Multidimensional Synchronous Dataflow

Praveen Murthy
Dept. of EECS
UC Berkeley

Joint work with Edward A. Lee
Synchronous Dataflow

Properties
- Flow of control is predictable at compile time
- Schedule can be constructed once and repeatedly executed
- Suitable for synchronous multirate signal processing
Balance equations:

\[ r_1 O_1 = r_2 I_2 \]
\[ r_2 O_2 = r_3 I_3 \]

Solve for the smallest integers \( r_i \).

Then schedule according to data dependencies until repetitions \( r_i \) have been met for all actors.

The balance equations have no solution if the graph is *inconsistent*. For example:
Balance equations:

\[ r_{A,1} O_{A,1} = r_{B,1} I_{B,1} \]
\[ r_{A,2} O_{A,2} = r_{B,2} I_{B,2} \]

Solve for the smallest integers \( r_{X,i} \), which then give the number of repetitions of actor \( X \) in dimension \( i \).

Higher dimensionality follows similarly.
Example of Multidimensional Dataflow

\[ (40, 48) \rightarrow (8, 8) \rightarrow (8, 8) \]

\[ r_{A, 1} = r_{A, 2} = 1 \]
\[ r_{DCT, 1} = 5, \quad r_{DCT, 2} = 6 \]
Awkwardness of Using SDF for MD Systems

2D FFT by row-column decomposition

Image \rightarrow 2D FFT Galaxy \rightarrow Image Viewer

2-D FFT in MD SDF

2D FFT Galaxy

1D FFT Star (256,1) \rightarrow (1,256) \rightarrow 1D FFT Star (1,256)

Column FFTs \rightarrow Row FFTs

1 Particle holding a 256x256 matrix

Image \rightarrow 2D FFT Star \rightarrow Image Viewer

No data parallelism

1 Particle holding a 256x256 matrix

Image \rightarrow 2D FFT Star \rightarrow Image Viewer

Too many extraneous actors

1x256 \rightarrow Vectors to Image \rightarrow 1D FFT Star \rightarrow Vectors to Image \rightarrow Transpose

256x256 \rightarrow Transpose \rightarrow 256x256

256x256 \rightarrow 1D FFT Star \rightarrow Vectors to Image \rightarrow 1x256

256x256 \rightarrow Image to Vectors \rightarrow 1x256

256x256

1x256

256x256

UNIVERSITY OF CALIFORNIA AT BERKELEY
Suppose we want data exchanged in the following order:

- 1-D SDF has no compact, scalable representation of this.

Multidimensional dataflow solves this problem.

1-D SDF has no compact, scalable representation of this. Multidimensional dataflow solves this problem.
More Flexible Data Exchange in MDSDF

Dataflow graph

Precedence graph
Example: Multilayer Perceptron

Dataflow graph

Precedence graph

A
B
C
D
E

A_1,1
A_2,1
A_{a,1}
B_{1,b}
C_{1,1}
C_{2,1}
C_{c,1}

... etc.

... etc.
MDSDF Example in Ptolemy

Generalize streams to multidimensional partial orderings for representing multidimensional operations.
Generalization to Arbitrary Lattices

- MDSDF handles only rectangularly sampled signals.
- GMDSDSF handles signals on arbitrary lattices, without sacrificing compile-time schedulability.
Uses of Non-rectangular Systems

Non-rectangular systems are used in a variety of contexts:

- 2:1 interlaced TV (NTSC) [Dub85][ManCorMia93].
- Directional decompositional filterbanks [Bam90].
- Digital TV with FCO and quincunx sampling [KovVet93].
- Filterbanks for interlaced to progressive conversion [VetKovLeG90].
- Array signal processing with hexagonal geometries [DudMer84].
- Filter design techniques for non-rectangular lattices [AnsLee91][EvaMcc94].
Non-rectangular Sampling

Rectangular sampling

\[
V = \begin{bmatrix}
2 & 0 \\
0 & 1
\end{bmatrix}
\]

Non-rectangular sampling

\[
V = \begin{bmatrix}
1 & -1 \\
1 & 2
\end{bmatrix}
\]

Definition: The set of all sample points given by \( \hat{t} = V\hat{n} \), \( \hat{n} \in \mathbb{R} \) is called the lattice generated by \( V \). It is denoted \( \text{LAT}(V) \).
The fundamental parallelepiped, denoted by $FPD(V)$, is the set of points given by $Vx$ where $x = [x_1, x_2]^T$ with $0 \leq x_1, x_2 < 1$.

Definition: The set of integer points in $FPD(V)$ is denoted as $N(V)$.

Lemma: $J(V) = |N(V)| = |det(V)|$ for an integer matrix $V$.

$L = \begin{bmatrix} 2 & -2 \\ 3 & 2 \end{bmatrix}$.
Multidimensional Decimators

M-D decimation is given by the relationship:

\[ y(\hat{n}) = x(\hat{n}), \hat{n} \in LAT(V_I M) \]

where \( x \) is defined on the points \( V_I k \), \( V_I \) being the sampling matrix.

Decimation ratio:

\[ |\text{det}(M)| \]
Multidimensional Expanders

M-D expander:

\[ y(n) = \begin{cases} 
  x(n) & n \in LAT(V_I) \\
  0 & \text{otherwise}
\end{cases} \forall n \in LAT(V_IL^{-1}) \]

where \( x \) is defined at the points \( V_Ik \), \( V_I \) being the sampling matrix.

Rectangular expansion

Non-rectangular expansion

Renumbered samples from the expanders output

Expansion ratio:

\[ |\det(L)| \]
Genarlized MDSDF (GMDSDF): Sources

**Definition:** The **containability condition:** let $X$ be a set of integer points in $\mathbb{R}^m$. We say that $X$ satisfies the containability condition if there exists an $m \times m$ matrix $W$ such that $N(W) = X$.

**Definition:** We will assume that any source actor in the system produces data in the following manner. A source $S$ will produce a set of samples $\zeta$ on each firing such that each sample in $\zeta$ will lie on the lattice $\text{LAT}(V_S)$. We assume that the renumbered set $\bar{\zeta}$ satisfies the containability condition.

\[
V_S = \begin{bmatrix} 1 & 1 \\ 2 & -2 \end{bmatrix}
\]

$\bar{\zeta} = \{ V_S^{-1}x : x \in \zeta \}$, $\bar{\zeta} = N(Q)$
Concise Problem Statement

**MDSDF**
- Rectangular lattice
- Regions of data produced = rectangular arrays
- Rectangular arrays specified concisely by tuples of produced/consumed.
- Coordinate axes for dataflow along arcs orthogonal to each other (x and y axes).

**GMDSDF**
- Arbitrary lattice
- Regions of data produced = parallelograms
- Parallelograms specified concisely as the set of integer points inside a support matrix.
- Coordinate axes for dataflow along arcs not necessarily orthogonal.
**Support Matrices**

Want to describe regions where the data is contained.

- In MDSDF, these are ordinary arrays
- In the extension, these are *support matrices*.

**Theorem:**

For the decimator, 

\[ V_f = V_e M \quad \text{and} \quad W_f = M^{-1} W_e. \]

For the expander, 

\[ V_f = V_e L^{-1} \quad \text{and} \quad W_f = L W_e. \]
Semantics of GMDSDF

\[ V_{SA} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad L = \begin{bmatrix} 2 & -2 \\ 3 & 2 \end{bmatrix}, |L| = 5 \times 2 \quad M = \begin{bmatrix} 1 & 1 \\ 2 & -2 \end{bmatrix}, |M| = 2 \times 2 \]

\[ W_{SA} = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \]

A consumes (1,1) and produces (5,2).

B consumes (2,2) and produces (1,1) on average.

T consumes (1,1)
• We don’t know yet exactly how many samples on each firing the decimator will produce.
• Idea: *Assume* that it produces (1,1) and solve balance equations:

\[
3r_{S,1} = 1r_{A,1} \\
5r_{A,1} = 2r_{B,1} \\
r_{B,1} = r_{T,1} \\
3r_{S,2} = 1r_{A,2} \\
2r_{A,2} = 2r_{B,2} \\
r_{B,2} = r_{T,2}
\]

• Solution:

\[
 r_{S,1} = 2, r_{S,2} = 1 \\
r_{A,1} = 6, r_{A,2} = 3 \\
r_{B,1} = 15, r_{B,2} = 3 \\
r_{T,1} = 15, r_{T,2} = 3
\]
Dataspace on arc AB

2x2 rectangle consumed by decimator

-2 -1 0 1 2

4 3 2 1

- Original samples produced by source
- Samples retained by decimator
- Samples added by expander, discarded by decimator
**Balance equations cont’d**

**Question:** Have we really “balanced”? 

**No:** by counting the number of samples that have been kept in the previous slide.

More systematically:

\[
W_{SA} = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} r_{S,1} & 0 \\ 0 & r_{S,2} \end{bmatrix} = \begin{bmatrix} 3r_{S,1} & 0 \\ 0 & 3r_{S,2} \end{bmatrix}
\]

\[
W_{AB} = LW_{SA} = \begin{bmatrix} 2 & -2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 3r_{S,1} & 0 \\ 0 & 3r_{S,2} \end{bmatrix} = \begin{bmatrix} 6r_{S,1} & -6r_{S,2} \\ 9r_{S,1} & 6r_{S,2} \end{bmatrix}
\]

\[
W_{BT} = M^{-1}W_{AB} = \frac{1}{4} \begin{bmatrix} 2 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} 6r_{S,1} & -6r_{S,2} \\ 9r_{S,1} & 6r_{S,2} \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 21r_{S,1} & -6r_{S,2} \\ 3r_{S,1} & -18r_{S,2} \end{bmatrix}
\]
Want to know if

$$|N(W_{BT})| = \frac{|N(W_{AB})|}{|M|}$$

We have

$$|N(W_{AB})| = |det(W_{AB})| = 90r_{S,1}r_{S,2}$$

The right hand side becomes

$$\frac{90r_{S,1}r_{S,2}}{4} = \frac{45r_{S,1}r_{S,2}}{2}$$

Therefore, we need

$$r_{S,1}r_{S,2} = 2k \quad k = 0, 1, 2, \ldots$$

The balance equations gave us $r_{S,1} = 2, r_{S,2} = 1$.

With these values, we get

$$W_{BT} = \begin{bmatrix} 21/2 & -3/2 \\ 3/2 & -9/2 \end{bmatrix}.$$ 

This matrix has 47 points inside its FPD (determined by drawing it out).

$$=\Rightarrow$$ Balance equation solution is not quite right.
Augmented Balance Equations

To get the correct balance, take into account the constraint given by:

\[ |N(W_{BT})| = \frac{|N(W_{AB})|}{|M|} \]

**Sufficiency**: force \( W_{BT} \) to be an integer matrix.

\[ \Rightarrow r_{S,1} = 4k, \quad k = 1, 2, \ldots \]
\[ \Rightarrow r_{S,2} = 2k, \quad k = 1, 2, \ldots \]

Therefore,

\[ r_{S,1} = 4, \quad r_{S,2} = 2. \]

- So decimator produces (1,1) on average but has cyclostatic behavior.

Production sequence: \( 2,1,1,2,1,0,1,1,0,1,2,1,1,2,1,\ldots \)

**Theorem:**

Always possible to solve these augmented balance equations.
Effect of Different Factorizations

Suppose we let $|det(M)| = 1 \times 4$ instead. Balance equations give:

$r_{S, 1} = 1, r_{S, 2} = 2$
$r_{A, 1} = 3, r_{A, 2} = 6$
$r_{B, 1} = 15, r_{B, 2} = 3$
$r_{T, 1} = 15, r_{T, 2} = 3$

Also,

$W_{BT} = \begin{bmatrix} 21/4 & -3 \\ 3/4 & -9 \end{bmatrix}$

It turns out that

$|N(W_{BT})| = 45$

as required.

$\Rightarrow$ Lower number of overall repetitions with this factoring choice.
Dataspace on Arc AB

1x4 rectangle consumed by decimator

- Original samples produced by source
- { samples retained by decimator
- Samples added by expander, discarded by decimator

-2 -1 0 1 2
4 3 2 1
Summary of Extended Model

- Each arc has associated with it a lattice-generating matrix, and a support matrix.
- The source actor for an arc establishes the ordering of the data on that arc.
- Expander: consumes (1,1) and produces $FPD(L)$, ordered as a $(L_1, L_2)$ rectangle where $L_1 L_2 = |det(L)|$.
- Decimator: consumes an $(M_1, M_2)$ rectangle, where $M_1 M_2 = |det(M)|$ and produces (1,1) on average.
- Write down balance equations.
- Additional equations for support matrices on decimator outputs.
- The above two sets are simultaneously solved to determine the smallest non-zero number of times each node is to be invoked in a periodic schedule.
- Actors are then scheduled as in SDF or MDSDF.
**Aspect Ratio Conversion**

Format conversion of 2:1 interlaced video from 4/3 aspect ratio to 16/9 aspect ratio.
**Future Work in GMDSDF**

### Concrete Data Structures (Semantics)

- \( \text{c0} \)
- \( \text{c1} \)
- \( (\text{c0,0})(\text{c1,0}) \rightarrow \text{c2} \)
- \( (\text{c0,1}) \rightarrow \text{c2} \)

### GMDSDF (Scheduling)

Array-oriented language (graphical syntax for enabling rules ?)
Concrete Data Structures

- “Cells” can have specific “Values”
- Enabling relationship says when a cell can be filled.
- “Cell” dependency partial order can be arbitrary
- Formalizes most forms of “real-world” data structures: lists, trees, arrays etc.
- Kahn-Plotkin sequential functions on CDS provide an elegant model of computation with many formal properties, like full abstraction.
- CDS approach has been mostly semantic; need to sort out operational issues (like scheduling).
Array-OL

- Array-oriented language developed at Thomson
- Graphical syntax for specifying “array access patterns”
  - In many multidimensional programs, manipulating data aligned in various dimensions is a challenge. For example: Transpose.
  - Patterns specified by “fitting” and “paving” relationships.
- Combine with MDSDF...
**Conclusion**

- MDSDF extension allows modeling of MD DSP systems using rectangular sampling schemes.
- GMDSDF allows modeling of MD DSP systems using arbitrary sampling schemes.
- Both models can be scheduled statically—thus ideally suited for prototyping.
- Integration of AOL concepts, along with CDS generalization might result in a very powerful MoC for multidimensional programming.