PART IV:

Extensions, Limitations & Conclusion

Why does ALPHA work?

ALPHA domains (unions of finitely many polyhedra) constitute an abstract data type (ADT), closed under:

- Intersection
- Union (of finitely many members of the ADT)
- Preimage by (arbitrary) affine functions (the class of dependencies in ALPHA).
- Image by unimodular affine functions (the class of transformations used for changes of bases).

Extension to Sparse Domains

Develop sparse polyhedra as such an ADT, show that these properties hold, and modify the the syntax, and tools for transformations, normalization, etc.

Mathematical details: $\mathcal{Z}$-Polyhedra

Def 1 (Integral polyhedron, IP): $D \equiv \{x \in \mathcal{Z}^n \mid Px \leq q\}$

Def 2 (Lattice, L:) Integral affine combination of linearly independent (*generating*) vectors in $\mathcal{Q}^n$. **Integral Lattice, IL:** generating vectors are integral.

Def 3 ($\mathcal{Z}$-Polyhedron:) Intersection of IP & (integral) lattice

Def 4 (Linearly Bounded Lattice, LBL:) Image of IP under an affine map, $\mathcal{M}$.

$$\boxed{\boxed{\text{IP} \subset \mathcal{Z}\text{-Polyhedra} \subset \text{LBL}}}$$

Representing $\mathcal{Z}$-Polyhedra

Lemma 1: $\mathcal{Z}$-Polyhedra are precisely that subset of LBLs for which the map $\mathcal{M}$ is square and non-singular.

Representation: image of a polyhedron by a square, non-singular affine map.

Canonic form: (to deal with multiple representations) A $\mathcal{Z}$-Polyhedra is uniquely specified, in a chosen unimodular basis, B, by

- $F_2 = \begin{pmatrix} H & 0 \\ 0 & Id \end{pmatrix}$ where H is in Hermite normal form; and
- P_2 a polyhedron which is equal to its integral hull

Closure properties of $\mathcal{Z}$-Polyhedra

- Intersection
- Unions (of finitely many Z-Polyhedra)
- Image by (invertible) rational afine functions
- Preimage by (arbitrary) rational affine functions

New Change of Basis & Normalization

- Change of Basis by pseudo-unimodular functions – not all non-singular rational transformations are allowed as a change of basis.

  Pseudo-unimodularity is a simple test on the transformation matrix and the canonic representation.

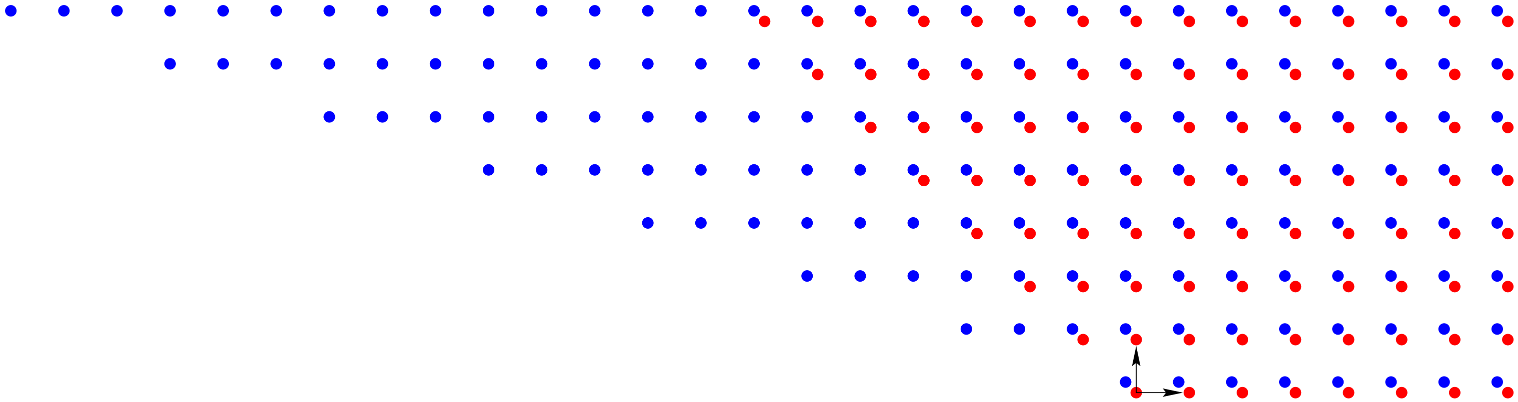
- Normalization All except one rule caries over verbatim .

Example: a multirate decimation filter

$$Y(i,j) = \begin{cases} i+j \geq 0, j = N-1 & \Rightarrow \quad w_j * X(3i,j) \\ i+j \geq 0, 0 \leq j < N-1 & \Rightarrow \quad Y(i-1,j+1) \\ & \qquad + w_j * X(3i,j) \end{cases} \tag{1}$$

$$X(i,j) = \begin{cases} -3N+3 \leq i < -2N+2; & \\ \qquad j = N-1 & \Rightarrow \quad 0 \\ -2N+2 \leq i, j = N-1 & \Rightarrow \quad x_{i+2N-2} \\ i+3j \geq 0, 0 \leq j < N-1 & \Rightarrow \quad X(i-2,j+1) \end{cases} \tag{2}$$

Domains of X and Y



After Dilating the Domain of X

system $decim : \{N \mid N \geq 1\}$
$\quad (\; x \;: \{i \mid i \geq 0\}$ **of real**)
$\quad\;\; w \;: \{i \mid N > i \geq 0\}$ **of real**)
returns $(\; y \;: \{i \mid i \geq 0\}$ **of real**) ;
var
$X \;: \{i, j \mid 0 \leq i + 3j \;;\; 0 \leq j \leq N-1\}$ **of real** ;
$Y \;: \{3i, j \mid 0 \leq i + j \;;\; 0 \leq j \leq N-1\}$ **of real** ;
let
$X =$ (∗ unchanged ∗)
$Y =$ **case**
$\quad \{3i, j \mid j = N-1\} \;: w.(i, j \rightarrow j) * X$;
$\quad \{3i, j \mid j < N-1\} : Y.(3i, j \rightarrow 3i-3, j+1) + w.(i, j \rightarrow j) * X$;
esac ;
$y = Y.(i \rightarrow 3i, N-1)$;
tel

Limitations of the polyhedral model

- Irregularities in the calculations
- Dynamic dependencies: example $A[i, j] = g(\ldots B[X[i, j]] \ldots)$
- Dynamic iteration spaces/domains
- Non linear scheduling (time as a partial order)
- Non linear allocations

Ongoing Work

- Partitioning
- Optimization of non linear criteria: number of processors, area, memory
- Dynamic dependencies
- Hierarchical systems of recurrences
- Links with everything else : compilers, conventional VLSI synthesis tools, FPGAs, . . .
- Efficient libraries

Conclusion

- Polyhedral model: increasing body of knowledge
- Connection between parallelizing compilers and architecture synthesis
- A framework for scheduling analysis
- A framework for allocation mapping
- Application to static analysis of programs