Loop parallelization using compiler analysis
Which of these loops is parallel?

- **Examples**

  ```plaintext
  FOR I = 0 to 5
  
  FOR I = 0 to 5
  
  For I = 0 to 5
  ```

How can we determine this automatically using compiler analysis?
Organization of a Modern Compiler

Source Program

Front-end
syntax analysis + type-checking + symbol table

Middle1
loop-level transformations

High-level Intermediate Representation
(loops, array references are preserved)

Middle2
conventional optimizations

Low-level Intermediate Representation
(array references converted into low level operations, loops converted to control flow)

Back-end
register allocation instruction selection

Assembly Code
**Key concepts:**

**Perfectly-nested loop:** Loop nest in which all assignment statements occur in body of innermost loop.

```plaintext
for J = 1, N
    for I = 1, N
        Y(I) = Y(I) + A(I,J)*X(J)
```

**Imperfectly-nested loop:** Loop nest in which some assignment statements occur within some but not all loops of loop nest.

```plaintext
for k = 1, N
    a(k,k) = sqrt (a(k,k))
    for i = k+1, N
        a(i,k) = a(i,k) / a(k,k)
    for i = k+1, N
        for j = k+1, i
            a(i,j) = a(i,k) * a(j,k)
```
Our focus for now: perfectly-nested loops
Iteration Space of a Perfectly-nested Loop

Each iteration of a loop nest with n loops can be viewed as an integer point in an n-dimensional space.

**Iteration space of loop**: all points in n-dimensional space corresponding to loop iterations

DO  I = 1, N  
DO  J = 1, M  
S

Execution order = lexicographic order on iteration space:

\((1, 1) \preceq (1, 2) \preceq \ldots \preceq (1, M) \preceq (2, 1) \preceq (2, 2) \ldots \preceq (N, M)\)
Intuition about systems of linear inequalities:

Equality: line (2D), plane (3D), hyperplane (>3D)

Inequality: half-plane (2D), half-space (>2D)

Region described by inequality is convex
(if two points are in region, all points in between them are in region)
Intuition about systems of linear inequalities:

Conjunction of inequalities = intersection of half-spaces
=> some convex region

Region described by inequalities is a convex polyhedron
(if two points are in region, all points in between them are in region)
Let us formulate correctness of loop permutation as ILP problem.

Intuition: If all iterations of a loop nest are independent, then permutation is certainly legal.

This is stronger than we need, but it is a good starting point.

What does independent mean?

Let us look at dependences.
Dependences in loops

\begin{verbatim}
FOR 10 I = 1, N
X(f(I))  = ...
10      = ...X(g(I))...
\end{verbatim}

• Conditions for flow dependence from iteration $I_w$ to $I_r$:
  • $1 \leq I_w \leq I_r \leq N$ (write before read)
  • $f(I_w) = g(I_r)$ (same array location)

• Conditions for anti-dependence from iteration $I_g$ to $I_o$:
  • $1 \leq I_g < I_o \leq N$ (read before write)
  • $f(I_o) = g(I_g)$ (same array location)

• Conditions for output dependence from iteration $I_{w1}$ to $I_{w2}$:
  • $1 \leq I_{w1} < I_{w2} \leq N$ (write in program order)
  • $f(I_{w1}) = f(I_{w2})$ (same array location)
Dependences in nested loops

FOR 10 I = 1, 100
FOR 10 J = 1, 200

\[ X(f(I,J),g(I,J)) = \ldots \]
\[ 10 = \ldots X(h(I,J),k(I,J)) \ldots \]

Conditions for flow dependence from iteration \((I_w, J_w)\) to \((I_r, J_r)\):
Recall: \(\leq\) is the lexicographic order on iterations of nested loops.

\[ 1 \leq I_w \leq 100 \]
\[ 1 \leq J_w \leq 200 \]
\[ 1 \leq I_r \leq 100 \]
\[ 1 \leq J_r \leq 200 \]

\[ (I_1, J_1) \leq (I_2, J_2) \]
\[ f(I_1, J_1) = h(I_2, J_2) \]
\[ g(I_1, J_1) = k(I_2, J_2) \]
Anti and output dependences can be defined analogously.
Array subscripts are affine functions of loop variables

=>

dependence testing can be formulated as a set of ILP problems
ILP Formulation

FOR $I = 1, 100$

$x(2I) = \ldots \ x(2I+1)\ldots$

Is there a flow dependence between different iterations?

$$1 \leq Iw < Ir \leq 100$$

$$2Iw = 2Ir + 1$$

which can be written as

$$1 \leq Iw$$

$$Iw \leq Ir - 1$$

$$Ir \leq 100$$

$$2Iw \leq 2Ir + 1$$

$$2Ir + 1 \leq 2Iw$$
The system

\[
\begin{align*}
1 & \leq Iw \\
Iw & \leq Ir - 1 \\
Ir & \leq 100 \\
2Iw & \leq 2Ir + 1 \\
2Ir + 1 & \leq 2Iw
\end{align*}
\]

can be expressed in the form \( Ax \leq b \) as follows

\[
\begin{pmatrix}
-1 & 0 \\
1 & -1 \\
0 & 1 \\
2 & -2 \\
-2 & 2
\end{pmatrix}
\begin{bmatrix}
Iw \\
Ir
\end{bmatrix}
\leq
\begin{bmatrix}
-1 \\
-1 \\
100 \\
1 \\
-1
\end{bmatrix}
\]
ILP Formulation for Nested Loops

FOR I = 1, 100
  FOR J = 1, 100
    X(I,J) = ..X(I-1,J+1)...

Is there a flow dependence between different iterations?

1 \leq I_w \leq 100
1 \leq I_r \leq 100
1 \leq J_w \leq 100
1 \leq J_r \leq 100

(I_w, J_w) \prec (I_r, J_r)(lexicographic order)

I_r - 1 = I_w
J_r + 1 = J_w

Convert lexicographic order \prec into integer equalities/inequalities.
\((I_w, J_w) \prec (I_r, J_r)\) is equivalent to
\(I_w < I_r \text{ OR } ((I_w = I_r) \text{ AND } (J_w < J_r))\)

We end up with two systems of inequalities:

\[
\begin{align*}
1 \leq I_w &\leq 100 & 1 \leq I_w &\leq 100 \\
1 \leq I_r &\leq 100 & 1 \leq I_r &\leq 100 \\
1 \leq J_w &\leq 100 & 1 \leq J_w &\leq 100 \\
1 \leq J_r &\leq 100 & 1 \leq J_r &\leq 100 \\
I_w < I_r & \quad \text{OR} \quad I_w = I_r \\
I_r - 1 & = I_w & J_w < J_r \\
J_r + 1 & = J_w & I_r - 1 & = I_w \\
& & J_r + 1 & = J_w
\end{align*}
\]

Dependence exists if either system has a solution.
What about affine loop bounds?

FOR I = 1, 100
   FOR J = 1, I
      X(I,J) = X(I-1,J+1)...

      1 ≤ Iw ≤ 100
      1 ≤ Ir ≤ 100
      1 ≤ Jw ≤ Iw
      1 ≤ Jr ≤ Ir

      (Iw, Jw) < (Ir, Jr) (lexicographic order)

      Ir - 1 = Iw
      Jr + 1 = Jw
We can actually handle fairly complicated bounds involving \text{min’s} and \text{max’s}.

\text{FOR I} = 1, 100
\text{FOR J} = \text{max(F1(I),F2(I)) , min(G1(I),G2(I))}
X(I,J) = \ldots X(I-1,J+1)\ldots

\text{\ldots}

\begin{align*}
F1(Ir) & \leq Jr \\
F2(Ir) & \leq Jr \\
Jr & \leq G1(Ir) \\
Jr & \leq G2(Ir) \\
\ldots
\end{align*}

\text{Caveat: F1, F2 etc. must be affine functions.}
For a given $I$, the $J$ co-ordinate of a point in the iteration space of the loop nest satisfies $\max(L_1(I),L_2(I)) \leq J \leq \min(U_1(I),U_2(I))$.

Min’s and max’s in loop bounds may seem weird, but actually they describe general polyhedral iteration spaces!
More important case in practice: variables in upper/lower bounds

FOR I = 1, N
    FOR J = 1 , N-1
        ....

Solution: Treat N as though it was an unknown in system

\[
1 \leq Iw \leq N \\
1 \leq Jw \leq N - 1
\]

....

This is equivalent to seeing if there is a solution for any value of N.

Note: if we have more information about the range of N, we can easily add it as additional inequalities.
Summary

Problem of determining if a dependence exists between two iterations of a perfectly nested loop can be framed as ILP problem of the form

Is there an integer solution to system $Ax \leq b$?

How do we solve this decision problem?
Presentation sequence:

- one equation, several variables
  \[ 2x + 3y = 5 \]

- several equations, several variables
  \[ 2x + 3y + 5z = 5 \]
  \[ 3x + 4y = 3 \]

- equations & inequalities
  \[ 2x + 3y = 5 \]
  \[ x \leq 5 \]
  \[ y \leq -9 \]

Diophantine equations:
use integer Gaussian elimination

Solve equalities first
then use Fourier-Motzkin elimination
One equation, many variables:

Thm: The linear Diophantine equation \( a_1 x_1 + a_2 x_2 + \ldots + a_n x_n = c \) has integer solutions iff \( \gcd(a_1,a_2,\ldots,a_n) \) divides \( c \).

Examples:

(1) \( 2x = 3 \) \hspace{1em} No solutions

(2) \( 2x = 6 \) \hspace{1em} One solution: \( x = 3 \)

(3) \( 2x + y = 3 \)

\( \gcd(2,1) = 1 \) which divides 3.

Solutions: \( x = t \), \( y = (3 - 2t) \)

(4) \( 2x + 3y = 3 \)

\( \gcd(2,3) = 1 \) which divides 3.

Let \( z = x + \text{floor}(3/2) \), \( y = x + y \)

Rewrite equation as \( 2z + y = 3 \)

Solutions: \( z = t \) \hspace{1em} \( x = (3t - 3) \)

\( y = (3 - 2t) \) \hspace{1em} \( y = (3 - 2t) \)

Intuition: Think of underdetermined systems of eqns over reals.
Caution: Integer constraint => Diophantine system may have no solns
Thm: The linear Diophantine equation \( a_1 x_1 + a_2 x_2 + \ldots + a_n x_n = c \) has integer solutions iff \( \text{gcd}(a_1, a_2, \ldots, a_n) \) divides \( c \).

Proof: WLOG, assume that all coefficients \( a_1, a_2, \ldots, a_n \) are positive.

We prove only the IF case by induction, the proof in the other direction is trivial. Induction is on \( \min(\text{smallest coefficient, number of variables}) \).

Base case:

If \( \# \text{ of variables} = 1 \), then equation is \( a_1 x_1 = c \) which has integer solutions if \( a_1 \) divides \( c \).

If \( \text{smallest coefficient} = 1 \), then \( \text{gcd}(a_1, a_2, \ldots, a_n) = 1 \) which divides \( c \).

Wlog, assume that \( a_1 = 1 \), and observe that the equation has solutions of the form \( (c - a_2 t_2 - a_3 t_3 - \ldots - a_n t_n, t_2, t_3, \ldots t_n) \).

Inductive case:

Suppose smallest coefficient is \( a_1 \), and let \( t = x_1 + \text{floor}(a_2/a_1) x_2 + \ldots + \text{floor}(a_n/a_1) x_n \)

In terms of this variable, the equation can be rewritten as

\[
(a_1) t + (a_2 \mod a_1) x_2 + \ldots + (a_n \mod a_1) x_n = c \tag{1}
\]

where we assume that all terms with zero coefficient have been deleted.

Observe that (1) has integer solutions iff original equation does too.

Now \( \text{gcd}(a, b) = \text{gcd}(a \mod b, b) \Rightarrow \text{gcd}(a_1, a_2, \ldots, a_n) = \text{gcd}(a_1, (a_2 \mod a_1), \ldots, (a_n \mod a_1)) \)

\[
\Rightarrow \text{gcd}(a_1, (a_2 \mod a_1), \ldots, (a_n \mod a_1)) \text{ divides } c.
\]

If \( a_1 \) is the smallest co-efficient in (1), we are left with 1 variable base case.

Otherwise, the size of the smallest co-efficient has decreased, so we have made progress in the induction.
Summary:

Eqn: \[ a_1 x_1 + a_2 x_2 + \ldots + a_n x_n = c \]

- Does this have integer solutions?
- Does gcd(a_1, a_2, \ldots, a_n) divide c?
Systems of Diophantine Equations:

**Key idea:** use integer Gaussian elimination

**Example:**

\[
\begin{align*}
2x + 3y + 4z &= 5 \\
x - y + 2z &= 5
\end{align*}
\]

\[
\begin{bmatrix}
2 & 3 & 4 \\
1 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
= 
\begin{bmatrix}
5 \\
5
\end{bmatrix}
\]

It is not easy to determine if this Diophantine system has solutions.

**Easy special case:** lower triangular matrix

\[
\begin{bmatrix}
1 & 0 & 0 \\
-2 & 5 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
= 
\begin{bmatrix}
5 \\
5
\end{bmatrix}
\]

\[
\begin{align*}
x &= 5 \\
y &= 3 \\
z &= \text{arbitrary integer}
\end{align*}
\]

**Question:** Can we convert general integer matrix into equivalent lower triangular system?

**INTEGER GAUSSIAN ELIMINATION**
Unimodular Column Operations:

(a) Interchange two columns

\[
\begin{pmatrix}
2 & 3 \\
6 & 7
\end{pmatrix}
\quad \rightarrow
\begin{pmatrix}
3 & 2 \\
7 & 6
\end{pmatrix}
\]

Check
Let \(x,y\) satisfy first eqn.
Let \(x',y'\) satisfy second eqn.
\(x' = y\), \(y' = x\)

(b) Negate a column

\[
\begin{pmatrix}
2 & 3 \\
6 & 7
\end{pmatrix}
\quad \rightarrow
\begin{pmatrix}
2 & -3 \\
6 & -7
\end{pmatrix}
\]

Check
\(x' = x\), \(y' = -y\)

(c) Add an integer multiple of one column to another

\[
\begin{pmatrix}
2 & 3 \\
6 & 7
\end{pmatrix}
\quad \rightarrow
\begin{pmatrix}
2 & 1 \\
6 & 1
\end{pmatrix}
\]

Check
\(n = -1\)
\(x = x' + n y'\)
\(y = y'\)
Example:

\[
\begin{bmatrix}
2 & 3 & 4 \\
1 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
= 
\begin{bmatrix}
5 \\
5
\end{bmatrix}
\]

\[
\begin{bmatrix}
2 & 3 & 4 \\
1 & -1 & 2
\end{bmatrix}
\Rightarrow 
\begin{bmatrix}
2 & 3 & 0 \\
1 & -1 & 0
\end{bmatrix}
\Rightarrow 
\begin{bmatrix}
2 & 1 & 0 \\
1 & -2 & 0
\end{bmatrix}
\Rightarrow 
\begin{bmatrix}
0 & 1 & 0 \\
5 & -2 & 0
\end{bmatrix}
\Rightarrow 
\begin{bmatrix}
1 & 0 & 0 \\
-2 & 5 & 0
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x' \\
y' \\
z'
\end{bmatrix}
= 
\begin{bmatrix}
5 \\
5
\end{bmatrix}
\Rightarrow 
\begin{bmatrix}
x' = 5 \\
y' = 3 \\
z' = t
\end{bmatrix}
\Rightarrow 
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
= 
\begin{bmatrix}
-1 & 3 & -2 \\
1 & -2 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
5 \\
3 \\
t
\end{bmatrix}
= 
\begin{bmatrix}
4-2t \\
-1 \\
t
\end{bmatrix}
\]
Systems of Inequalities
Goals:

Given system of inequalities of the form $Ax \leq b$

- determine if system has an integer solution
- enumerate all integer solutions
Running example:

\[3x + 4y \geq 16 \quad (1)\]
\[4x + 7y \leq 56 \quad (2)\]
\[4x - 7y \leq 20 \quad (3)\]
\[2x - 3y \geq -9 \quad (4)\]

Upper bounds for \(x\): (2) and (3)
Lower bounds for \(x\): (1) and (4)

Upper bounds for \(y\): (2) and (4)
Lower bounds for \(y\): (1) and (3)
MATLAB graphs:
Code for enumerating integer points in polyhedron: (see graph)

Outer loop: Y, Inner loop: X

DO Y=[4/37], [74/13]
    DO X=[max(16/3 - 4y/3, -9/2 + 3y/2)], [min(5 + 7y/4, 14 - 7y/4)]
    ....

Outer loop: X, Inner loop: Y

DO X=1, 9
    DO Y=[max(4 - 3y/4, (4x - 20)/7)], [min(8 - 4x/5, (2x + 9)/3)]
    ....

How do we can determine loop bounds?
Fourier-Motzkin elimination: variable elimination technique for inequalities

\[ \begin{align*}
3x + 4y & \geq 16 \\
4x + 7y & \leq 56 \\
4x - 7y & \leq 20 \\
2x - 3y & \geq -9
\end{align*} \tag{5-8} \]

Let us project out \( x \).
First, express all inequalities as upper or lower bounds on \( x \).

\[ \begin{align*}
x & \geq 16/3 - 4y/3 \\
x & \leq 14 - 7y/4 \\
x & \leq 5 + 7y/4 \\
x & \geq -9/2 + 3y/2
\end{align*} \tag{9-12} \]
For any $y$, if there is an $x$ that satisfies all inequalities, then every lower bound on $x$ must be less than or equal to every upper bound on $x$.

Generate a new system of inequalities from each pair (upper,lower) bounds.

\[
\begin{align*}
5 + \frac{7y}{4} & \geq \frac{16}{3} - \frac{4y}{3} \quad \text{(Inequalities3, 1)} \\
5 + \frac{7y}{4} & \geq -\frac{9}{2} + \frac{3y}{2} \quad \text{(Inequalities3, 4)} \\
14 - \frac{7y}{4} & \geq \frac{16}{3} - \frac{4y}{3} \quad \text{(Inequalities2, 1)} \\
14 - \frac{7y}{4} & \geq -\frac{9}{2} + \frac{3y}{2} \quad \text{(Inequalities2, 4)}
\end{align*}
\]
Simplify:

\[
\begin{align*}
y & \geq \frac{4}{37} \\
y & \geq -38 \\
y & \leq \frac{104}{5} \\
y & \leq \frac{74}{13}
\end{align*}
\]

\[\Rightarrow\]

\[
\begin{align*}
\max(\frac{4}{37}, -38) & \leq y \leq \min(\frac{104}{5}, \frac{74}{13}) \\
\Rightarrow \quad \frac{4}{37} & \leq y \leq \frac{74}{13}
\end{align*}
\]

This means there are rational solutions to original system of inequalities.
We can now express solutions in closed form as follows:

\[ \frac{4}{37} \leq y \leq \frac{4}{37} \]

\[ \max \left( \frac{16}{3} - \frac{4y}{3}, -\frac{9}{2} + \frac{3y}{2} \right) \leq x \leq \min \left( 5 + \frac{7y}{4}, 14 - \frac{7y}{4} \right) \]
**Fourier-Motzkin elimination:** iterative algorithm

**Iterative step:**
- obtain reduced system by projecting out a variable
- if reduced system has a rational solution, so does the original

**Termination:** no variables left

**Projection along variable** $x$: Divide inequalities into three categories

- $a_1 \cdot y + a_2 \cdot z + \ldots \leq c_1 (no \ x)$
- $b_1 \cdot x \leq c_2 + b_2 \cdot y + b_3 \cdot z + \ldots (upper \ bound)$
- $d_1 \cdot x \geq c_3 + d_2 \cdot y + d_3 \cdot z + \ldots (lower \ bound)$

New system of inequalities:
- All inequalities that do not involve $x$
- Each pair (lower, upper) bounds gives rise to one inequality:

$$b_1 [c_3 + d_2 \cdot y + d_3 \cdot z + \ldots] \leq d_1 [c_2 + b_2 \cdot y + b_3 \cdot z + \ldots]$$
Theorem: If \((y_1, z_1, \ldots)\) satisfies the reduced system, then 
\((x_1, y_1, z_1\ldots)\) satisfies the original system, where \(x_1\) is a rational number between 
\[
\min (1/b_1(c_2 + b_2 y_1 + b_3 z_1 + \ldots), \ldots) \quad \text{(over all upper bounds)}
\]
and 
\[
\max (1/d_1(c_3 + d_2 y_1 + d_3 z_1 + \ldots), \ldots) \quad \text{(over all lower bounds)}
\]
Proof: trivial
What can we conclude about integer solutions?

**Corollary:** If reduced system has no integer solutions, neither does the original system.

**Not true:** Reduced system has integer solutions $\Rightarrow$ original system does too.

Key problem: Multiplying one inequality by $b_1$ and other by $d_1$ is not guaranteed to preserve ”integrality” (cf. equalities)

**Exact projection:** If all upper bound coefficients $b_i$ or all lower bound coefficients $d_i$ happen to be 1, then integer solution to reduced system implies integer solution to original system.
More accurate algorithm for determining existence

Just because there are integers between $4/37$ and $74/13$, we cannot assume there are integers in feasible region.

However, if gap between lower and upper bounds is greater than or equal to 1 for some integer value of $y$, there must be an integer in feasible region.
Dark shadow: region of $y$ for which gap between upper and lower bounds of $x$ is guaranteed to be greater than or equal to 1.

Determining dark shadow region:

Ordinary FM elimination:

$$x \leq u, \ x \geq l \implies u \geq l$$

Dark shadow:

$$x \leq u, \ x \geq l \implies u \geq l + 1$$
For our example, dark shadow projection along $x$ gives system

\[
\begin{align*}
5 + 7y/4 & \geq 16/3 - 4y/3 + 1 (\text{Inequalities3, 1}) \\
5 + 7y/4 & \geq -9/2 + 3y/2 + 1 (\text{Inequalities3, 4}) \\
14 - 7y/4 & \geq 16/3 - 4y/3 + 1 (\text{Inequalities2, 1}) \\
14 - 7y/4 & \geq -9/2 + 3y/2 + 1 (\text{Inequalities2, 4})
\end{align*}
\]

$\Rightarrow$

\[
66/13 \geq y \geq 16/37
\]

There is an integer value of $y$ in this range $\Rightarrow$ integer in polyhedron.
More accurate estimate of dark shadow

For integer values of $y_1,z_1,...$, there is no integer value $x_1$ between lower and upper bounds if

$$\frac{1}{d_1}(c_3+d_2y_1+d_3z_1+... - \frac{1}{b_1}(c_2+b_2y_1+b_3z_1+...) + \frac{1}{b_1} + \frac{1}{d_1} \leq 1$$

This means there is an integer between upper and lower bounds if

$$\frac{1}{d_1}(c_3+d_2y_1+d_3z_1+... - \frac{1}{b_1}(c_2+b_2y_1+b_3z_1+...) + \frac{1}{b_1} + \frac{1}{d_1} > 1$$

To convert this to $\geq$, notice that smallest change of lhs value is $1/b_1d_1$.

So the inequality is

$$\frac{1}{d_1}(c_3+d_2y_1+d_3z_1+... - \frac{1}{b_1}(c_2+b_2y_1+b_3z_1+...) + \frac{1}{b_1} + \frac{1}{d_1} \geq 1 + \frac{1}{b_1d_1}$$

$$\Rightarrow$$

$$\frac{1}{d_1}(c_3+d_2y_1+d_3z_1+... - \frac{1}{b_1}(c_2+b_2y_1+b_3z_1+...) \geq (1 - \frac{1}{b_1})(1 - \frac{1}{d_1})$$
Note: If \((b_1 = 1)\) or \((d_1 = 1)\), dark shadow constraint = real shadow constraint
Example:

\[ 3x \geq 16 - 4y \]
\[ 4x \leq 20 + 7y \]

**Real shadow:** \((20 + 7y) \cdot 3 \geq 4(16 - 4y)\)

**Dark shadow:** \((20 + 7y) \cdot 3 - 4(16 - 4y) \geq 12\)

**Dark shadow (improved):** \((20 + 7y) \cdot 3 - 4(16 - 4y) \geq 6\)
What if dark shadow has no integers?

There may still be integer points nested closely between an upper and lower bound.
One enumeration idea: splintering

Scan the corners with hyperplanes, looking for integer points.

Generate a succession of problems in which each lower bound is replaced with a sequence of hyperplanes. How many hyperplanes are needed?

Equation for lower bound: \( x = \frac{1}{b_1(c_2+b_2y+b_3z+...)} \)

Hyperplanes:
\[
\begin{align*}
  x &= \frac{1}{b_1(c_2+b_2y+b_3z+...)} \\
  x &= \frac{1}{b_1(c_2+b_2y+b_3z+...)} + \frac{1}{b_1} \\
  x &= \frac{1}{b_1(c_2+b_2y+b_3z+...)} + \frac{2}{b_1} \\
  x &= \frac{1}{b_1(c_2+b_2y+b_3z+...)} + \frac{3}{b_1} \\
  \vdots \\
  x &= \frac{1}{b_1(c_2+b_2y+b_3z+...)} + 1 \quad \text{(in dark shadow region; if this is integer, so is)}
\end{align*}
\]
Summary

• Two integer linear programming problems
  – Is there an integer point within a polyhedron $Ax \leq b$ where $A$ is an integer matrix and $b$ is an integer vector?
    • used for dependence analysis
  – Enumerate the integer points within a polyhedron $Ax \leq b$ where $A$ is an integer matrix and $b$ is an integer vector.
    • used for code generation after affine loop transformation