DEFINITIONS -

SDF Graph – a collection of nodes and edges

Token – an element of data from a stream. These are stored in buffers, consumed by nodes from edges, and produced from nodes into edges

Node – a function that consumes 0 or more tokens from each of its input edges on each firing and produces 0 or more tokens on each of its out edges on each firing.

Edge - a buffer that connects two nodes. It is of a finite size that is determined by a static scheduling algorithm. An edge can start pre-loaded with a certain number of tokens before each firing.

Delay – the tokens that an edge starts with when the SDF graph is initialized

Cycle – An execution of the network so that:

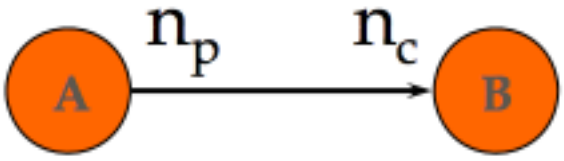
1. Every node has fired at least once
2. Every node had sufficient input tokens when it fired
3. Each edge has the same number of tokens as it started with (more formally: the number of tokens in each edge's buffer equals the number of tokens in the initial delays)

Firing Vector – v_s – an integer vector that describes the number of times each node in the SDF graph fires per cycle

Token Consumption Rate (input rate in rigel paper) – n_c - the number of tokens placed by a node A onto an outgoing edge on each of its firings.

Token Production Rate (output rate in rigel paper) - n_p - the number tokens taken by node B from an incoming edge each of its firings.

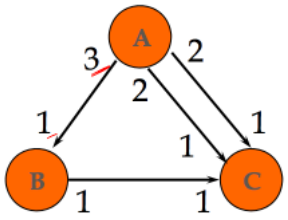
Each (edge, producer node, consumer node) triple has the following structure:



$v_s(A)$ states how many times node A fires in a cycle. $v_s(B)$ states how many times node B fires in a cycle.

How to create an SDF schedule?

1. Establish firing rates by "balancing the equations", or finding v_s such that M v_s=0



topology matrix

$$M = \begin{vmatrix} 3 & -1 & 0 \\ 0 & 1 & -1 \\ 2 & 0 & -1 \\ 2 & 0 & -1 \end{vmatrix}$$

- Balance for each edge:
  - $3\,v_S(A) - v_S(B) = 0$
  - $v_S(B) - v_S(C) = 0$
  - $2\,v_S(A) - v_S(C) = 0$
  - $2\,v_S(A) - v_S(C) = 0$

the (c, r)th entry in the matrix is the amount of data produced by node c on arc r each time it is involved

SDF theorem – For n actors (n nodes), iff the rank of M is n-1, then there will be a periodic schedule solution, aka unique smallest integer solution for v_s

2. Finding the schedule:

Not exactly sure, but it is essentially

1. Pick any edge that has enough input to fire, fire that one
2. Repeat 1 until every edge has fired the number of times specified in v_s
3. Add delay nodes as necessary to make buffers have sufficient nodes

Problem – it appears that all SDF graphs, including the ones used in rigel, consume a constant amount of data every time they fire. They do not have variable token consumption/production rates. Even rigel, which claims to have multi-rate firing, seems to have constant firings that they make look like multiple rates. (Ex: devectorize fires once where it reads in a single vector of n elements and emits n elements separately on the output edge. The paper calls this multi-rate firing, as it is claimed that this is n firings where a vector is read in on the first and 1 element is emitted on every firing. However, its more inline with the SDF literature to call it single rate firing and Rigel must be inline with the literature since it uses the classic SDF approach for solving for rates.) However, for some of our stuff, I though we have variable token consumption/production rates. For example, some of our complex shifting and stenciling reads in multiple elements differently depending on where we are in the cycle and image, right? I'll think more about this.

Graphs from - https://www.react.uni-saarland.de/teaching/embedded-systems-10-11/downloads/lec11-2010-W.pdf