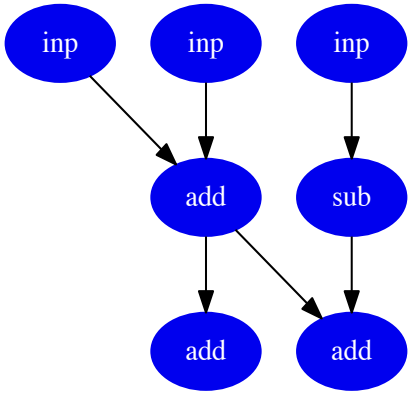
CSE 598
Algorithms for Synthesis and Optimization of Digital Systems

Programming Project 1
Scheduling of DFG

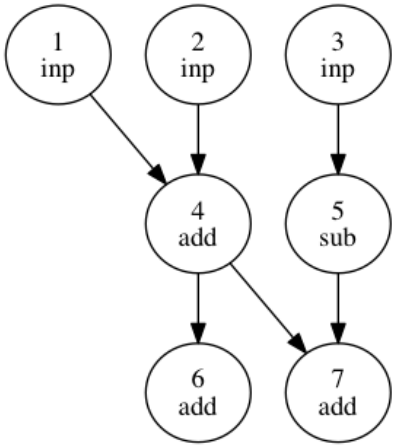
Input Files

- Data Flow Graph File (X_dfg.dot) (http://www.imm.dtu.dk/SoC-Mobinet/modules/HLS/benchmarks/index.html)
- This file will be given in the .dot format. http://www.graphviz.org/Documentation/dotguide.pdf See example below.
- The dot format is used by graphviz – one of the most extraordinary software tools for drawing a graph. It will be well worth your time to visit this page and learn about Graphviz: www.graphviz.org. Download graphviz and supply it an input file in dot format and see what happens.
- The reason I am using the .dot format is to eliminate the tedious task of reading the input and constructing a graph. NetworkX reads the .dot format. So all you'll need to do is focus on the algorithm instead of struggling with parsing the input. For example, all you need to do is the following in python:
  - Import networkx as nx
  - G = nx.DiGraph(nx.read_dot('filename.dot'))
- You can easily make your algorithms produce the output in .dot format using Networkx functions.

```
digraph test {
   node [fontcolor=white,style=filled,color=blue2];
   1 [label = inp];
   2 [label = inp];
   3 [label = inp];
   4 [label = add];
   5 [label = sub];
   6 [label = add];
   7 [label = add];
   1 -> 4 [name = 1 ];
   2 -> 4 [name = 2 ];
   4 -> 6 [name = 3 ];
   4 -> 7 [name = 4 ];
   3 -> 5 [name = 5 ];
   5 -> 7 [name = 6 ];
}
```




```
digraph test {
   node [fontcolor=black];
   1 [label = "1\ninp"];
   2 [label = "2\ninp"];
   3 [label = "3\ninp"];
   4 [label = "4\nadd"];
   5 [label = "5\nsub"];
   6 [label = "6\nadd"];
   7 [label = "7\nadd"];
   1 -> 4 [name = 1 ];
   2 -> 4 [name = 2 ];
   4 -> 6 [name = 3 ];
   4 -> 7 [name = 4 ];
   3 -> 5 [name = 5 ];
   5 -> 7 [name = 6 ];
}
```



Input Files

- **Resource Definition File (X_rdf.txt)**

Example

Name	Operation	Latency	δ_0	Area	Area
multiplier	mul	30	15	29196	17
adder	add	20	10	3077	2
subtractor	sub	11	6	3077	2
comparator	com	20	20	1771	1

This file describes the resources and their characteristics. In general, a resource can perform more than one operation, and its delay depends on the operation. The latency is given in some *time units*. δ_0 is the initiation interval – how often a new input can be processed by the resource.

For the present assignment you can assume that there is only one distinct resource for a given operation, e.g. there is only one unit that performs multiplication, one that performs addition, one that performs subtraction, etc.

With multiple resources associated with a given operation, the problem of resource selection enters into the problem. We'll deal with that later.

Programming Assignment 1 – Scheduling of a DFG

In this project, you are to design and implement several procedures in Python for scheduling DFGs.

1. MLRC (minimum latency, resource constrained) using both the ***list based scheduling heuristic*** **and the exact solution using ILP**. Experiment with combinations of resource constraints to see tradeoffs between numbers of resources vs latency.

2. MRLC (minimum resource, latency constrained) using both the ***list based scheduling heuristic*** **and** the **exact solution using ILP**. Experiment with different values of latency to see what the minimum resources would be for different bounds on latency

i. You will need to write routines to implement ASAP, ALAP, determine the mobility of each node, etc. In addition, the *list based scheduler* must examine different ways of selecting nodes to schedule, based on various measures of *urgency*.

ii. A couple of test dfgs in dot format will be provided.

iii. Provide a comparison of these two approaches in terms of the performance (optimality and run times) of the heuristic vs the ILP solutions.

Inputs are X_rdf.txt, X_dfg.dot, and result should be a scheduled DFG with each operation assigned to a time step. The output file should be another dot file, clearly showing the graph partitioned into time steps, and labeled with the mobility intervals.

Programming Assignment 1 – Scheduling of a DFG

What is to be submitted:

1. A brief narrative that describes the problems, e.g. what are MRLC and MLRC, the assumptions, the inputs, the constraints, the objectives, and the results. This is a succinct description of your understanding of the algorithms.
2. Description of the algorithms, in pseudo-code, and the collection of results on the benchmark results. A comparative analysis of the results, e.g. comparing the ILP vs List scheduling, with results shown in plots and/or tables. Here your creativity, writing and presentation skills will be evaluated. A completely working solution, but a poor report, will receive lower scores.
3. A short user manual of how to run your python script. Input files, output files, results, etc. should be described. Example of how to run the programs, inputs and expected outputs.
4. Single file with Python source code. In addition to all other functions and class definitions, there should be 4 top-level functions that perform MLRC and MRLC. These should called MLRC_List, MLRC_ILP, MRLC_List, MRLC_ILP.

   - Your source code should be thoroughly and clearly documented. Use the documentation features (docstrings) within Python to document each function. Look at the the source code of some of the packages you use. Also below are a few sources for you to read. Once you develop the habit of writing good documentation, programming becomes a pleasure. A suggestion: *Write the description of the function first, before you write the code. Describe the input arguments (types and restrictions, if any), describe the return values, check all inputs before using them, and check outputs before exit. Then write the steps of the code. Try to limit the size of each function to a few lines.*

   1. *http://sphinxcontrib-napoleon.readthedocs.org/en/latest/example_google.html*
   2. *https://docs.python.org/devguide/documenting.html*
   3. *http://docs.python-guide.org/en/latest/writing/documentation/*