ALPA LECTURE 4: POLYHEDRAL MODEL (1)

Outline

- Motivation
- Notation: polyhedra, parametric polyhedra, recurrence equations
- Affine control loops (ACLs)
- More notation: lexicographic order, program-execution order, piecewise affine functions, lex-max of piecewise affine functions
- Exact data-flow analysis of ACLs

Motivation

Main Objective: construct a parallel (PRAM) program (from a given sequential program). In general, very hard (impossible).

Solution: restrict to (ACLs). Question: Why ACLs?

- Useful class – many applications dominated by ACLs
- Powerful tools to deal with ACLs (polyhedral model)

Hurdles of Automatic Parallelization

In sequential code:

- Reductions are serialized
- Memory is reused (false dependences)

For ACLs, the polyhedral model can resolve both problems by:

- Exact data-flow analysis (to disambiguate memory writes). It transforms ACLs into SAREs (Systems of Affine Recurrence Equations)
- Detection of reductions and scans

Parallelization of ACLs

- Analyze the ACL to get a polyhedral SARE
- Determine two functions for each value computed by the original program:
  - A time instant, i.e., a schedule
  - A place (where it is computed, and where the result is stored) i.e., a (memory and processor) allocation.
- Use these functions to transform the SARE into a new SAREs were some indices denote time, some others denote processors and the rest represent memory addresses
- From the transformed SARE generate parallel code that respects the schedule, exploits the parallelism of the processors, and allocates and utilizes the specified memory

Notation: Parameterized Polyhedral Domains

- Integral Polyhedron: (finite or infinte) set of points satisfying a finite number of linear inequalities

$$\{z \in \mathcal{Z}^n \mid Qz \geq q\}$$

- Parameterized family of polyhedra: the q may include a linear function of m size parameters, p

$$\{z \in \mathcal{Z}^n \mid Qz \geq q - Pp\}$$

Alternatively:

$$\left\{\begin{pmatrix} z \\ p \end{pmatrix} \in \mathcal{Z}^{n+m} \mid \begin{bmatrix} Q & P \end{bmatrix}\begin{pmatrix} z \\ p \end{pmatrix} \geq q\right\}$$

Dual Representation of Polyhedra

- Generators: Set of convex combinations of a set of vertices

$$\{z \in \mathcal{Z}^n \mid z = a^T G;\ \sum_i a_i = 1\}$$

- Each Representations is useful in certain contexts.

Standard (self dual) algorithm to convert from one to other.

Examples

1. $\mathcal{P}_1 = \{i, j, n \mid 0 \leq j \leq i; j \leq n; 1 \leq n\}$: a 3-D polyhedron with 2 vertices and 3 rays
2. $\mathcal{P}_1$ can also be viewed as a 2-D polyhedron (parameterized by n). It has 2 vertices, $\{[0,0],[n,0]\}$ and a ray, $[1,0]$.
3. $\mathcal{P}_2 = \{i, j, n, m \mid 0 \leq j \leq i \leq n; j \leq m; 1 \leq n, m\}$: a 2-D polytope (no rays) parameterized by n and m.

   What are its vertices?

Answer

Depends on the relative values of of n and m: either a triangle (if $n \leq m$) or a trapezium (otherwise).

$$\begin{cases} \{1 \leq n \leq m\} & \Rightarrow \quad \{[0,0],[n,0],[n,n]\} \\ \{1 \leq m < n\} & \Rightarrow \quad \{[0,0],[n,0],[n,m],[m,m]\} \end{cases}$$

Set of vertices of a parameterized polytope (polyhedron) is a piecewise affine function of the parameters

Recurrence Equations: Definition

$$X[z] = \{\, z \in D \,\} : \quad \Rightarrow \quad g(\ldots X[f(z)]\ldots)$$

- X n-dimensional data variable;
- z n-dimensional index variable.
- g strict, ("atomic") function;
- $f(z)$ *dependency function*, $f : \mathcal{Z}^n \rightarrow \mathcal{Z}^n$;
- "..." — other such arguments;
- $D \subseteq \mathcal{Z}^n$, domain of the equation.

Recurrence Equations: Taxonomy

- **Uniform Recurrence Equations** (URE) — $f(z)$ has the form $z + \delta$.
- **Affine Recurrence Equations** (ARE) — $f(z)$ is an affine function, $Az + a$.
- **System of Recurrence Equations** (SRE) — set of (mutually recursive) equations. Could be uniform or affine (SURE or SARE).

More Notation: ACLs

- Only control structure is the `for` loop
- Lower (cf. upper) bounds of loops are affine expressions of surrounding loops, or max (cf. min) of such expressions
- Body of a loop is either (i) a loop, or (ii) an assignment statement, or (iii) a sequence of (any of) these two
- Program variables: either index-vars, or params, or (multidimensional) array variables (scalars are zero-dimensional arrays)
- Assignment statement: lhs variable is an array var, accessed by an affine function of indices (and params); rhs is an expression containing such (indexed) array vars.

Example of ACL: Forward Substitution

```
S1:     x[1] := b[1]/a[1,1];
        for i = 2 to n do
S2:         s := 0;
            for j = 1 to i-1 do
S3:             s := s + x[j] * a[i, j];
            enddo
S4:         x[i] := (b[i] - s) / a[i,i]
        enddo
```

In an ACL

- each assignment statement is given a unique label, S_i, and
- represents many instances of operarations,
- one instance for each valid value of surrounding indices.
- set of valid values is a polyhedron D_i (parameterized by program params)
- Each operation (i.e., each value computed by the program is uniquely identified by $\langle S_i, z \rangle$ for $z \in D_i$
- For two statements, S_i and S_j, define n_{ij} as the number of common loop indices

Even more notation

- lexicographic order, $\prec$
- program execution order, $\triangleleft$

$$\langle S_1, z_1 \rangle \triangleleft \langle S_2, z_2 \rangle \equiv \begin{cases} z_1' \preceq z_2' & \text{if } S_1 \text{ appears \textbf{before} } S_2 \text{ in pgm txt} \\ z_1' \prec z_2' & \text{sinon} \end{cases}$$

where z_1' and z_2' are respectively the first n_{ij}-th elements of z_1 and z_2

- Lmax (cf. Lmin) is the lexicographic maximum (cf. minimum) of two or more vectors.
- Remark: Lmax (or Lmin) of all points in a polytope is one of its vertices, i.e., a piecewise affine function of its params.
- Remark: Lmax (cf. Lmin) of two or more piecewise affine functions is also a piecewise affine function.

So what's all this machinery for?

Exact data-flow analysis: ACL $\Rightarrow$ SARE

- Variables of the SARE = statements of the ACL
- Variable domains = domains of the statements
- Consider an ACL:

$$\mathsf{S_i}: \quad \mathtt{X}[\mathtt{f}_\mathtt{i}^0(\mathtt{z})] = \mathtt{g}(\ldots \; \mathtt{Y}[\mathtt{f}_\mathtt{i}^\mathtt{k}(\mathtt{z})] \ldots)$$

- Key question:

  Which instance of which statement wrote the value of Y read by the z-th instance of S_i?

Solution

- Must be some statement that has Y on the left – a list of potential candidates. For each candidate of the form $\mathsf{S_j}: \; \mathtt{Y}[\mathtt{f}_\mathtt{j}^0(\mathtt{z}')] = \ldots$, the solution must satisfy at least:

$$\left\| \begin{array}{lcl} z & \in & D_i \\ z' & \in & D_j \\ f_i^k(z,p) & = & f_j^0(z',p) \\ \langle S_j, z' \rangle & \triangleleft & \langle S_i, z \rangle \end{array} \right. \qquad (1)$$

  Among all such points, we need to take the lexicographic max

- Eqn (1) is a union of polyhedra
- Overall solution is a piecewise affine function of z and the program params

Forward Substitution

Find the source of s in S3

Two statements write into s, $\mathsf{Src}(\mathtt{s},\mathtt{S3}) = \mathsf{Last}(\langle \mathtt{S2}, f_1(i,j)\rangle, \langle \mathtt{S3}, f_2(i,j)\rangle)$, where

$$f_1(i,j) = \mathsf{Last}\left\{(i' \mid i,j) \left\| \begin{array}{l} i' \in \mathcal{D}_2 \\ (i,j) \in \mathcal{D}_3 \\ \langle \mathtt{S2}, i'\rangle \lhd \langle \mathtt{S3}, (i,j)\rangle \end{array}\right.\right\}$$

$$f_2(i,j) = \mathsf{Last}\left\{(i',j' \mid i,j) \left\| \begin{array}{l} (i',j') \in \mathcal{D}_3 \\ (i,j) \in \mathcal{D}_3 \\ \langle \mathtt{S3}, (i',j')\rangle \lhd \langle \mathtt{S3}, (i,j)\rangle \end{array}\right.\right\}$$

Now, $\langle \mathtt{S2}, i'\rangle \lhd \langle \mathtt{S3}, (i,j)\rangle$ iff $i' \leq i$.

Hence

$$\begin{aligned} f_1(i,j) &= \mathsf{Last}\left\{(i' \mid i,j) \left\| \begin{array}{l} i' \in \mathcal{D}_2 \\ (i,j) \in \mathcal{D}_3 \\ i' \leq i \end{array}\right.\right\} \\ &= \mathsf{Last}\left\{(i' \mid i,j) \;\|\; 2 \leq i' \leq i \leq n; 1 \leq j < i \leq n\right\} \\ &= i \end{aligned}$$

For $f_2(i,j)$, $\langle \mathtt{S3}, (i',j')\rangle \lhd \langle \mathtt{S3}, (i,j)\rangle$ is equivalent to $i > i' \vee (i' = i \wedge j' < j)$, a disjunction of two sets of inequality constraints.

Hence,

$$\begin{aligned} f_2(i,j) &= \mathsf{Last}\left(\left\{(i',j' \mid i,j) \left\| \begin{array}{l} (i',j') \in \mathcal{D}_3 \\ (i,j) \in \mathcal{D}_3 \\ i' < i \end{array}\right.\right\} \bigcup \left\{(i',j' \mid i,j) \left\| \begin{array}{l} (i',j') \in \mathcal{D}_3 \\ (i,j) \in \mathcal{D}_3 \\ i' = i; j' < j \end{array}\right.\right\}\right) \\ &= \mathsf{Last}\left(\mathsf{Last}\left\{(i',j' \mid i,j) \left\| \begin{array}{l} (i',j') \in \mathcal{D}_3 \\ (i,j) \in \mathcal{D}_3 \\ i' < i \end{array}\right.\right\}, \mathsf{Last}\left\{(i',j' \mid i,j) \left\| \begin{array}{l} (i',j') \in \mathcal{D}_3 \\ (i,j) \in \mathcal{D}_3 \\ i' = i; j' < j \end{array}\right.\right\}\right) \\ &= \mathsf{Last}\left(\mathsf{Last}\{(i',j' \mid i,j) \| 1 \leq j' < i' < i \leq n; 1 \leq j < i \leq n\} \right. \\ &\qquad \left. \mathsf{Last}\{(i',j' \mid i,j) \| 1 \leq j < i' = i \leq n; j' < j\}\right) \end{aligned}$$

This is $\mathsf{Last}(f_2'(i,j), f_2''(i,j))$

where

$$f_2'(i,j) = \left\{ \begin{array}{lcl} \{i,j | i=2; j=1\} & : & \perp \\ \{i,j | 1 \leq j < i \leq n; i \geq 3\} & : & (i-1, i-2) \end{array}\right.$$

$$f_2''(i,j) = \left\{ \begin{array}{lcl} \{i,j | 1 = j < i \leq n\} & : & \perp \\ \{i,j | 1 < j < i \leq n\} & : & (i, j-1) \end{array}\right.$$

and hence

$$f_2(i,j) = \begin{cases} \{i,j \mid i=2; j=1\} & : \mathsf{Last}(\perp,\perp) \\ \{i,j \mid i \geq 3; j=1\} & : \mathsf{Last}((i-1,i-2),\perp) \\ \{i,j \mid 1 < j < i \leq n\} & : \mathsf{Last}((i-1,i-2),(i,j-1)) \end{cases}$$
$$= \begin{cases} \{i,j \mid i=2; j=1\} & : \perp \\ \{i,j \mid j=1; i \geq 3\} & : (i-1,i-2) \\ \{i,j \mid 1 < j < i \leq n\} & : (i,j-1) \end{cases}$$

Finally, remember that

$$\mathsf{Src}(\mathtt{s},\mathtt{S3}) = \mathsf{Last}\left(\langle \mathtt{S2}, f_1(i,j)\rangle, \langle \mathtt{S3}, f_2(i,j)\rangle\right)$$
$$= \mathsf{Last}\left(\langle \mathtt{S2}, i\rangle, \left\langle \mathtt{S3}, \begin{cases} \{i,j \mid i=2; j=1\} & : \perp \\ \{i,j \mid j=1; i \geq 3\} & : (i-1,i-2) \\ \{i,j \mid 1 < j < i \leq n\} & : (i,j-1) \end{cases} \right\rangle\right)$$
$$= \mathsf{Last}\left(\langle \mathtt{S2}, i\rangle, \begin{cases} \{i,j \mid i=2; j=1\} & : \langle \mathtt{S3},\perp\rangle \\ \{i,j \mid j=1; i \geq 3\} & : \langle \mathtt{S3},(i-1,i-2)\rangle \\ \{i,j \mid 1 < j < i \leq n\} & : \langle \mathtt{S3},(i,j-1)\rangle \end{cases}\right)$$

$$= \begin{cases} \{i,j \mid i=2; j=1\} & : \mathsf{Last}(\langle \mathtt{S2}, i\rangle, \langle \mathtt{S3},\perp\rangle) \\ \{i,j \mid j=1; i \geq 3\} & : \mathsf{Last}(\langle \mathtt{S2}, i\rangle, \langle \mathtt{S3},(i-1,i-2)\rangle) \\ \{i,j \mid 1 < j < i \leq n\} & : \mathsf{Last}(\langle \mathtt{S2}, i\rangle, \langle \mathtt{S3},(i,j-1)\rangle) \end{cases}$$

$$= \begin{cases} \{i,j \mid i=2; j=1\} & : \langle \mathtt{S2}, i\rangle \\ \{i,j \mid j=1; i \geq 3\} & : \langle \mathtt{S2}, i\rangle \\ \{i,j \mid 1 < j < i \leq n\} & : \langle \mathtt{S3},(i,j-1)\rangle \end{cases}$$

$$= \begin{cases} \{i,j \mid 1 = j < i \leq n\} & : \langle \mathtt{S2}, i\rangle \\ \{i,j \mid 1 < j < i \leq n\} & : \langle \mathtt{S3},(i,j-1)\rangle \end{cases}$$

Exercise: Backward substitution

```
S1:     x[n] := b[n]/u[n,n];
        for i = n-1 down to 1 do
S2:         s := 0;
            for j = i+1 to n do
S3:             s := s + x[j] * u[i, n-j];
            enddo
S4:         x[i] := (b[i] - s) / u[i,i]
        enddo
```