SOFTWARE SYNTHESIS

• Generation of executable code from data-flow graphs: single-processor schedules
• Used for:
  – Production software
  – Simulation software
• Based on following paper (all examples are taken from it):

TOPICS

• Synchronous data flow (recap)
• Cyclo-static data flow
• Optimization criteria
• Vectorized schedules

SYNCHRONOUS DATA FLOW (SDF)

• Already discussed.
• Each firing of a node consumes a fixed number of tokens and produces a fixed number of tokens (these numbers are annotated along the edges).
• An edge can have delay (initial tokens).
• Consistency:
  – The repetitions vector (relative number of invocations for each node) should exist.
  – There should be no deadlock (situation where nodes are waiting for each other to produce tokens).

CONSISTENT SDF EXAMPLE
EXAMPLE OF SDF WITH DEADLOCK

- Easiest check for deadlock: simulation

\[ nD \text{ on an edge means, } n \text{ initial tokens.} \]

4D on edge AB removes deadlock.

CYCLO-STATIC DATA FLOW (CSDF)

- Static data flow in which computation cycles through phases.
- Production and consumption rates on an edge are replaced by vectors, with elements for each phase.

CSDF FOR RESOURCE SHARING

\[ y_n = k(ky_{n-1} + x_n) + x_{n-1} \]

- Graph has 2 multiplications
- They are “scheduled” on the same actor.

CSDF FOR MEMORY REDUCTION

- CSDF exposes opportunities for buffer-memory reduction
OPTIMIZATION CRITERIA

- Buffer memory
- Code memory
- Number of context switches

IMPLEMENTATION

- Inlined code
- Subroutines
- Hybrid

MINIMAL-BUFFER SCHEDULE

\[ S_1 = YZYZYYZYXYZYZYZYZYZYZ \]

- Buffer size: \( \text{buf}(S_1) = 11 \)
- Code size: \( \text{c.size}(S_1) = \kappa(X) + 10\kappa(Y) + 10\kappa(Z) \)
- Context switches: \( \text{c.sw}(S_1) = 21 \)

MINIMAL-CODE-SIZE SCHEDULE (1)

\[ S_2 = (5YZ)X(5YZ) \]

- Buffer size: \( \text{buf}(S_2) = 11 \)
- Code size: \( \text{c.size}(S_2) \approx \kappa(X) + 2\kappa(Y) + 2\kappa(Z) \)
- Context switches: \( \text{c.sw}(S_2) = 21 \)

\[ S_3 = X(10Y)(10Z) \]

- Buffer size: \( \text{buf}(S_3) = 25 \)
- Code size: \( \text{c.size}(S_3) \approx \kappa(X) + \kappa(Y) + \kappa(Z) \)
- Context switches: \( \text{c.sw}(S_3) = 3 \)
MINIMAL-CODE-SIZE SCHEDULE (2)

\[ S_4 = X(10YZ) \]

- Buffer size: \( \text{buf}(S_4) = 16 \)
- Code size: \( c_{\text{size}}(S_4) \approx \kappa(X) + \kappa(Y) + \kappa(Z) \)
- Context switches: \( c_{\text{SW}}(S_4) = 21 \)

VECTORIZATION

- Can reduce the number of actor activations per iteration
- One iteration:
  \[ N_{\text{act}}((2(2B)(5A))(5C)) = 5 \]
- Two iterations:
  \[ N_{\text{act}}((4(2B)(5A))(10C)) = 4.5 \]

OPTIMAL VECTORIZATION WITHOUT MINIMAL APPEARANCE

\[ N_{\text{act}}(BAC) = 3 \]
\[ N_{\text{act}}((4B)(4A)(4C)) = \frac{3}{4} = 0.75 \]
\[ N_{\text{act}}((4B)(4A)(3B)(3A)(7C)) = \frac{5}{7} = 0.71 \]