, instead of $O(n^2)$ steps[2].

Finding bounds by back substitution:

The bounds in one dimension, say y, can often be obtained by back substituting bounds in another dimension, say x. If we back substitute the bounds of x, and find that the *same* inequality corresponds to the largest value of y for higher and lower bounds of x, then the larger value of y is indeed the upper bound y. This observation generalizes to n dimensions, and is useful in reducing the elimination steps for finding a bounding rectangle.

[1] Heuristics for choosing a variable for elimination such that the total amount of work is minimized, are discussed in [4].

[2] We should not conclude from this that the total work is reduced from n^2 to $n * log(n)$ since different steps take different amount of time.

4 Identifying integer solutions

The methods in section 2 either solve a given problem, or identify a convex region which has integer points if and only if the original problem had an integer solution. We present efficient solution procedures for convex regions in two dimensions. Because of the small size of the problem instances, previous processing steps would generally solve the problem or yield a convex region in one or two dimensions. Thus, an efficient solution method for two dimensional convex regions extends the class of problems that can be solved efficiently. (Solution for one dimensional convex regions is, of course, trivial).

4.1 Recursive search procedure

We describe a recursive search procedure for convex regions in two dimensions that would identify an integer solution or prove that none exists. The procedure is based on the observation that if a convex region contains any integer points, there is likely to be one close to the center of the region.

A system of inequalities in two variables describes a convex region in two dimensions. Using the Fourier-Motzkin elimination, we determine the bounding rectangle for the convex region and the *centroid* of the bounding rectangle, which is the point with co-ordinates equal to the midpoint of the range of the bounding rectangle in each dimension. contains any integer points.

Theorem 1 *If a finite convex region in two dimensions is bounded by a rectangle, then the centroid of the rectangle is always a part of the convex region.*

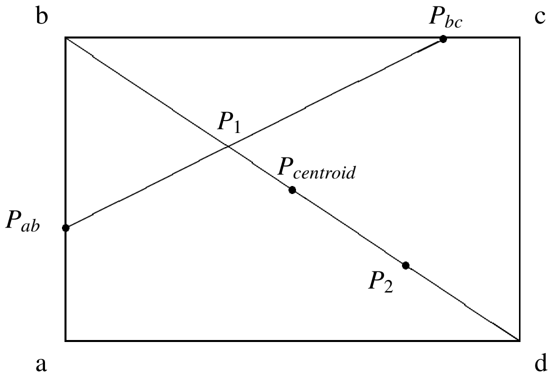


Figure 1: Illustration for Theorem 1

Proof:

Let C be a convex region in two dimensions, with the bounding rectangle R. Let $abcd$ be the vertices of the rectangle R as shown in Figure 1. Let $P_{centroid}$ be the centroid of R, which is also the point of intersection of the diagonals ac and bd. Since R bounds C tightly, each of the sides ab, bc, cd and da must contain at least one point that belongs to C. Let P_{ab} and P_{bc} be points on the sides ab and bc respectively, that belong to C. Since C is convex, the line segment $P_{ab}P_{bc}$ is contained in C. Also, the line segment $P_{ab}P_{bc}$ is inside the triangle abc and must intersect the diagonal bd at a point between b and $P_{centroid}$ (which is on the diagonal ac) , say P_1. Thus there is one point on the diagonal bd between b and $P_{centroid}$ that is also in C. Similarly we can show that there is one point on the diagonal bd between d and $P_{centroid}$, say P_2, that is also in C. Therefore, since C is convex, $P_{centroid}$ is part of C.
End of Proof.

This theorem suggests that the neighborhood of the centroid is a good place to look for possible integer solutions. Furthermore, if none of the integer point neighbors of the centroid are in the convex region, the convex region can only be of a special shape, and we present a set of results to establish that.

Lemma 2 *Let R (vertices $r_1r_2r_3r_4$) and R' (vertices $r'_1r'_2r'_3r'_4$) be rectangles with parallel sides such that R' is enclosed inside R, as shown in Figure 2. Let p be a point inside R'. A set of line segments that connect p to one point on each side of R, must intersect at least two sides of R'.*

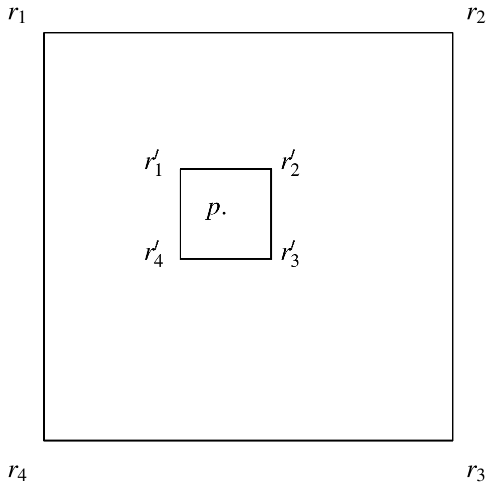


Figure 2: Illustration for Lemma 2

Proof:

We will use Figure 2 for illustration. Any line segment connecting p to a point on a side of R must intersect at least one side of the rectangle R' since all of R is entirely outside R' and p is inside R'. Suppose $r'_4r'_3$ is the only side of R' that the set of line segments connecting p to each side of R intersect. Now p is a point between parallel lines passing through r_1r_2 and $r'_4r'_3$, and a line segment connecting a point on r_1r_2 to p cannot intersect the line through $r'_4r'_3$. Thus $r'_4r'_3$ cannot be the only side of R' that the set of line segments connecting p to each side of R intersect. Hence, there must be at least two such sides, and that proves the lemma.
End of Proof.

Theorem 3 *Let C be a convex region in two dimensions with a bounding rectangle R. Assume that the centroid of R is not an integer point, and let R' be the smallest rectangle(square) with integer vertices that includes the centroid of R. If none of the vertices of R' belong to C, and R is more than 4 integer units in length and width, then C intersects exactly two sides of R'.*

Proof:

Let R, shown as $r_1r_2r_3r_4$ in Figure 3(i), be the bounding rectangle for the convex region C (not shown in the figure). R' is the rectangle shown as $r'_1r'_2r'_3r'_4$ and the centroid of R is an interior point of R'. The sides of R' are extrapolated until they meet a side of R.

Since R tightly bounds C, there is at least one point belonging to C on each side of the rectangle R. Also, there is at least one point belonging to C inside R' (centroid of R). Since C is convex, there is a line segment from a point on each side of R, to a point inside R', which is completely contained in C. From Lemma 2, C must intersect at least two sides of R'.

We now prove that C can intersect at most two sides of R'. Suppose C intersects three sides of R'. Without loss of generality, we assume that the sides are $r'_2r'_3$, $r'_3r'_4$ and $r'_4r'_1$. We represent this by drawing these sides with thick lines in Figure 3(i). In Figure 3 we will mark every line segment that we prove has a point belonging to C, to thick lines. Every line segment which is proved *not* to contain any points in C is marked by dashed lines.

We have assumed that there is at least one point on the line segment $r'_4r'_1$ that belongs to C. If any point on the line segment r'_1a belongs to C, then the point r'_1 must belong to C, since C is convex. But r'_1 is a vertex of R' and does not belong to C. Therefore, the line segment r'_1a cannot intersect C. Similarly we can show that line segments r'_1a, r'_4f, r'_2b, r'_3e, r'_4g and r'_3d cannot intersect C. These inferences are marked with dashed lines in Figure 3(ii).

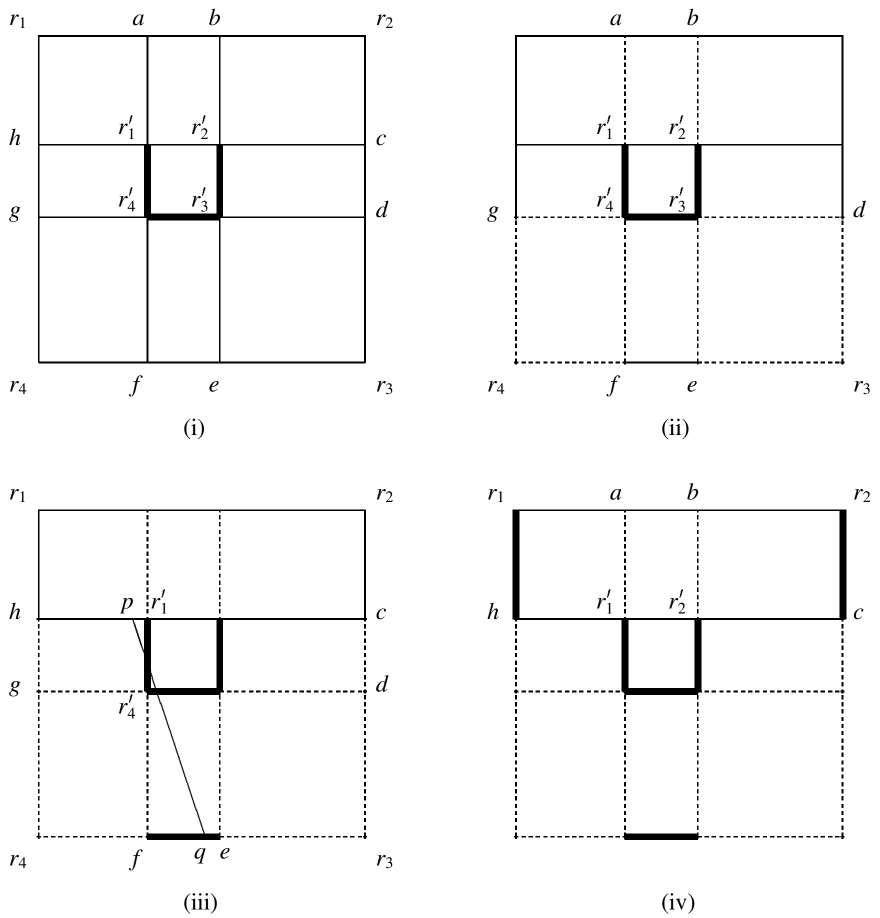


Figure 3: Illustration for Theorem 3

If there is any point of C inside the rectangle $gr'_4f\,r_4$, then a line segment from that point to a point inside R' must be contained in C. But such a line segment must intersect gr'_4 or r'_4f, and we have established that these line segments do not intersect C. So there can be no points inside the rectangle $gr'_4f\,r_4$ that belong to C. In particular, line segments $f\,r_4$ and r_4g do not intersect C. Similarly we can show that line segments dr_3 and r_3e do not contain any points belonging to C. The corresponding segments are marked as dashed lines in Figure 3(ii).

Since R tightly bounds C, there must be at least one point on r_3r_4 that is in C. Since we have proved that line segments r_3e and $f\,r_4$ do not contain any points belonging to C, the line segment ef must intersect C, and is marked as a bold line in Figure 3(iii).

Since the sides of R are more than 4 integer units long (or more than 4 times the sides of R'), the line segments gr'_4, hr'_1, r'_4f are all longer than one integer unit (or longer than the sides of R'). Also, sides of R' and line segments ef and gh are of unit length.

It can be seen that any line through a point on the line segment ef that does not intersect the line segment $f\,r'_4$ cannot intersect the line through points h and c at a point more than 1 unit left (towards h) of point r'_1, and hence cannot intersect the line segment gh (see pq in Figure 3(iii)). There cannot be points belonging to C on both line segments ef and gh, since a line segment connecting those points would have to intersect the line segment $f\,r'_4$ and we have proved that the line segment $f\,r'_4$ does not intersect C. Since we have also proved that the line segment ef has at least one point belonging to C, the line segment gh cannot intersect C. Similarly we can show that the line segment cd cannot intersect C. We reach the situation shown in Figure 3(iii).

Since the side r_4r_1 of rectangle R must have at least one point that is in C and the line segment r_4h has been shown to not have any points belonging to C, the line segment r_1h must intersect C. Similarly we can show that the line segment r_2c must intersect C (see Figure 3(iv)). Then there must be a line segment from a point on the line segment

$r_1 h$ to a point on the line segment $r_2 c$, which is contained in C. But every such line segment must intersect the line segment ar_1' (and br_2') which we have proved does not intersect C. Hence we have a contradiction.

We conclude that the convex region C cannot intersect three or more sides of R'. Since we have already shown that it must intersect at least two sides of R', the convex region C must intersect exactly two sides of R'.
End of Proof.

Furthermore, a convex region that satisfies the requirements of this theorem must be located around one of the diagonals of the bounding rectangle. Specifically we have the following theorem.

Theorem 4 *Let C, R and R' be defined as in Theorem 3. Let R_1, R_2, R_3, and R_4 be the rectangles formed by the extensions of the sides of R' and the rectangle R, as shown in Figure 4. Then under the conditions of Theorem 3, two diagonally opposite rectangles R_i intersect C, and the remaining two diagonally opposite R_i do not intersect C.*

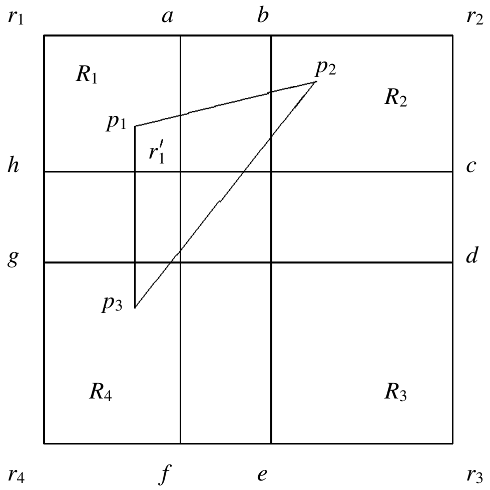


Figure 4: Illustration for Theorem 4

Proof:

We first show that at least two of the rectangles R_i intersect C. Suppose none of the rectangles R_i intersect C. Since R strictly bounds C, all sides of R intersect C. Thus C must intersect each of the line segments ab, cd, ef, and gh, since those are the only parts of the sides of R that do not belong to any of R_i. But a line segment connecting any point on ab to any point on gh must intersect R_1, which contradicts that none of the rectangles R_i intersect C. Hence there must be at least one rectangle R_i that intersects C.

Suppose R_1 is the only rectangle R_i that intersects C. Then C must intersect the line segments cd and ef since those are the only parts of the sides $r_2 r_3$ and $r_3 r_4$, respectively, that do not belong to any of the R_i assumed not to intersect C. But a line segment connecting any point on cd to any point on ef must intersect the rectangle R_3, which contradicts that R_1 is the only rectangle that intersects C. We conclude that at least two of the rectangles R_i must intersect C.

We now show that not more than two of the rectangles R_i can intersect C. Suppose three of the rectangles R_i intersect C. Let the rectangles be R_1, R_2 and R_3. Let p_1, p_2 and p_3 be points belonging to C in the rectangles R_1, R_2 and R_3 respectively. It is easy to see from Figure 4 that point r_1' is inside the triangle $p_1 p_2 p_3$, and hence inside C, which is a contradiction. Hence we can have at most two rectangles R_i that intersect C. We conclude that exactly two of the rectangles R_i intersect C.

We now show that the two rectangles R_i that intersect C must be diagonally opposite. Suppose two adjacent rectangles R_1 and R_2 intersect C. Then R_3 and R_4 cannot intersect C since exactly two of R_i intersect C. The line segment ef must intersect C since that is the only part of the side $r_3 r_4$ that is not in the rectangles R_3 or R_4. Let p_1 and p_2 be points in the rectangles R_1 and R_2 respectively that belong to C, as shown in the Figure 5. A line connecting p_1 and p_2 will intersect the line segment br_2'. Let the point of intersection be q_1. Also, since C is convex and tightly bounded by R, there must be a line segment contained in C that connects a point on the side $r_2 r_3$ to a point on the line

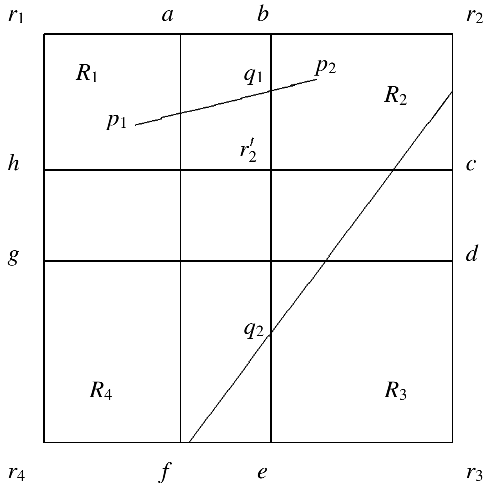


Figure 5: Illustration for Theorem 4

segment $f\,e$. Since the line segment ab is shorter than the line segment br_2, and the line segment er_2' is longer than the line segment $r_2'b$, any line segment connecting a point on the side r_2r_3 to a point on the line segment ef must intersect line segment $r_2'e$, say at point q_2. Since q_1 and q_2 belong to C and r_2' lies on the line connecting them, r_2' must belong to C. This contradicts the conditions of this theorem. Hence two adjacent R_i cannot intersect C.

Therefore, exactly two diagonally opposite R_i intersect C.

End of Proof.

Note that the two diagonally opposite rectangles that intersect with the convex region are the only regions that can potentially contain integer points, since remaining integer points in the bounding rectangle are either in the square around the centroid, or in one of the other two diagonally opposite rectangles.

The significance of the results of this section is as follows. If a convex region (larger than a certain size) contains an integer point, then there is a strong possibility that one of the integer points near the centroid of its bounding rectangle belongs to the convex region, and this can be checked. However, if none of the integer points close to the centroid belong to the convex region, and it cannot be readily shown that the convex region has no integer points, we can identify two smaller rectangles that enclose the parts of the convex region that potentially contain integer points. This leads to the following recursive procedure for searching for integer points in a convex region.

procedure: FindIntegers(C, R)

input: Convex region C defined by a set of inequalities in two variables, say x and y. Rectangle R tightly bounding C.

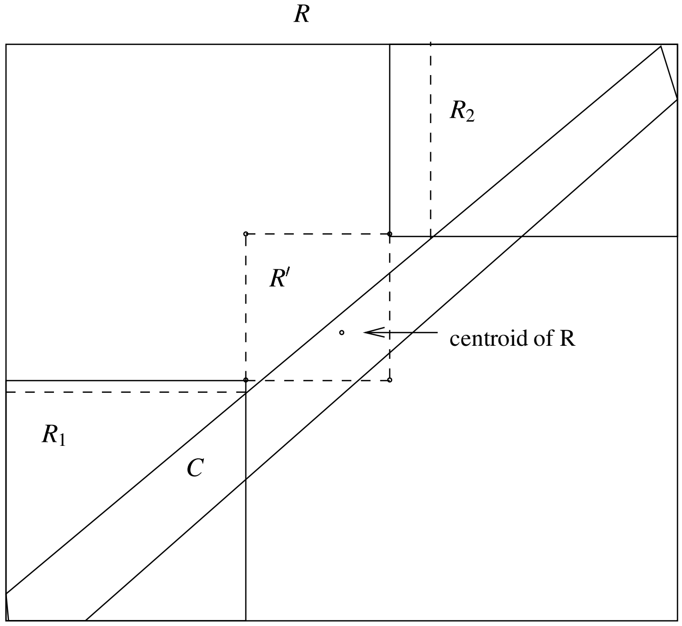
output: Boolean value representing whether C contains an integer point.

1. If R covers more than 4 integers in both dimensions, GOTO step 2. Otherwise, use a simple search as follows. Say x is the variable that can take at most 4 integer values. For each possible integer value of x, determine the range of values y can take. If an integer is found in the value range of y, return TRUE, else return FALSE.

2. Find the centroid of R. If it is an integer point, return TRUE. Let R' be the smallest square with integer points as corners enclosing the centroid of R. If any of the four corner points of R' is inside C, return TRUE. Otherwise GOTO step 3.

3. Find the two sides of R' that intersect C. Determine the two rectangles R_1 and R_2, shown as solid rectangles in Figure 6. Define C_1 and C_2 by adding the new boundaries of R_1 and R_2, respectively, to C as additional inequality constraints. Find the new strictly bounding rectangles R_1' and R_2' shown with one dashed side, for C_1 and C_2 respectively. The new bounding rectangle can be found simply by back substitution, since three of the four bounds are already determined (all except the one shown by a dashed line) and Fourier-Motzkin elimination is not necessary.

   Return (FindIntegers(C_1, R_1') *OR* FindIntegers(C_2, R_2')).



C : Convex region under consideration.

R : Bounding rectangle for C

R' : Smallest integer square enclosing the centroid of R

R_1, R_2 : Bounding rectangles for the new problem

Figure 6: Nature of convex regions with no integer points in the vicinity of the centroid

end of procedure

The worst case complexity of this procedure is $O(n)$, where n is the number of points in (or the area of) the original bounding rectangle[3]. The number of points in each recursively obtained bounding rectangle is less than 1/4 of the previous bounding rectangle, hence the height of the search tree is $O(log(n))$, but the number of nodes in the search tree is still $O(n)$, and the complexity is no better than that obtained by a naive search. However, the average behavior is much better since the recursive step is needed only in a special situation.

4.2 Searching with coordinate transformations

We observed in the last section that the presence or absence of integer solutions is easily established except in the case when the convex region is a thin strip along a diagonal of the bounding rectangle. Coordinate transformations can be used to align a such a convex region close to one of the coordinate axes and compress its bounding rectangle. We present a search procedure based on repeated coordinate transformations that takes logarithmic number of steps in the worst case.

Consider a convex region C with a bounding rectangle R that satisfies the conditions of Theorem 3, as shown in Figure 6. Let X and Y be the lengths of the sides of the bounding rectangle R along the x and y dimensions, respectively. Further assume that X and Y are at least 8 integer units in length, and that $X > Y$.

Consider the line represented by the equation:

$$x - My = 0 \qquad \text{where} \qquad M = \lfloor X/Y + 1/2 \rfloor \geq 1$$

[3]Complexity can be shown to be linear in the length of the bounding rectangle also.

The slope of this line is close to the diagonal of the bounding box R which is along the convex region C. Let δ_y be the absolute distance along the y axis between the pair of lines parallel to the line $x - My = 0$ that tightly enclose C. Then it can be shown that:

$$\delta_y < Y/2M$$

Consider the coordinate transformation represented by the following matrix:

$$\begin{bmatrix} M & M+1 \\ 1 & 1 \end{bmatrix}$$

This would transform a line $a_0x + a_1y = c$ to $(a_0M + a_1)x + (a_0(M + 1) + a_1)y = c$. We apply this transformation to all the inequalities that define the convex region C. This transformation is *unimodular* since the determinant of the transformation matrix has the value (-1), which implies that the integer points in the transformed convex region have a 1-1 correspondence with the integer points in the original convex region.

On applying the above transformation, the lines parallel to $x - My = 0$ become parallel to the line $y = 0$, the x axis. By examining how the sides of the original bounding rectangle appear in the transformed space, we infer:

1. The absolute span of the new bounding rectangle in the y dimension is at most $(\delta_y * M)$.
2. The absolute span of the new bounding rectangle in the x dimension is at most $(X/M + \delta_y)$.

Multiplying these quantities to compute the area, and using $\delta_y < Y/2M$, $X \geq Y$, and $M \geq 1$, we find that the area of the new bounding rectangle is less than or equal to $3XY/4$, that is, at most $3/4$ of the area of the original bounding rectangle. We now present a search procedure based on repeatedly checking for solutions and using coordinate transformations to shrink the search space.

procedure: FindIntegers2(C, R)
input: Convex region C defined by a set of inequalities in two variables, say x and y. Rectangle R tightly bounding C.
output: Boolean value representing whether C contains an integer point.

1. If R covers more than 8 integers in both dimensions, GOTO step 2. Otherwise use a direct search method discussed in procedure FindIntegers.
2. Find the centroid of R. If it is an integer point, return TRUE. Let R' be the smallest square with integer points as corners enclosing the centroid of R. If any of the four corner points of R' is inside C, return TRUE. Otherwise GOTO step 3.
3. Identify the diagonal along which C is aligned, and use the coordinate transformations discussed above to obtain a new convex region C'. Use Fourier-Motzkin elimination to obtain the new bounding rectangle R'.

   Return (FindIntegers2(C', R'))

end of procedure

This procedure is guaranteed to terminate in $O(log(n))$ steps, where n is the area of the original bounding rectangle, since the area is reduced by at least a factor of $3/4$ in each step. We have presented a simple coordinate transformations that ensures logarithmic behavior; there certainly can be other methods of deriving a coordinate transformation that would lead to faster convergence.

This procedure takes only logarithmic number of steps, but each step can be more expensive since a complete Fourier-Motzkin elimination is performed on a new set of inequalities. Whether the improved complexity justifies the added per step cost is not clear, and will depend on the properties of the problem instances.

4.3 Comparison with branch-and-bound

The solution method discussed in section 4.1 is a variation of the branch-and-bound method. Simple branch-and-bound can be stated as follows:

1. Select a dimension, say x, and find the range in that dimension. Fourier-Motzkin elimination can be used for this step.

2. Find the midpoint of this range and f the nearest integer to the midpoint, say x_0. Find the range in the y dimension for $x = x_0$. If there is an integer in this range, the problem is solved, else go to step 3.

3. Add constraints $x < x_0$ and $x > x_0$ separately to the original set of constraints, to generate two new problems, and solve the problem recursively.

Simple branch-and-bound has some advantages over our method. The Fourier-Motzkin elimination is applied only to find the range in one dimension, which can be a significant saving. Moreover, the method is applicable to a convex region in n dimensions.

However, there are several advantages that our method has over simple branch-and-bound. Firstly, if there is an integer point in the convex region, there is a larger chance that our method will find it at a given step in recursion, thus the number of recursive steps will be lesser on average. Alternately stated, there are fewer shapes of convex regions with integer points in them, that can escape detection with our method, as compared to a simple branch-and-bound.

Another advantage of our method is that the Fourier-Motzkin elimination is applied only in the first step. In subsequent recursive steps, the bounding rectangle is inferred directly from the previous bounding rectangle and back substitution. This can lead to a significant reduction in computation, particularly when several recursive steps are needed.

The solution procedure discussed in Section 4.2 is qualitatively different from branch-and-bound methods, and takes logarithmic rather than linear number of steps.

4.4 Approach to a general solution

We have described a solution method for finite convex regions in two dimensions. We now discuss the relaxation of these conditions.

Solution for higher dimensions:

The results obtained for 2 dimensions cannot be directly extended to n dimensions. We are exploring results in 3 or more dimensions that can be used for developing an efficient algorithm. In general, we have to fall back on a simple branch-and-bound strategy. However, intuition suggests that the integer points near the centroid are most likely to be part of the convex region, and should be tested in a branch-and-bound approach.

Handling infinite regions:

We have assumed that the ranges for all variables are finite. This may not be true in general. An infinite convex region has zero or infinite integer points. It is obvious that an infinite convex region bounded by diverging hyperplanes would include infinite integer points. A necessary condition for an infinite convex region to have no integer points is that there are parallel hyperplanes bounding the region (with an infinite thin strip with no integer points between them). We have the following lemma:

Lemma 5 *An infinite size convex region must have infinite integer points unless at least two of the hyperplanes defining it are parallel.*

For our analysis, we check if any of the hyperplanes defining the convex region are parallel. If no two hyperplanes are parallel, no further analysis is required. If parallel hyperplanes do exist, we generate heuristic values for bounds, based on the coefficients in the inequalities defining the convex region, and solve the approximate problem with constant bounds, obtained as a result.

5 Related work

Array subscript analysis for parallelizing compilers has been addressed by several research groups in the recent past. Feauterier [5] extended simplex method to integer programming for this purpose. This method is general and does not take advantage of the special characteristics of the array subscript analysis problems.

Our approach builds on the methods used in Maydan et. al. [9], Omega test [10] and Power test [14]. For most practical problem instances, any of the above methods will yield a solution rapidly. Our contribution is a fast solution procedure for a class of problems that cannot be solved easily and exactly. For the problems in this category, other methods will either yield an inexact solution, or be computationally very expensive. We do not have enough experience to judge the practical advantage of our approach relative to others, but it is simple and without any significant drawbacks.

The unique feature of our approach is the method to determine if a convex region contains any integer points. Maydan et. al. suggest using a branch and bound strategy at this stage in the solution procedure, which we compared to our method in section 4.3. The Omega test uses a quick test for identifying integer solutions, which succeeds in most practical cases, and generates a set of problems that are guaranteed to solve the problem exactly when the quick test fails. The process is bounded by the sizes of the coefficients in the problem. We employ a sophisticated search strategy based on the geometric properties of convex regions.

A pseudo-polynomial algorithm for solving two variable integer programming problems was developed by Feit [6]. The algorithm is based on the use of coordinate transformations to transform the problem of maximizing an arbitrary objective function to that of finding a point with minimal (say) x co-ordinate. We also use coordinate transformations in a similar way to obtain an asymptotically faster algorithm, but our objective is to show the existence of an integer solution, not to optimize a given objective function.

6 Conclusions

We have presented a method for determining whether a non-negative integer solution for a system of equalities and inequalities exists. The specific contribution is two related new methods to establish the existence of integer solutions in a convex region. They are based on the observation that if a convex region contains any integer points, at least one of the integer points close to the centroid usually belongs to the convex region. These methods can be combined with other known results to build an efficient system for exact dependence and synchronization analysis in parallel programs.

7 Acknowledgements

This paper has benefited considerably from the comments by the reviewers.

References

[1] ALLEN, R., AND KENNEDY, K. Automatic translation of FORTRAN programs to vector form. *ACM Transactions on Programming Languages and Systems 9*, 4 (October 1987), 491–542.

[2] BANERJEE, U. *Dependence Analysis for Supercomputing*. Kluwer Academic Publishers, 1988.

[3] CALLAHAN, D., KENNEDY, K., AND SUBHLOK, J. Analysis of event synchronization in a parallel programming tool. In *Proceedings of the Second ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming* (Seattle, WA, March 1990).

[4] DUFFIN, R. J. On Fourier's analysis of linear inequality systems. *Mathematical Programming Study 1* (1974), 71–95.

[5] FEAUTRIER, P. Parametric integer programming. *Recherche Operationnelle 22*, 3 (Sept 1988), 243–268.

[6] FEIT, S. D. A fast algorithm for the two-variable integer programming problem. *Journal of the ACM 33*, 1 (January 1984), 99–113.

[7] GARFINKEL, R., AND NEMHAUSER, G. *Integer Programming*. John Wiley and Sons, New York, 1972.

[8] KAUFMANN, A., AND HENRY-LABORDERE, A. *Integer and Mixed Programming*. Academic Press, New York, 1977.

[9] MAYDAN, D., HENNESSY, J., AND LAM, M. Efficient and exact data dependence analysis. In *Proceedings of the ACM SIGPLAN 91' Conference on Program Language Design and Implementation* (Toronto, Canada, June 1991), pp. 1–14.

[10] PUGH, W. The Omega test: a fast and practical integer programming algorithm for dependence analysis. In *Proceedings of Supercomputing '91* (Albuquerque, NM, November 1991), pp. 4–13.

[11] SMITH, H. On systems of linear indeterminate equations and congruences. *Philosophical Transactions of the Royal Society of London A*, 151 (1861), 293–326.

[12] SUBHLOK, J. Solving integer programs from dependence and synchronization problems. Tech. Rep. CMU-CS-93-130, School of Computer Science, Carnegie Mellon University, 1993.

[13] WOLFE, M. *Optimizing Supercompilers for Supercomputers*. PhD thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1982.

[14] WOLFE, M., AND TSENG, C. The power test for data dependence. *IEEE Transactions on Parallel and Distributed Systems 3*, 5 (September 1992), 591–601.